Advances in Service Robotics

Advances in Service Robotics

Edited by
Ho Seok Ahn

In-Teh

Preface

Industrial robots for working in factory environment were widely researched and lead enormous development in the 20th century. But the research subjects are moving to Service Robotics with the busy life style of humans in the 21st century. Humans have great concern in healthy life and do not want to get 3D jobs (difficult, dangerous and dirty) as well as repeated simple jobs. For these reasons, Service Robots which do these jobs instead of humans are the main focus of research nowadays.

As Service Robots perform their jobs in the same environment as humans, Service Robots should have essential abilities humans have. They should recognize faces, gestures, characters, objects, speech and atmosphere. They should find their way to reach the goal without collisions and destructions, and accomplish the task at hand successfully. They should grab and deliver some objects. They should communicate with humans based on emotion. These all research subjects are included in Service Robotics area.

This book consists of 18 chapters about current research results of service robots. Topics covered include various kinds of service robots, development environments, architectures of service robots, Human-Robot Interaction, networks of service robots and basic researches such as SLAM, sensor network, etc.

This book has some examples of the research activities on Service Robotics going on around the globe, but many chapters in this book concern advanced research on this area and cover interesting topics. Therefore I hope that all who read this book will find lots of helpful information and be interested in Service Robotics. I am really appreciative of all authors who have invested a great deal of time to write such interesting and high quality chapters.

Editor
Ho Seok Ahn
Perception and Intelligence Lab.
Seoul National University
Republic of Korea
E-mail: hoseoka@gmail.com

Contents

Intelligent Unmanned Store Service Robot "Part Timer"

Ho Seok Ahn*, In-Kyu Sa**, Young Min Baek* and Jin Young Choi*
Seoul National University, Samsung Electronics Co.**
Republic of Korea

1. Introduction

In the 21st century, life for humans being has become busy to the extent that they strive for a comfortable and easy life. Service robots can provide this comfort to humans by doing all their difficult and dirty work. In order to provide this service, robots should be able to work in the same environment as humans. Service robots should have intelligent abilities such as greeting, conversation with humans, moving while avoiding obstacles, grabbing objects, etc (Sakai K. et al., 2005). However, service robots can just walk or run slowly (Riezenman, M.J., 2002), recognize little behavior (Waldherr S., 1998), and work in simple or limited environment until now. Therefore it is essential for service robots to have communication and recognition abilities that can be used in any kind of situation to help humans in a real environment. Lots of studies have focused on these abilities of service robots and many system architectures for intelligent service robots are introduced.

Mumolo has proposed the algorithm based on Prosodic model that processes vocal interaction using natural language (Mumolo E. et al., 2001). Kleinehagenbrock has introduced the system architecture of intelligent robots based on agent for humans robot interaction and have applied it to real robot (Kleinehagenbrock M. et al., 2004). Hyung-Min Koo has proposed new software architecture that changes its functions and compositions according to situation (Hyung-Min Koo et al., 2005). Kanda has proposed the system architecture that performs more than 100 behaviors with simple voice (T. Kanda et al., 2002). Mikio Nakano has proposed the two-layer model for behavior and dialogue planning in conversational service robots (Nakano, M. et al., 2005). Gluer has proposed the system architecture for context based exception handling in the case of an unexpected situation (Gluer D., 2000).

The expectation of the service robot market is getting bigger and lots of researchers are interested in developing intelligent service robots recently (Sakai K. et al., 2005). ApriAlpha is home automation robots for intelligent home and used for communication with humans (Yoshimi T. et al., 2004). Besides there are lots of examples of home automation robots; ISSAC (Dong To Nguyen et al., 2005), ETRO (Jeonghye Han et al., 2005), SHR100 (Moonzoo Kim et al., 2005), MARY (Taipalus, T. Et al., 2005), PBMoRo (Ho Seok Ahn et al., 2006), PaPeRo (Sato M. Et al., 2006). The most popular robots in the market are cleaning robots. The examples are home cleaning robots such as Roomba (Jones, J.L., 2006), Roboking (Sewan Kim, 2004), office cleaning robots such as DAVID (Prassler E. et al., 1997), glass-wall

cleaning robots such as Cleaner 3 (Houxiang Zhang et al., 2006). Besides there are various kinds of service robots; welfare robots such as ROMAN (Hanebeck U.D. et al., 1997), guidance robots (Koide Y. et al., 2004), entertainment robots such as AIBO (Fujita M., 2004), therapy robots such as PARO (Shibata T., 2004).

We have designed modular system architecture for intelligent service robot and developed intelligent unmanned store service robot 'Part Timer'. It works in store environment instead of humans and manages the store without any humans' intervention. It recognizes humans and makes conversation with customers who order some products. It can move autonomously, grab some products, and deliver them to customers. It finds suitable information from the internet and gives search results to humans using voice synthesis. Administrator can monitor the status of the store anywhere and anytime by connecting Part Timer or the store server which has most of the information of the store. As Part Timer has lots of abilities like this, it is required that system architecture need to be easy to add or remove functional engine modules. This system architecture is useful for service robots which have complex functions.

2. Part timer

2.1 Overall description

Part Timer is an intelligent service robot specified for unmanned store developed by Perception and Intelligent Lab.(PIL) at Seoul National University and Software Membership at Samsung Electronics Co. It is 100cm tall and weighs 30kgs. The frame of the robot is composed of acryl pannel and aluminum poles for light weight. A lithium-polymer battery is used and it supplies for about three hours without external power source. One embedded computer which uses Intel pentium 4 processor is located inside the robot and Microsoft Embedded Windows XP is used for operating system. Main system accesses the internet using wireless communication and controls most of the appliances in the store using Bluetooth and wireless LAN. It has two wheels for moving and uses ten ultrasonic sensors for measuring distance to objects. It has one camera for image processing and one microphone for communicating with humans by recognizing voice. It has a manipulator for grasping some objects. The manipulator has five degrees of freedon (DOF) and located at the front of the robot. All data is stored to the store server. Table 1 shows the overall specifications of Part Timer.

Height	100cm
Weight	30kg
Frame	Acryl pannel & aluminum poles
Power	Lithium-polymer
Operating System	Microsoft Embedded Windows XP
Communication	Wireless LAN, Bouetooth
Camera	Logitech Quickcam pro 4000 x 2
Moving speed	0 ~ 2m/s
Manipulator	DC motor x 6
Sensor	Ultrasonic x 11 (10 for SLAM, 1 for Manipulator) Gas sensor x 1

Table 1. The overall specifications of Part Timer

It recognizes humans by face recognition engine. It can read characters written in Korean, English and Japanese. It can recognize some objects and grab them. It communicates with humans by recognizing voice using STT(Speech To Text), TTS(Text To Speech) and reacts using conversation engine. When users ask something, it finds the suitable information by surfing the internet autonomously such as weather, news. It also plays music and video through the internet. It has navigation capability and moves to its destination by avoiding obstacles and mapping automatically. It measures the distance between robot and obstacle using ten ultrasonic sensors and calculates the trajectory and robot velocity. Using these capabilities, it receives orders, sells and delivers products. It controls several the electrical appliances, blind curtains, windows etc.

It is controlled remotely as well as directly. Humans manages Part Timer by PDA and mobile phone. Humans can monitor the status of the store and use every functinos which are provided as local service. For example, humans can open the door and close the window in the store remotely. It sends the streaming camera image from the robot to PDA and store server. If an intruder comes in the store and destorys the robot, it is possible to arrest him using store server information, because every status of the store and streaming image are stored in the store server. Fig. 1 shows the functions of Part Timer according to location. Fig 2 shows the environment of the unmanned store service robot Part Timer.

Fig. 3 shows some examples of using Part Timer. In the case of the administration of the store, humans makes schedule of Part Timer remotely as well as directly. It is possible to send some commands to Part Timer and to control it to move anywhere. In the case of guest, when guest meet Part Timer, he just behave same as in a normal case. Part Timer recognizes who the guest is, when the guest visits again, which products guest bought, etc. Part Timer makes conversation with the guest using these data. Part Timer gets the order and sells it. If guest asks something such as the weather of today, the nearest bank form here, Part Timer searches suitable information from the internet and gives the answer to the guest.
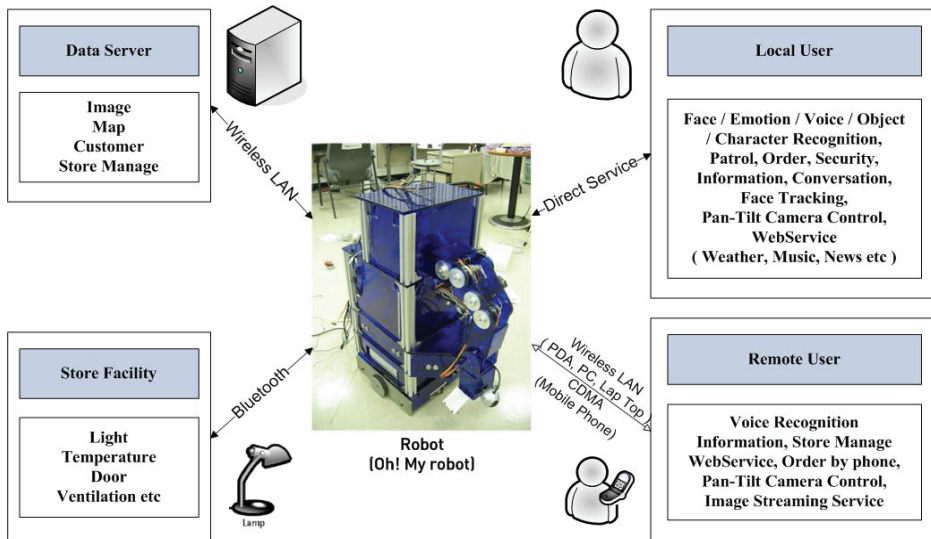

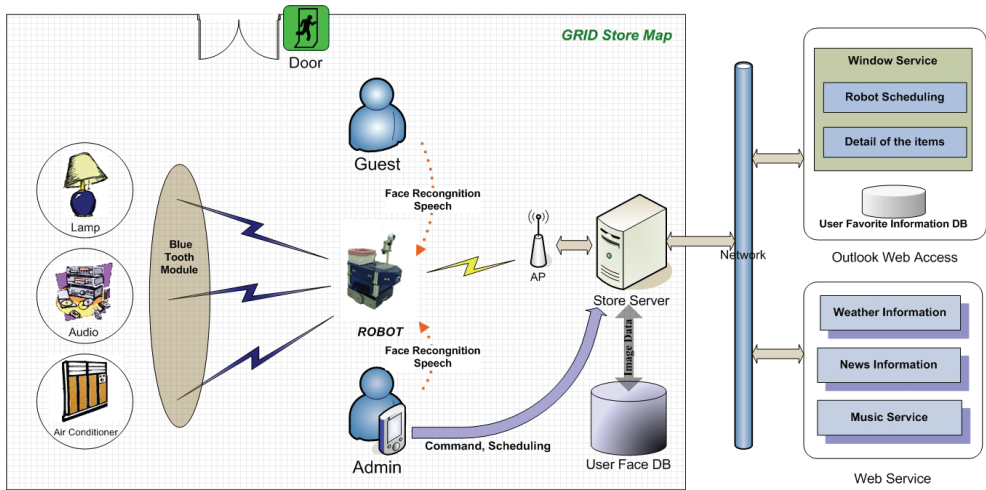Fig. 1. The functions of Part Timer according to location.

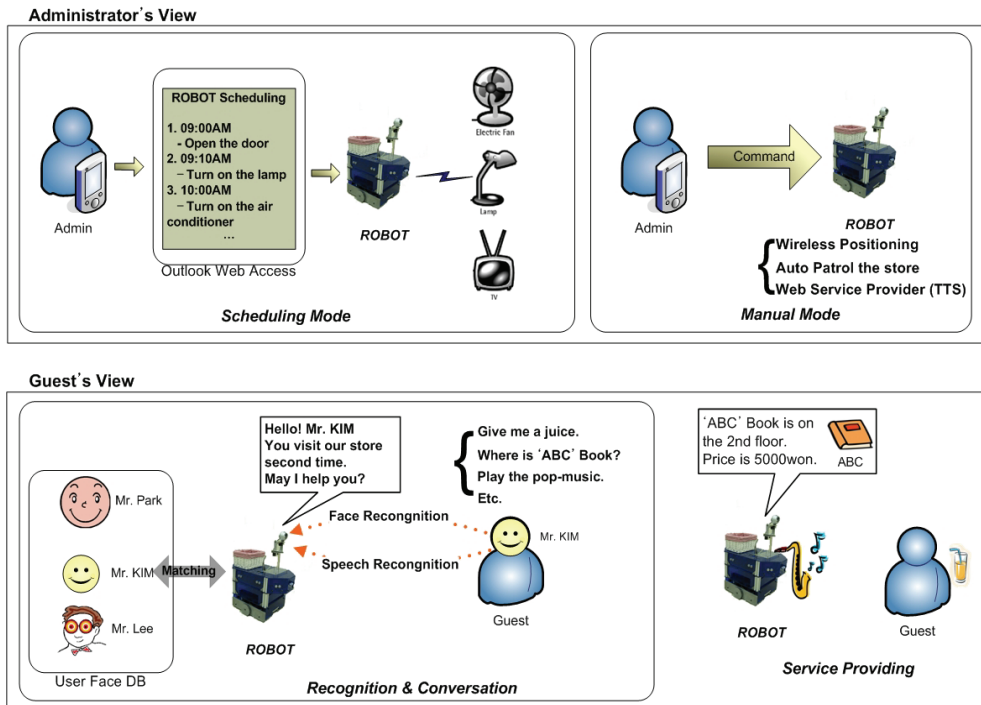Fig. 2. The environment of the unmanned store service robot Part Timer.

Fig. 3. Some examples of using Part Timer.

2.2 System architecture

Notion for software development is divided into structured method and object oriented method (Erich Gamma et al., 1994). Although object oriented method costs a lot for developing, it is more favourable method because of low costs of maintaining, re-using, and revising. Especially in the case of developing intelligent service robots, this method is efficient as lots of researchers work together and it is required to upgrade the robot continuously. Part Timer has lots of functions such as image processing, voice recognition and navigation which are required capabilities for intelligent service robots. Part Timer uses modular system architecture using object oriented method for connecting each functions efficiently. Since it can be different that which intelligent functional engine is needed according to the purpose of the robots. Therefore system architecture for intelligent service robot should allow addition or removal of intelligent functional engine easily.

Fig. 4 shows the system architecture of Part Timer. The system architecture consists of four layers; User Interface layer (UIL), Behavior Scheduling Layer (BSL), Intelligent Module Layer (IML), and Hardware LayerHWL). UIL is the layer for communication with humans. This layer takes part in Humans-Robot Interaction. UIL has some devices for interaction with humans such as touch screen and speaker. UIL offers various kinds of devices for efficiency such as PDA and mobile phone. User can get some information by these devices. These devices can be changed according to the purpose of the robots. Part Timer includes various behaviors such as conversation, delivering and gesture in UIL.

BSL is the core layer that connects each intelligent functional engine organically and decides how to act. Connections are different according to robots and their purpose. Main system in BSL schedules whole works of robot. IML is the layer for each intelligent functional engine. HWL is the layer that contains hardware for each IML. IML and HWL consists system module of each function. As each intelligent algorithm requires some equipments for performing functions, suitable equipments should be added when some algorithms are added. Hence, these two layers organize system modules and operate together.

The system architecture of Part Timer is reconfigurable architecture connecting each intelligent functional engine efficiently. If each intelligent functional engine is regulated to specific rule, it is easy to add or remove it and saves cost for developing and maintaining. As each engine is able to choose suitable algorithms according to situation, it can have better processing results. For these reasons, we have designed Evolvable and Reconfigurable Intelligent (ERI) architecture (Jin Hee Na et al., 2005). We have defined intelligent functional engine as Intelligent Macro Core (IMC) and some IMCs are merged to one module performing one function. ERI architecture has hierarchical structure; each module has upper IMCs and lower IMCs. Upper IMC performs the job using the results from each lower IMC which performs each function. Connection manager chooses and connects suitable module according to situation.

Fig. 6 shows the functional architecture of Part Timer using ERI architecture. Behavior manager is in BSL and performs some jobs; task planning, behavior scheduling, communication, and bridge of modules. Behavior manager has lots of modules which is intelligent functional engine in IML. Part Timer has six modules; SLAM module, vision module, arm module, voice module, web service module, and facility control module. Each module is developed based on ERI architecture. Fig. 7 shows the structure of each module using ERI architecture. Each module has some IMCs which have intelligent functional algorithms.
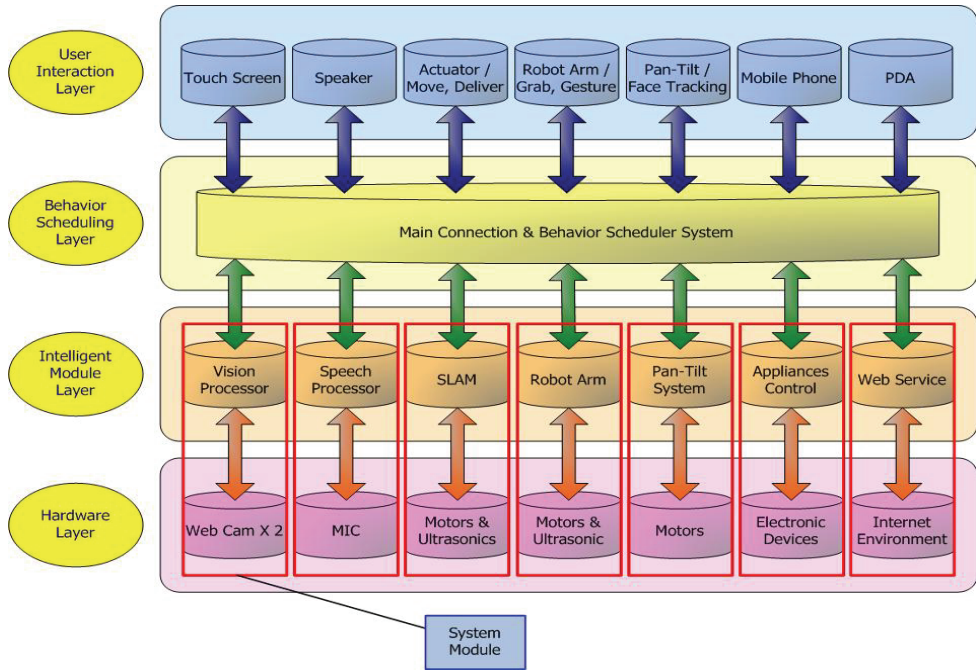
Fig. 4. The system architecture of Part Timer.
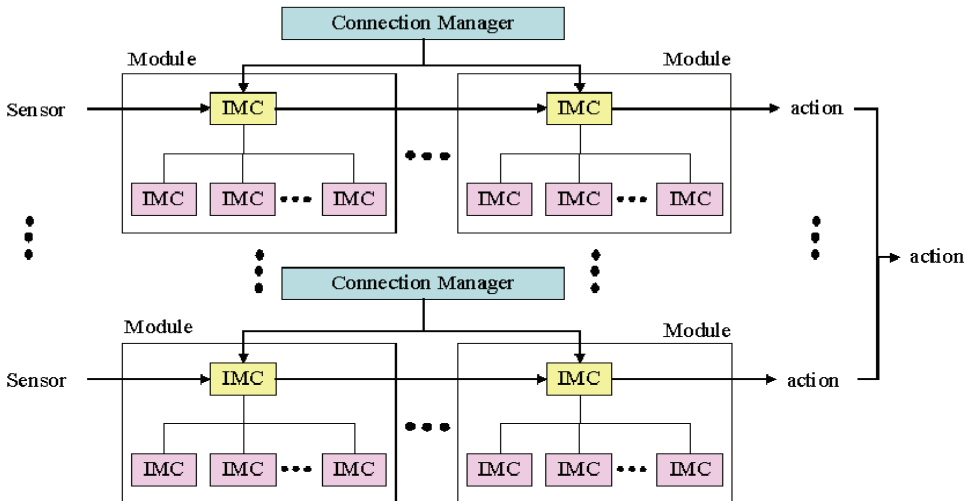
Fig. 5. The system architecture of Part Timer.
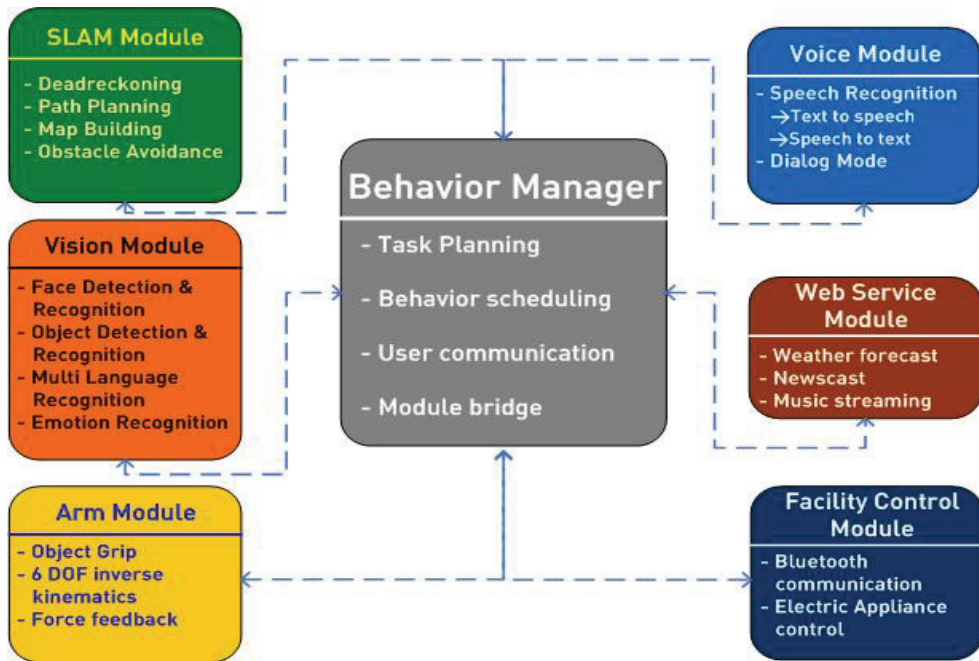
Fig. 6. The functional architecture of Part Timer using ERI architecture.
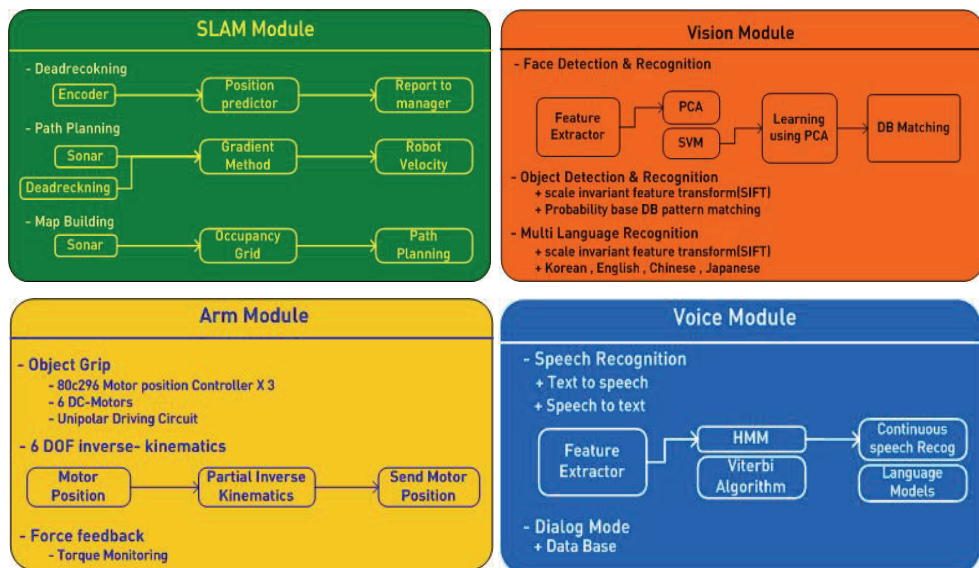
Fig. 7. The structure of each module using ERI architecture.

2.3 Mechanical design
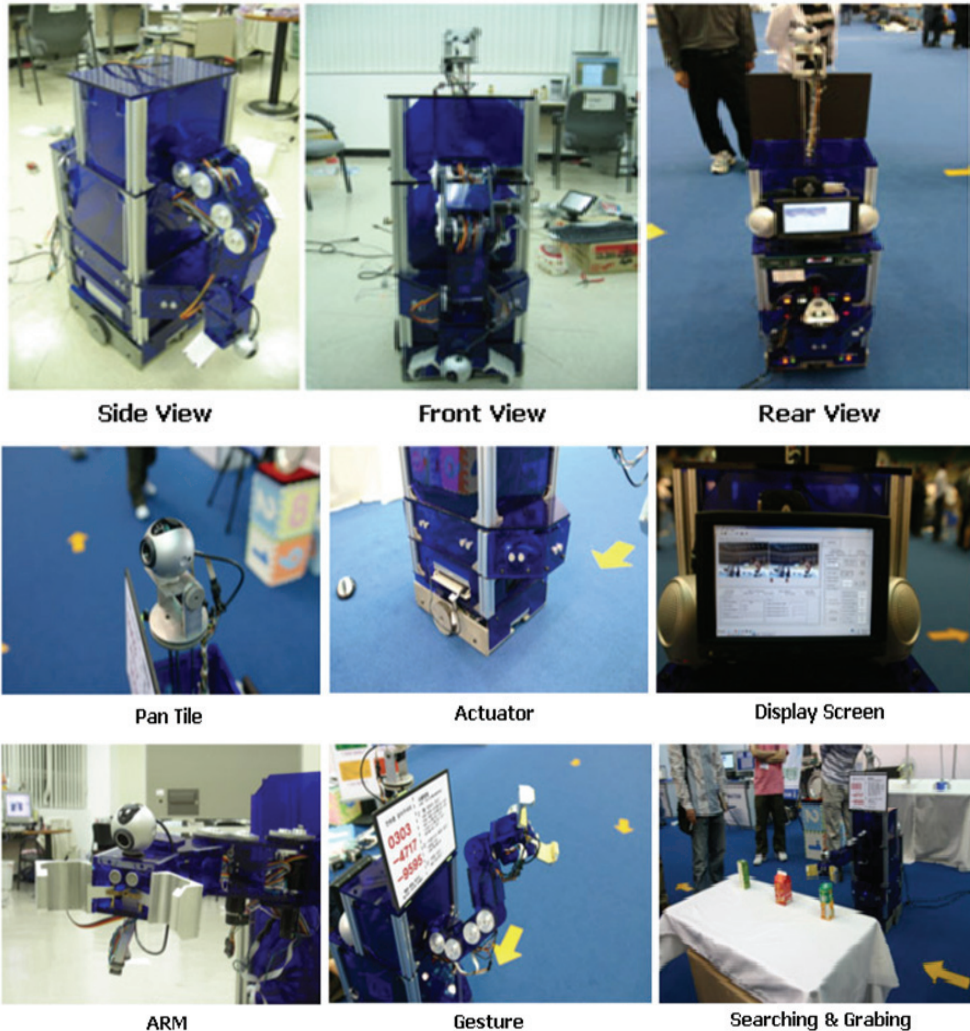
Fig. 8. The mechanical design of Part Timer.

3. Navigation

Navigation of Mobile Robots is a broad topic, covering a large spectrum of different technologies and applications. Briefly, navigation is designated moving technique from current pose to desired pose. It is remarkably easy to handle for humans, but not for robot. It is difficult that a robot recognizes its surroundings to get a current pose and to avoid an obstacles. So this book covers kinematics, motion generator, path planning and map building on mobile robot.

3.1 Kinematics on mobile robot (Seung-Min Baek, 2001).

V_R and V_L denote the velocity of wheels. Hence the linear velocity is directly proportional to the radius of a wheel. We can obtain the lineal velocity (V_C) and the angular velocity (ω_C) of the robot from the velocity of wheels. Fig. 9 shows kinematics on mobile robot.
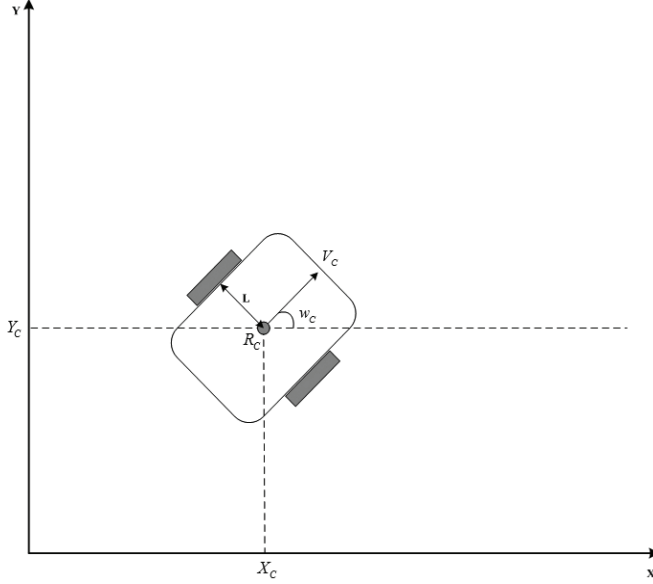
Fig. 9. Kinematics on mobile robot.

The mobile robot must recognize its pose to move freely in the room. It is required to generate proper velocity of robot for reaching the desired pose. It is called 'motion-generator'. We propose a method which generates trapezoidal velocity profile from reactive trace generator. As shown in Fig. 10, the pose of the robot is selected randomly and the desired pose is fixed.

State vector of robot ($S_r(t)$) and Goal Vector ($G(t)$) are described as follows:

$$S_r(t) = [x_r(t) \quad y_r(t) \quad \theta_r(t) \quad V_r(t) \quad \omega_r(t) \quad]^T$$
$$G(t) = [x_g(t) \quad y_g(t)]^T \tag{1}$$

where $x_r(t)$, $y_r(t)$, $\theta_r(t)$ mean robot pose in Eq. (1). $V_r(t)$ denotes linear velocity and $\omega_r(t)$ denotes angular velocity respectively in Eq. (1). Hence motion generator makes next period $V_r(t+1)$ and $\omega_r(t+1)$ by means of current pose in Eq. (2). They can be described as following:

$$V_r(t+1) = \Pr[\; V_r(t) + \delta_t \nabla_v(t+1) \;, where \;|\nabla_v(t+1)| \leq |\nabla_{vMAX}|$$
$$\omega_r(t+1) = \Pr[\; \omega_r(t) + \delta_t \nabla_\omega(t+1) \;, where \;|\nabla_\omega(t+1)| \leq |\nabla_{\omega MAX}| \tag{2}$$

Next step is to decide $\nabla_v(t+1)$ and $\nabla_\omega(t+1)$ reasonably by considering $S_r(t)$ and $G(t)$.
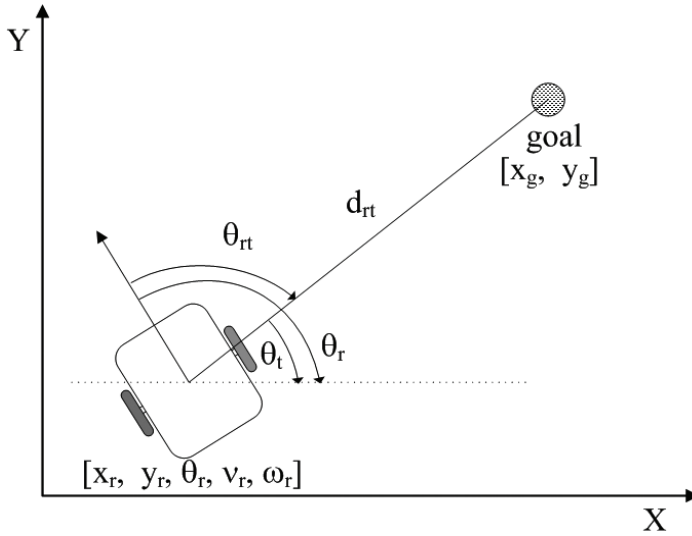
Fig. 10. The motion generator of Part Timer.

3.2 The motion generator

To determine the linear velocity of the robot, we should know the distance from robot to desired pose at first. And the velocity of the robot should be changed by acceleration section and deceleration section. θ_{rt} denotes the difference of angle between θ_r and θ_t in Fig. 10. θ_t denotes the absolute angle of the robot and target respectively. d_{rt}, the distance between the robot and the goal, can be calculated by using max velocity and max acceleration.

$$\theta_{rt} = \theta_r - \theta_t$$

$$d_{dec} = d_{end} - d_1$$

$$= \frac{V^2{}_{MAX}}{2\Delta_{VMAX}} \tag{3}$$

$$V_s = \begin{cases} \sqrt{2 \cdot d_{rt} \cdot \Delta_{vMAX}} \,, & if \; d_{rt} < d_{dec} \\ V_{MAX} \,, & otherwise \end{cases}$$

$$V_s = standard \;\; linear \;\; velocity$$

$$\Delta_v(t+1) = \begin{cases} \Delta_{vMAX} \,, & if \; V_e(t) > \delta_t \cdot \Delta_{vMAX} \\ -\Delta_{vMAX} \,, & if \; V_e(t) < -\delta_t \cdot \Delta_{vMAX} \\ V_e(t) & , otherwise \end{cases} \tag{4}$$

$$where, \quad V_e(t) = V_s(t) - V_r(t)$$

$$V_r(t+1) = \Pr[V_r(t) + \delta_t \Delta_v(t+1)] \,, \; only \; |\Delta_v(t+1)| \leq |\Delta_{vMAX}| \tag{5}$$

$$\Delta_v(t) = linear \;\; acceleration \;\; when \;\; time(t)$$

Where $V_r(t+1)$ represents the next linear velocity in Eq. (5). We can define next angular velocity same as linear velocity.

$$\theta_{dec+} = \frac{\omega^2_{MAX}}{2\Delta_{\omega MAX}}$$

$$\theta_{dec-} = -\frac{\omega^2_{MAX}}{2\Delta_{\omega MAX}}$$

$$\omega_s = \begin{cases} -\omega_{MAX} & , \ if \ \theta_{rt} < \theta_{dec-} \\ -\sqrt{2\cdot|\theta_{rt}|\cdot\Delta_{\omega MAX}} & , \ if \ \theta_{dec-} \le \theta_{rt} < 0 \\ \sqrt{2\cdot|\theta_{rt}|\cdot\Delta_{\omega MAX}} & , \ if \ 0 \le \theta_{rt} < \theta_{dec+} \\ \omega_{MAX} & , \ otherwise \end{cases} \qquad (6)$$

$$\Delta_\omega(t+1) = \begin{cases} +\Delta_{\omega MAX} , & if \ \omega_e(t) > \delta_t \cdot \Delta_{\omega MAX} \\ -\Delta_{\omega MAX} , & if \ \omega_e(t) < -\delta_t \cdot \Delta_{\omega MAX} \\ \omega_e(t) & , \ otherwise \end{cases}$$

$$where, \quad \omega_e(t) = \omega_s(t) - \omega_r(t)$$

$$\omega_r(t+1) = \Pr[\omega_r(t) + \delta_t \Delta_\omega(t+1)] , \ only \ |\Delta_\omega(t+1)| \le |\Delta_{\omega MAX}|$$
$$\Delta_\omega(t) = angular \ acceleration \ when \ time(t) \qquad (7)$$

It is important to decide the motion of the robot using Eq. (5) and Eq. (7). Before applying the motion generator to real robot, we tested the algorithm using simulator as shown in Fig. 11.
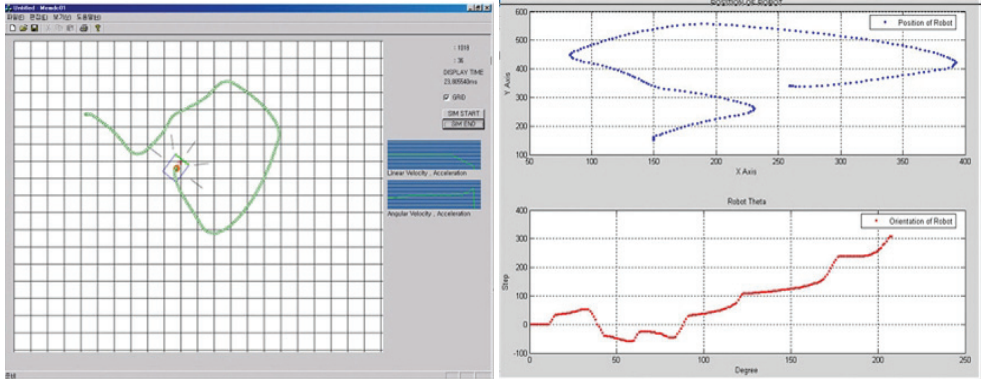
Fig. 11. The experimental results of motion generator simulation.

Right upper graph means the trajectory of the robot and right lower graph means the angular velocity of the robot. Fig. 12 shows simulation of motion generator with obstacle avoidance which is based on rules. You can download the source code of the program from our website (www.andyhouse.net).
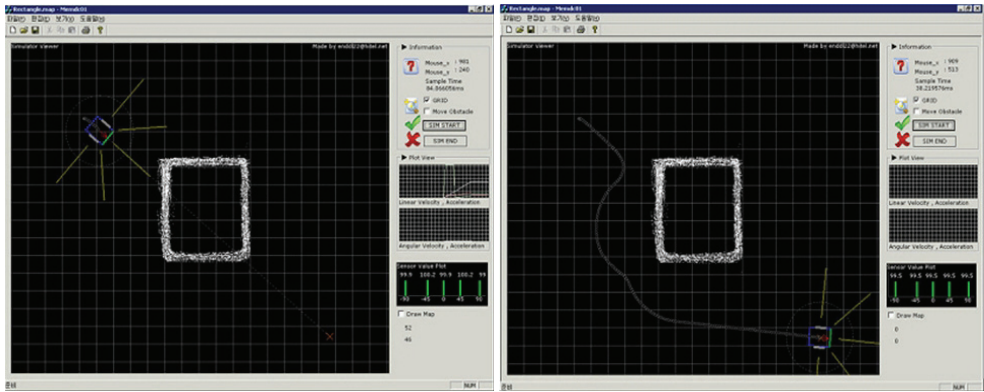
Fig. 12. Obstacle avoidance based on rules.

The greatest merit of obstacle avoidance based on rules is that it can generate avoidance angle without powerful computing ability. Using IF-Else sentence, we can develop the obstacle avoidance function easily. The disadvantage of obstacle avoidance based on rules is that robot may be trapped in loop. Fig. 13 shows the trapped situation. We call this as 'Local minima'. Many methods are used to overcome it. First, the robot can generate a virtual goal when it detects a trapped situation has occurred. Next step is path planning such as the A* and the gradient-method. These algorithms help to find the shortest path from global map which is the result of the map-building.
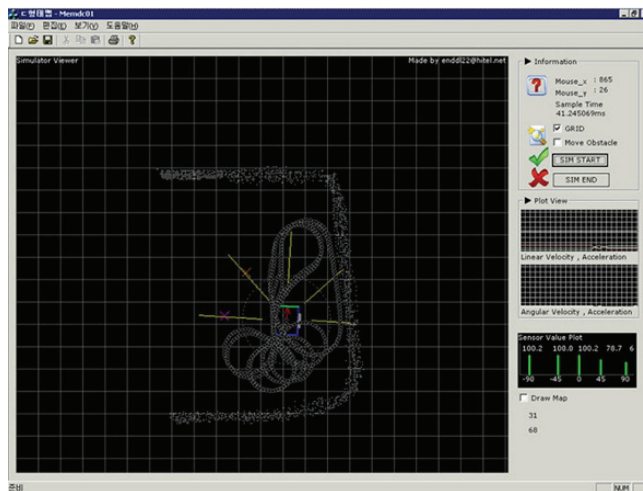
Fig. 13. Trapped situation.

3.2 Path planning and map-building method.

In this book, we use the gradient method which is based on grid-map to find shortest path. The gradient method is a kind of wavefront algorithm. The wavefront algorithm scans grid

map through breadth first algorithm. It can be finished when robots reach the goal. If you want to know further, visit following sites:

Wavefront algorithm:
http://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/16311/www/current/labs/lab05/
Gradient method (Kurt Konolige, 2000):
http://vision.cis.udel.edu/mrp/readings/gradient.pdf

The gradient method is fairly simple and powerful algorithm in path planning. We have developed the simulator to verify the gradient method we implemented. Fig. 14 shows the results of the path planning simulation.
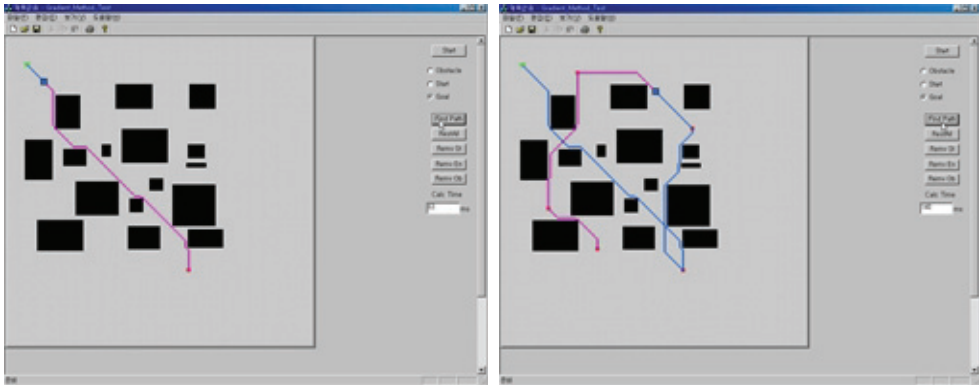
Fig. 14. The results of the path planning simulation

It took about 30ms to find the path in 500 x 500 grid map which is 25m x 25m. We have tested the experiment of path planning using real robot with the same algorithm we have tested in this simulator. It shows a precise motion to avoid several obstacles.
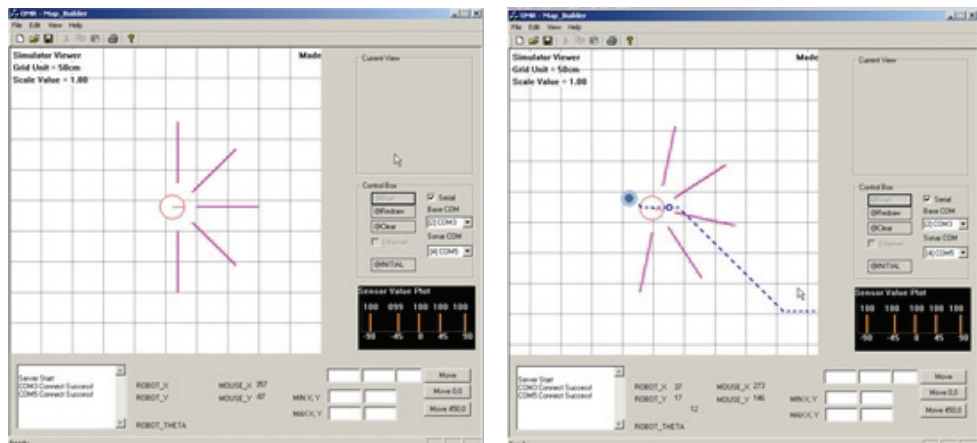
Fig. 15. The experimental results of navigation from start position to goal position.

The reason why we select the gradient method is speed. This algorithm reduces calculation time by stopping if it finds the goal position. Moreover, it can lessen the consumption by linear programming, adds the path which was newly calculated to the existing path. Finally, it can solve local minima problem which most of the path planning algorithms have. However it takes the time proportional to scale of the map. And the map must be stored in physical memory. Recently the power of computer is getting more and more powerful. Hence the path can be calculated in real time even up to 500 x 500 scaled map.
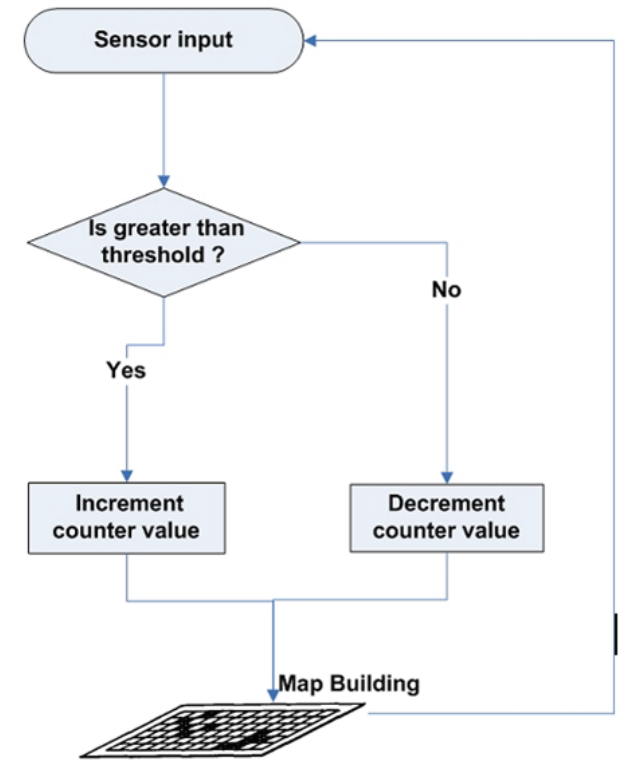
Fig. 16. Simple map-building process.

Part Timer uses ultrasonic sensors to make map. The characteristic of ultrasonic sensor is spreading. Hence it is limited by the shapes of surfaces and the density or consistency of the material. For example, foam on the surface of a fluid in a tank could distort a reading. The main issue is the updating of map. The map could be changed by an accumulated error while robot is moving or stationary. We couldn't compensate the dead-reckoning error because of limited ultrasonic sensors. Fig. 16 shows our simple map-building algorithm.

Fig. 17 shows the simple map-building on the real robot. The map building takes about 100ms. It means that the map building is executed up to 10 times in a second. Consequently we could embed the simple map-building into our real robot. It also describes the map building process while the robot is moving through a confusion area.
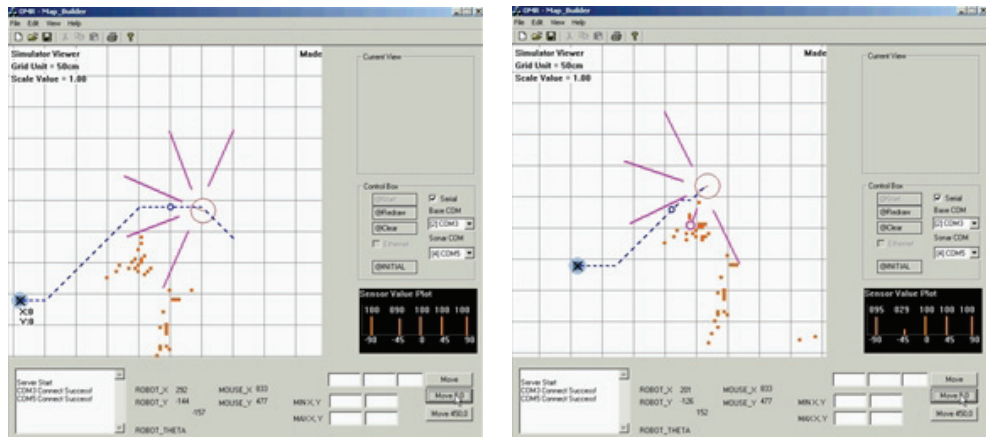
Fig. 17. simple map building experiment.

4. Manipulator

Manipulator is necessary to perform humans-like motion for robots. Especially, it requires grasping action for service robot as well. Recent service robots have manipulators even through they have no legs. It shows the importance of robot manipulators compared to moving actuators. Actually, the design of elaborate manipulator is not easy work. External factors like gravitation, collision with obstacles, and backlash of motors should be considered. Sensing scheme gives the robot surrounding information. In Part Timer system, 5-DoF Manipulator is designated with vision system that uses SIFT Algorithm for object recognition.

4.1 Considerations

There are several issues to implement vision-based grasping manipulation (Smith.C.E. et al., 1996). Two kinds of control systems; Open-Loop and Closed-Loop are mainly introduced. In case of the open-loop control, the vision system is used only to determine the object's position and orientation prior to a blind grasp. On the contrary, closed-loop control allows visual data to compensate for manipulator positioning inaccuracies and sensor noise. Typically, open-loop control requires more accurate calibration while closed-loop control requires faster vision system performance. For our system, we developed a high-performance object recognition algorithm choosing the close-loop control using SIFT (D.G.Lowe, 1999). It gives us more accurate results than open-loop systems because of compensating for noise data in real time manner. Although object's position may be changed during grasping time, closed-loop system would be able to track transition of target object. Because vision system requires high computational power, Pentium IV 2.4GHz is used for object recognition system.

For this, we placed a web-camera on the back of the hand. This placement allows camera to see wider scenes while grasping objects. We use a monocular camera that required no calibration and reduced hardware requirements. Unfortunately, monocular camera systems

cannot provide three-dimensional location of target object. That is why it is needed to recover depth through some different way. We have implemented approximation distance calculator by comparing camera image size to database image size.

 The shape and complexity of a gripper is different from the object that the manipulator would try to grasp. We have designed the gripper that can grasp beverage cans to work in practical store services. The gripper has one degree of freedom so that it can work in two ways; closing, opening. It has one DC motor with a worm gear. Fig. 18 shows mechanical design of Part Timer's gripper.
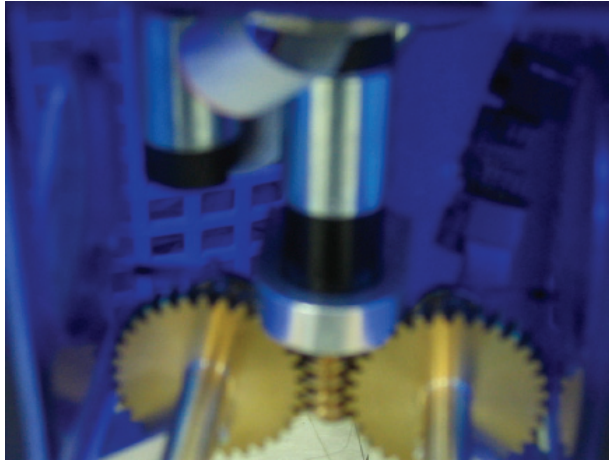
Fig. 18. Mechanical Design of Part Timer's Gripper.

Fig. 19 shows overall system architecture of the manipulator system. There are two main parts; vision system and control system. The vision system takes charge of the object recognition, and the control system is in charge of positioning the robot manipulator to the desired position. Therefore, the vision system includes the object recognition system and depth finder to calculate the 3-dimensional position in real time manner. In the control system, angle value of the each shaft will be calculated by inverse kinematics rule. Then, the manipulator positioning controls the motors to each angle value. These two separated systems are working independently from the main system to improve the performance.

The data flow of the whole manipulator is like below. First, the web-camera captures an image. The image coming from the web-camera goes to the object recognition system. The recognition system calculates a distance from the target object to the gripper. The method to find depth will be discussed in section 4.3. After that, the robot can get approximate coordinates (x, y, z) based on the pre-stored database information. In this time, the ultrasonic sensor is used to compensate errors because depth value may not guarantee accuracy. Next, the control system solves the inverse kinematics problem and moves motors to the desired position. The object recognition system receives the object position while the gripper is moving. Closed-loop system using the feedback of visual data is constituted in Part Timer System.
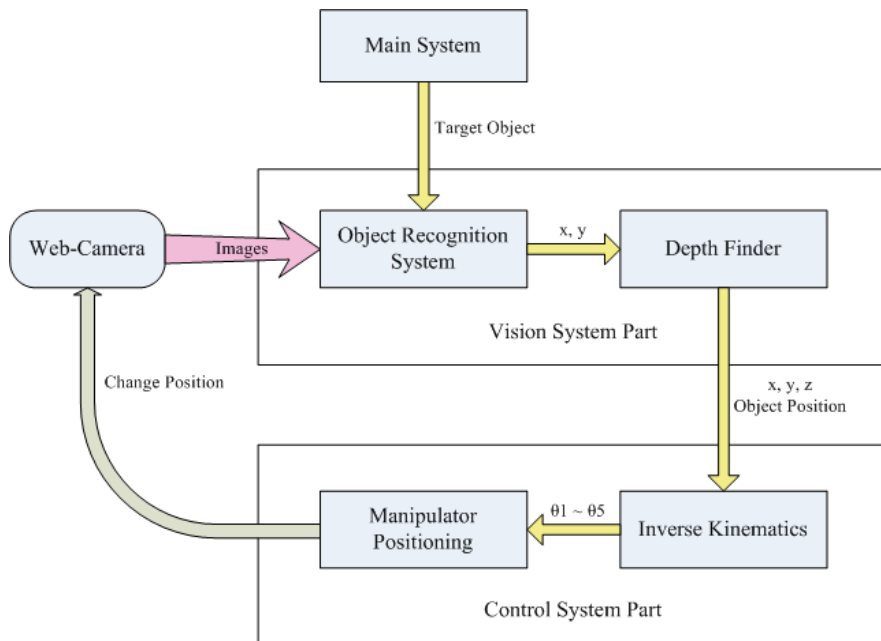
Fig. 19. Overall Manipulator System Architecture.

4.2 Design of robot manipulator

We have designed a 5-DoF robot manipulator to grasp objects. There are two motors on the shoulder (Roll, Pitch), one motor on the elbow (Pitch), and two motors on the wrist (Roll, Pitch). The end effecter also has one motor to drive the worm gear that controls the gripper. There is one manipulator and we put it in front of the robot for the efficiency of the entire robot system. As each shaft has boundary in some area, entire workspace is quite small. To fulfill the delivery service, the robot has moving parts. Moving parts can work concurrently with manipulator through main scheduling system.
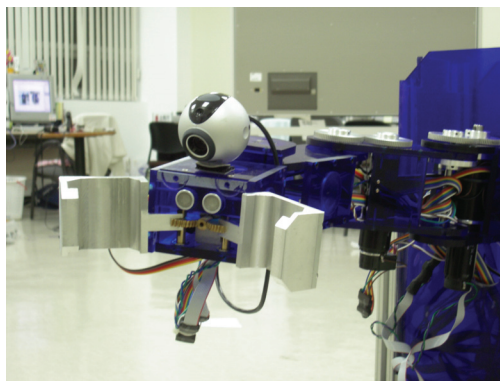
Fig. 20. The gripper with a camera and an ultrasonic sensor.

To grasp objects, the robot should interact with its environment. For example, it should perceive where the desired object is. If there is a camera on the robot on same height as humans eyes are, the robot cannot recognize objects which are far from the robot. General web camera specification is insufficient to see far objects. We put the web-camera on the back of the hand so that it can see the object closer and move its position during searching objects. Even if there are no objects on the camera screen, the robot can try to find the object by locating its end effecter to another position. Placing the camera on the hand is more useful than placing it on the head. One problem occurs if one camera is used. It is that the distance from camera to object is hard to calculate. Therefore, vision system roughly estimates the distance, and compensates the distance by using ultrasonic sensors. Fig. 20 shows the robot arm with a web-camera and an ultrasonic sensor.

4.3 Object recognition system

We use Scale-Invariant Feature Transform to recognize objects (D.G.Lowe, 1999). SIFT uses local features of input image, so it is robustness to scale, rotation and change of illuminations. Closed-loop vision system needs not only robust but also speed. Therefore, we have implemented basic SIFT algorithm and customized it for our robot system for speed up. Fig. 21 shows example of our object recognition system.
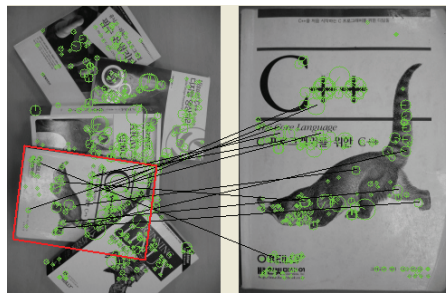
Fig. 21. Example of object recognition.

Unfortunately, the robot has just one camera on the hand, so it is not able to estimate exact distance like when using stereo vision. Therefore, more specific distance for the object database is required to calculate the distance using only one camera. When we make the object database, the robot should know the distance from the object to the camera. Then we calculate the distance comparing the area of object and the size of the database. The size of the object is inversely proportional to the square of the distance. Fig. 22 shows the relationship between the area size and the distance.
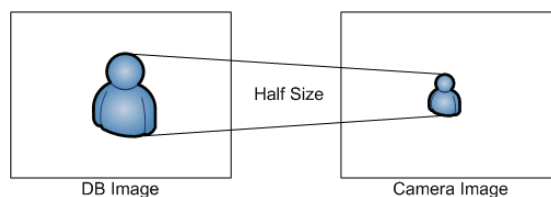
Fig. 22. The relationship between the area and the distance.

We assume that if the object is half the size of the database image, the area of the object will be quarter of the database image. The result of this equation is not correct, but we can adapt the equation to our system to get the distance roughly. Using this law, the robot can calculate the ratio between the area and the distance. The equations are,

$$d_a : d_b \cong s_a : \sqrt{s_b} \qquad (8)$$

$$d_b \cong \frac{d_a \times \sqrt{s_b}}{s_a} \qquad (9)$$

Variable d indicates the distance, and s indicates the size(or area). Small a and b indicate input image and database image respectively. Eq. (8) shows the relationship between distance and area. According to the relation, Eq. (9) shows how we get the approximate distance from the difference of area.

We use SIFT transformation matrix to locate the position of the object in the scene. We can get the transformation matrix if there are three matching points at least (D.G.Lowe, 1999). The transformation matrix indicates the object's location and its orientation. Then the manipulator control system moves motors to locate the end effecter at the center position of the object. However, there are some errors about 3~4cm within work space because of the object shape and database error. Even if very small error occurs, the manipulator has many chances to fail to grasp objects. That is why we use ultrasonic sensors, SRF-04 (Robot Electronics Ltd.,), to compensate errors. The robot computes the interval by measuring the ultrasonic returning time. This sensor fusion scheme reduces most failure ratio.

4.4 Flow chart

Even if manipulator system is accurate and robust, it may not be possible to grasp the object, only using the manipulator system. It requires that the integration of entire robot system. We present the overall robot system and flowchart for grasping objects. Grasping strategy plays an important role in system integration. We assumed some cases and started to integrate systems based on a scenario. In practical environment, there are many exceptional cases that we could not imagine. We though that the main purpose of the system integration is to solve the problems that we faced. Fig. 23 shows the flowchart of grasp processing.

First, the robot goes to the pre-defined position where the desired object is near by. Here, we assumed that the robot knows approximately the place where the object is located. After moving, the robot searches the object by using its manipulator. If the robot finds the desired object, it moves to the location of the object in the workspace. That is why the scanning process is necessary as the web camera is able to search further range than the manipulation workspace. The moving part of the robot is using different computing resources, so we can process the main scheduler and the object recognition in parallel. Fig. 24 shows the movement of the robot when the object is outside of the workspace.

After that, the robot moves the manipulator, so that the object is at the center of the camera by solving inverse kinematics problem. In this time, the image data will be captured and continually used for vision processing. If the object is in the workspace, the robot holds out its manipulator while the ultrasonic sensor is checking whether the robot can grasp the object or not. If the robot decides that it is enough distance to grasp the object, the gripper would be closed to grasp. Using the processing described above, the robot can grasp the object.
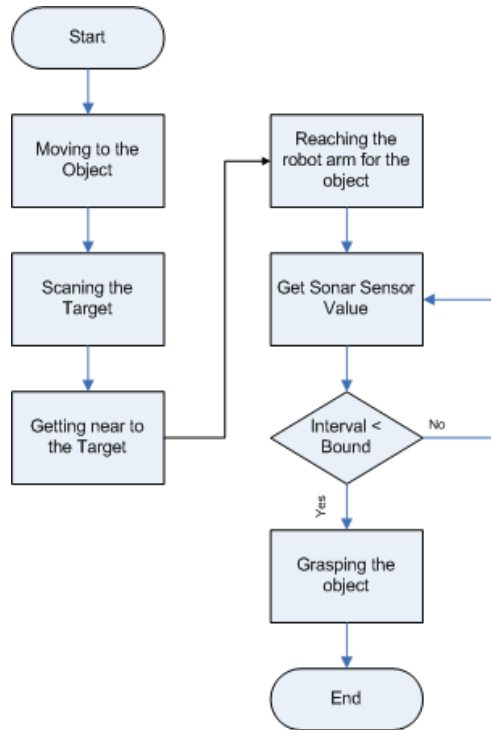
Fig. 23. The flowchart of grasping processing.

(a) Scanning Position   (b) Grasping Position
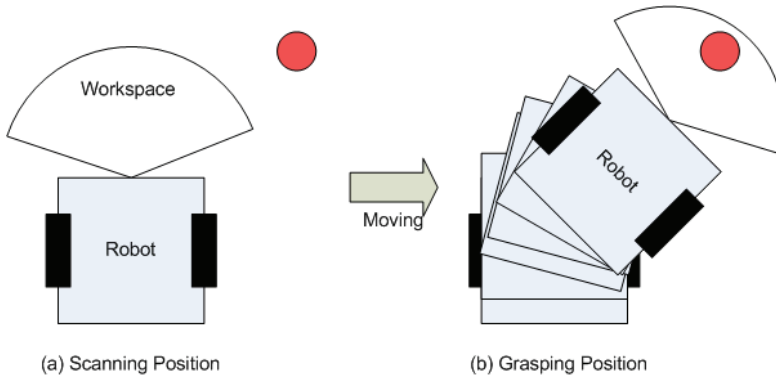
Fig. 24. Movement of the robot after scanning objects.

Fig. 25 presents the processing after the robot has found the desired object. First, the robot arm is in an initial state. If the robot receives a scanning command from the main scheduler, the object recognition system starts to work and the robot locates its manipulator to other position. If the robot finds the object, the manipulator will reach out. The ultrasonic sensor is used in this state. Reaching the robot manipulator, the ultrasonic sensor is checking the distance from the object. Finally, the gripper is closed.
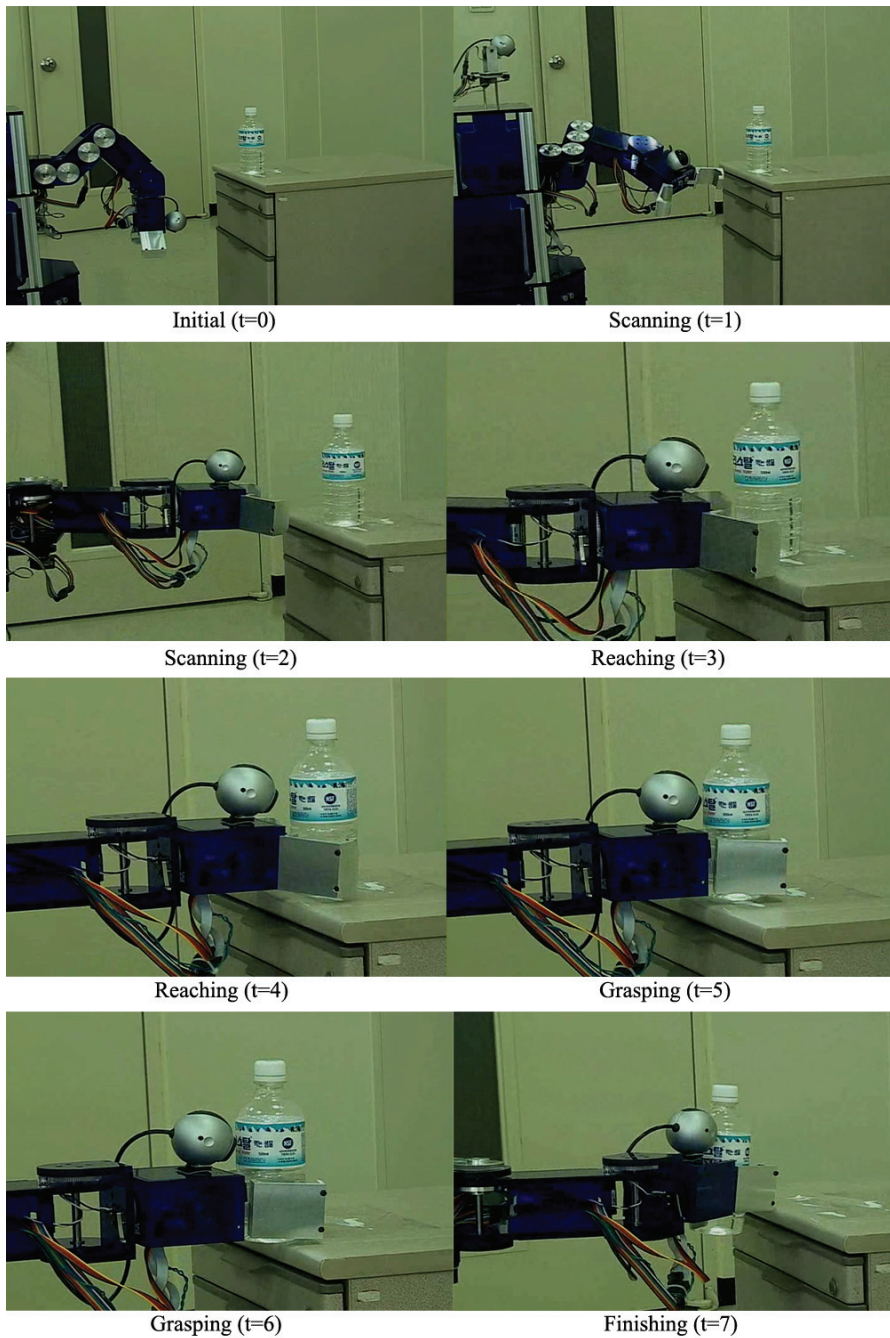
Fig. 25. Time progressing of the grasping action.

5. Face feature processing

The face feature engine consists of three parts; the face detection module, the face tracking module, and the face recognition module. In the face detection module, the final result is the nearest face image from the continuous camera images using CBCH algorithm. In the face tracking module, Part Timer tracks the detected face image using pan-tilt control system and fuzzy controller to make the movement smooth. In the face recognition module, it recognizes who the person is using CA-PCA. Fig. 26 shows the block diagram of face feature engine. The system captures the image from the camera and sends it to the face detection module to detect the nearest face image among the detected face images. And the face image is sent to the face tracking module and the face recognition module.
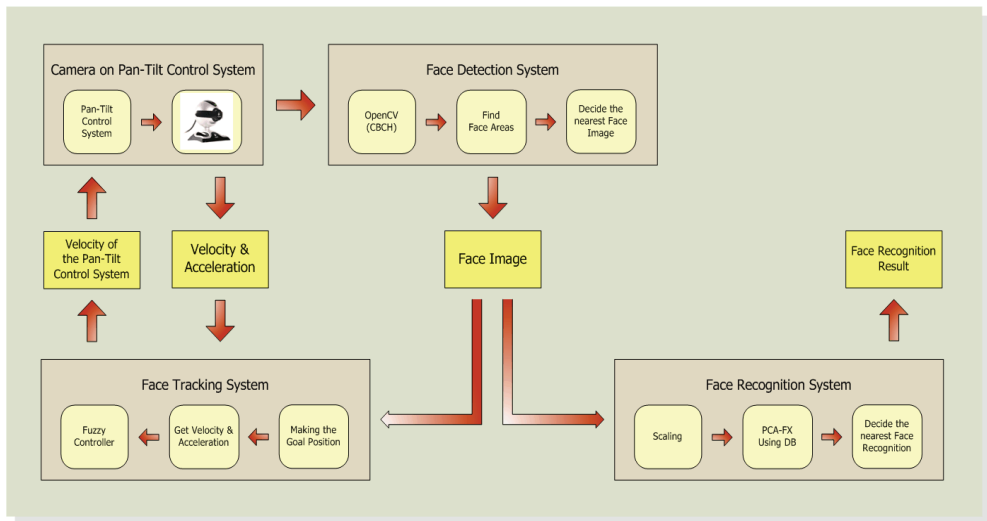
Fig. 26. The block diagram of face feature engine.

The face detection module uses facial feature invariant approach. This algorithm aims to find structural features that exist even when the pose, viewpoint, or lighting conditions vary, and then use these to locate faces. This method is designed mainly for face localization. It uses OpenCV library (Open Computer Vision Library) that is image processing library made by Intel Corporation. It not only has lots of image processing algorithms but also is optimized for Intel CPU so it shows fast execution speed. And they opened the sources about algorithms of OpenCV so we can amend the algorithms in our own way. To detect face, we use the face detection algorithm using CBCH (cascade of boosted classifiers working with Haar-like features) (Jos´e Barreto et al., 2004). The characteristic of the CBCH is fast detection speed, high precision and simple calculation of assorter. We use the adaboost algorithm to find fit compounding of Haar assorter and it extracts the fittest Haar assorter to detect face among all of the possible Haar assorter in order. Of course it shows the calculated result, the weight for each Haar assorter, because each one has different performance. And the extracted Haar assorters discriminate whether it is face area or not and distinguish whether it is face image or not by a majority decision. Fig. 27 shows the results of the face detection module.

Fig. 27. The results of the face detection module. In case there is just one person (left), there are three persons (right).

The face tracking module uses the fuzzy controller to make the movement of pan-tilt control system stable. Generally, the fuzzy controller is used to compensate real time operation of the output about its input. And it is also used by the system which is impossible to model mathematically. We use the velocity and the acceleration of the pan-tilt system for the input of the fuzzy controller and get the velocity of the pan-tilt system for the output. Table 2 shows the inputs and the output. We design the fuzzy rule like Fig. 28 and Fig. 29 is its graph. Fig. 28 means that if the face is far from center of camera image, move fast and if the face is near to center of camera image, move little.

	Pan (horizontality)	Tilt (verticality)
Input 1	Velocity (-90 ~ 90)	Velocity (-90 ~ 90)
Input 2	Acceleration (-180 ~ 180)	Acceleration (-180 ~ 180)
Output	Velocity of Pan (-50 ~ 50)	Velocity of Tilt (-50 ~ 50)

Table 2. The Input and output of the pan-tilt control system.

Fig. 28. The fuzzy controller of pan-tilt system.
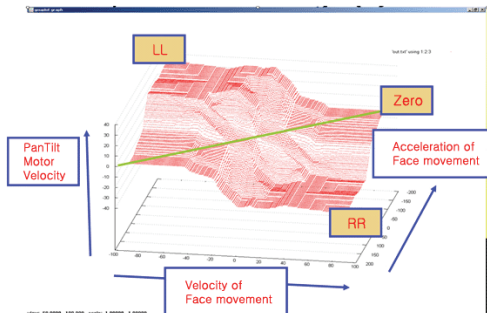
Fig. 29. The graph of Fig. 28.

The face recognition engine uses CA-PCA algorithm using both the input and the class information to extract features which results in better performance than conventional PCA (Myuong Soo Park et al., 2006). We built the facial database to train recognition module. Our facial database consists of 300 gray scale images of 10 individuals with 30 different images. Fig. 30 shows the results of face classification in Part Timer.
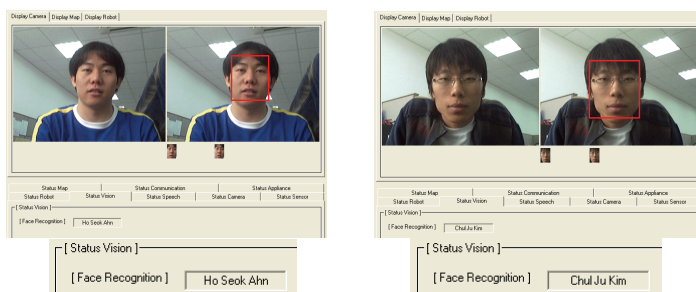
Fig. 30. The results of face classification in Part Timer.

6. Conclusion

Part Timer is unmanned store service robot and it has lots of intelligent functions; navigation, grabbing objects, gesture, communication with humans, recognizing face, object and character, surfing the internet, receiving calls, etc. It has a modular system architecture which uses intelligent macro core module for easy composition of whole robot. It offers remote management system for humans who are outside of the store. We have participated in many intelligent robot competitions and exhibitions to verify the performance. We won the first prize in many of these competitions; Korea Intelligent Robot Competition, Intelligent Robot Competition of Korea, Robot Grand Challenge, Intelligent Creative Robot Competition, Samsung Electronics Software Membership Exhibition, Intelligent Electronics Competition, Altera NIOS Embedded System Design Contest, IEEE RO-MAN Robot Design Competition, etc. Although Part Timer is unmanned store service robot, it can be used for office or home robots as well. As essential functions for service robot similar, it could be used for other purpose if the system architecture we introduced is used. For future work, we are applying the system architecture to multi robot system in which it is possible to cooperate with other robots.

8. References

Sakai K., Yasukawa Y., Murase Y., Kanda S. & Sawasaki N. (2005). Developing a service robot with communication abilities, *In Proceedings of the 2005 IEEE International Workshop on Robot and Humans Interactive Communication (ROMAN 2005)*, pp. 91-96.

Riezenman, M.J. (2002). Robots stand on own two feet, *Spectrum, IEEE*, Vol. 39, Issue 8, pp. 24-25.

Waldherr S., Thrun S. & Romero R. (1998). A neural-network based approach for recognition of pose and motion gestures on a mobile robot, *In Proceedings of Brazilian Symposium on Neural Networks*, pp. 79-84.

Mumolo E., Nolich M. & Vercelli G. (2001). Pro-active service robots in a health care framework: vocal interaction using natural language and prosody, *In Proceedings of*

the 2001 IEEE International Workshop on Robot and Humans Interactive Communication (ROMAN 2001), pp. 606-611.

Kleinehagenbrock M., Fritsch J. & Sagerer G. (2004). Supporting advanced interaction capabilities on a mobile robot with a flexible control system, In Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004), Vol. 4, pp. 3649-3655.

Hyung-Min Koo & In-Young Ko (2005). A Repository Framework for Self-Growing Robot Software, 12th Asia-Pacific Software Engineering Conference (APSEC '05).

T. Kanda, H. Ishiguro, M. Imai, T. Ono & K. Mase (2002). A constructive approach for developing interactive humansoid robots, In Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2002), pp. 1265–1270.

Nakano, M. et. al. (2005). A two-layer model for behavior and dialogue planning in conversational service robots, In Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), pp. 3329-3335.

Gluer D. & Schmidt G. (2000). A new approach for context based exception handling in autonomous mobile service robots, In Proceedings of the 2000 IEEE/RSJ International Conference on Robotics & Automation (ICRA 2000), Vol. 4, pp. 3272-3277.

Yoshimi T. et. al. (2004). Development of a concept model of a robotic information home appliance, ApriAlpha, In Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004), Vol. 1, pp. 205-211.

Dong To Nguyen, Sang-Rok Oh & Bum-Jae You (2005). A framework for Internet-based interaction of humanss, robots, and responsive environments using agent technology, IEEE Transactions on Industrial Electronics, Vol. 52, Issue 6, pp. 1521-1529.

Jeonghye Han, Jaeyeon Lee & Youngjo Cho (2005). Evolutionary role model and basic emotions of service robots originated from computers, In Proceedings of the 2005 IEEE International Workshop on Robot and Humans Interactive Communication (ROMAN 2005), pp. 30-35.

Moonzoo Kim, Kyo Chul Kang & Hyoungki Lee (2005). Formal Verification of Robot Movements - a Case Study on Home Service Robot SHR100, In Proceedings of the 2005 IEEE/RSJ International Conference on Robotics & Automation (ICRA 2005), pp. 4739-4744.

Taipalus, T. & Kazuhiro Kosuge (2005). Development of service robot for fetching objects in home environment, In Proceedings of the 2005 IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA 2005), pp. 451-456.

Ho Seok Ahn, In-kyu Sa & Jin Young Choi (2006). 3D Remote Home Viewer for Home Automation Using Intelligent Mobile Robot, In Proceedings of the SICE-ICASE International Joint Conference 2006 (ICCAS2006), pp. 3011-3016.

Sato M., Sugiyama A. & Ohnaka S. (2006). Auditory System in a Personal Robot, PaPeRo, In Proceedings of the International Conference on Consumer Electronics, pp. 19-20.

Jones, J.L. (2006). Robots at the tipping point- the road to iRobot Roomba, IEEE Robotics & Automation Magazine, Vol. 13, Issue 1, pp. 76-78.

Sewan Kim (2004). Autonomous cleaning robot: Roboking system integration and overview, In Proceedings of the 2004 IEEE/RSJ International Conference on Robotics & Automation (ICRA 2004), Vol. 5, pp. 4437-4441.

Prassler E., Stroulia E. & Strobel M. (1997). Office waste cleanup: an application for service robots, *In Proceedings of the 1997 IEEE/RSJ International Conference on Robotics & Automation (ICRA 1997)*, Vol. 3, pp. 1863-1868.

Houxiang Zhang, Jianwei Zhang, Guanghua Zong, Wei Wang & Rong Liu (2006). Sky Cleaner 3: a real pneumatic climbing robot for glass-wall cleaning, *IEEE Robotics & Automation Magazine*, Vol. 13, Issue 1, pp. 32-41.

Hanebeck U.D., Fischer C. & Schmidt G. (1997). ROMAN: a mobile robotic assistant for indoor service applications, *In Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 1997)*, Vol. 2, pp. 518-525.

Koide Y., Kanda T., Sumi Y., Kogure K. & Ishiguro H. (2004). An approach to integrating an interactive guide robot with ubiquitous sensors, *In Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004)*, Vol. 3, pp. 2500-2505.

Fujita M. (2004). On activating humans communications with pet-type robot AIBO, *Proceedings of the IEEE*, Vol. 92, Issue 11, pp. 1804-1813.

Shibata T. (2004). An overview of humans interactive robots for psychological enrichment, *Proceedings of the IEEE*, Vol. 92, Issue 11, pp. 1749-1758.

Erich Gamma, Richard Helm, Ralph Johnson & John Vlissides (1994). *Design Patterns*, Addison Wesley

Jin Hee Na, Ho Seok Ahn, Myoung Soo Park & Jin Young Choi (2005). Development of Reconfigurable and Evolvable Architecture for Intelligence Implement. *Journal of Fuzzy Logic and Intelligent Systems*, Vol. 15, No. 6, pp. 35-39.

Konolige, K. (2000). A gradient method for realtime robot control, *In Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)*, Vol. 1, pp. 639-646

Rosenblatt J (1995). DAWN: A distributed Architecture for Mobile Navigation, *Spring Symposium on Lessons Learned for Implemented Software Architecture for Physical Agent*, pp. 167-178.

Roland Siegwart (2007). Simultaneous localization and odometry self calibration for mobile robot, *Autonomous Robots*. Vol. 22, pp. 75–85.

Seung-Min Baek (2001). Intelligent Hybrid Control of Mobile Robotics System. *The Graduate School of Sung Kyun Kwan University*.

Smith.C.E. & Papanikolopoulos.N.P(1996). Vision-Guided Robotic Grasping: Issues and Experiments. *In Proceedings of the 1996 IEEE/RSJ International Conference on Robotics & Automation (ICRA 2004)*, Vol. 4, pp. 3203-3208.

D.G.Lowe(1999). Object recognition from local scale-invariant features. *In Proceedings of the 1999 International Conference on Computer Vision (ICCV 1999)*, pp. 1150-1157.

D.G.Lowe(2004). Distinctive image features from scale invariant keypoints. *In Proceedings of the 2004 International Journal of Computer Vision (IJCV 2004)*, pp. 91-110.

Jos´e Barreto, Paulo Menezes & Jorge Dias (2004). Humans-Robot Interaction based on Haar-like Features and Eigenfaces. *In Proceedings of the 2004 IEEE/RSJ International Conference on Robotics & Automation (ICRA 2004)*, Vol. 2, pp. 1888-1893.

Park, M.S., Na, J.H. & Choi, J.Y. (2006). Feature extraction using class-augmented principal component analysis, *In Proceedings of the International Conference on Artificial Neural Networks)*, Vol. 4131, pp. 606-615.

The Development of an Autonomous Library Assistant Service Robot

Julie Behan
Digital Health Group, Intel,
University of Limerick, Limerick,
Ireland

1. Introduction

In modern society, service robots are becoming increasingly integrated into the lives of ordinary people. This is primarily due to the fact that the world is becoming an aged society (a society in which 10% of the population is over 60 years of age). Service robots may provide support to this increasing pool of aged individuals in a variety of forms, such as social interaction robots (Bruce et al., 2001; Breazeal 2002; Fong et al., 2003), task manipulation in rehabilitation robotics (Casals et al., 1993; Bolmsjo et al., 1995; Dario et al., 1995) and through assistive functionality such as nurse bots, tour guide robots etc (Evans 1994; Thrun et al., 2000; Graf et al., 2004). This chapter describes the development of an autonomous service robotic assistant known as "LUCAS": Limerick University Computerized Assistive System, whose functionality includes the assistance of individuals within a library environment. The robot is described in this role through environment interaction, user interaction and integrated functionality. The robot acts as a guide for users within the library to locate user specific textbooks. A complete autonomous system has been implemented, which allows for simple user interaction to initiate functionality and is described specifically in terms of its implemented localization system and its human–robot interaction system.

When evaluating the overall success of a service robot, three important factors need to be considered: 1. A successful service robot must have complete autonomous capabilities. 2. It must initiate meaningful social interaction with the user and 3. It must be successful in its task. To address these issues, factors 1 and 3 are grouped together and described with respect to the localization algorithm implemented for the application. The goal of the proposed localization system is to implement a low cost accurate navigation system to be applied to a real world environment. Due to cost constraints, the sensors used were limited to odometry, sonar and monocular vision. The implementation of the three sensor models ensures that a two dimensional constraint is provided for the position of the robot as well as the orientation. The localization system described here implements a fused mixture of existing localization techniques, incorporating landmark based recognition, applied to a unique setting. In classical approaches to landmark based pose determination, two distinguished interrelated problems are identified. The first is the correspondence problem, which is concerned with finding pairs of corresponding landmark and image features. The

second stage is the pose problem, which consists of finding the 3D camera coordinates with respect to the origin of the world model given the pair of corresponding features. Within the described approach, rather than extracting features and creating a comparison to environment models as described in the correspondence problem, the features within the image are fitted to known environment landmarks at that estimated position. This can be achieved due to the regularity of common landmarks (bookshelves) within the environment. The landmark features of interest, which consist of bookshelves, are recognized primarily through the extraction of vertical and horizontal line segments which may be reliably manipulated even under changing illumination conditions. The method proposes a simple direct fitting of line segments from the image to the expected features from the environment using a number of techniques. Once the features in the image are fitted to the environment model the robot's pose may then be calculated. This is achieved through the fusion of validated sonar data, through an application of the Extended Kalman Filter, and the matched environment features. The results section for the given localization technique shows the level of accuracy and robustness achievable using fusion of simplistic sensors and limited image manipulation techniques within a real life dynamic environment.

The second factor, which is used to evaluate the success of a service robot is its ability to interact meaningfully with its users. Through the development and implementation of existing service robotic systems within real time public dynamic environments, researchers have realised that several design principles need to be implemented for a robotic application to be a success. From existing literature, within the field of service and assistive robotics, these principles have been categorised into eight major topics. To incorporate the essential design principles one of the most important aspects in the creation of a successful assistant robot is the human-robot interaction system. To achieve successful human-robot interaction, between "LUCAS" and the user, a graphical interface displaying a human-like animated character was implemented. The software used in the creation of the character is known as the Rapid Application Developer (RAD), and is part of the CSLU toolkit created by the Oregon Health & Science University (OHSU) Centre for Spoken Language (RAD 2005). The implementation of the RAD, as well as the supported functionality of the robot, allows for the critical design principles to be implemented, helping to ensure that the robotic application may be successfully applied to a real life environment.

The remainder of this chapter is divided into the following topics: Section 2 discusses the introduction of service robots and their applications. Section 3 describes the implemented localization system. Section 4 details the main design requirements for a successful service robot, in terms of human-robot interaction and usability, and how they are applied to "LUCAS". Finally Section 5 discusses the overall implementation and successful application of the robot.

2. Why service robots?

According to the United Nations Population Division the number of elderly people is increasing dramatically in modern day society (U.N. 2004), for example, in Ireland it is predicted that by the year 2050, the number of people aged 65+ will be 24% of the total population (today 11.4%) and the number of people aged 80+ will be 6.4% (today 1.3%). This trend is being experienced world wide and is known as "greying population" (U.N. 2004). If we follow disability trends associated with ageing, we can predict that in this new modern society 16% or more of these persons over the age of 65 will be living with one or more impairments that disrupt their ability to complete activities of daily living in their homes

(Johnson et al., 2003). In Ireland by 2050 this will amount to approximately 192 thousand people. This increasing trend has encouraged the development of service robots in a variety of shapes, forms and functional abilities to maintain the well-being of the population, both through social interaction and as technical aids to assist with every day tasks.

The types of robots used in assistance vary greatly in their shape and form. These robots range from desktop manipulators to autonomous walking agents. Examples include: workstation-based robots such as HANDY1 (Topping & Smith 1998), stand alone manipulator systems such as ISAC (Kawamura et al., 1994), and wheelchair based systems such as MANUS (Dallaway et al., 1992). The mobile robot field is the area that has expanded the most in service and assistive robotics. Examples of which include HELPMATE, an autonomous robot that carries out delivery missions between hospital departments and nursing stations (Evans 1994), RHINO and MINERVA guide people through museum exhibits (Thrun et al., 2000), CERO is a delivery robot that was developed as a fetch-and-carry robot for motion-impaired users within an office environment (Severinson-Eklundh, et al., 2003). PEARL (Pineau et al., 2003) is a robot that is situated in a home for the elderly and its functions include guiding users through their environment and reminding users about routine activities such as taking medicine etc. New commercial applications are emerging where the ability to interact with people in a socially compelling and enjoyable manner is an important part of the robot's functionality (Breazeal 2001). A social robot has been described as a robot who is able to communicate and interact with its users, understand and even relate to its users in a personal way (Breazeal 2002).

The goal of this project was to develop an autonomous robotic aid to assist and benefit the increasing pool of potential users within a library environment. The users may include elderly individuals who find the library cataloguing system confusing, individuals with various impairments such as impaired vision, degenerative gait impairments (the robot leads the user directly to the location of the specific textbook and avoids the user traversing the aisles of the library to locate the textbook), cognitively impaired individuals or users who are simply unfamiliar with the existing library structure. The robot used in the application was specifically designed and built in-house and may be seen in Fig. 1.
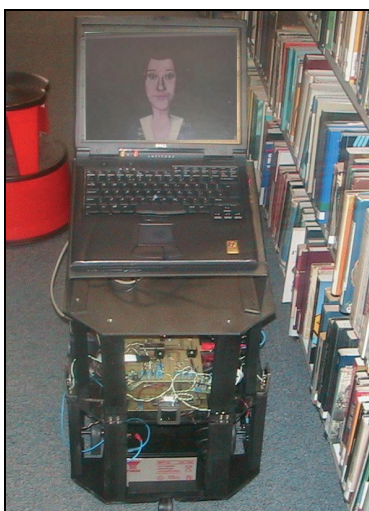
Fig. 1. "LUCAS": Limerick University Computerized Assistive System

3. Localization

In literature there are three main methods to solve the problem of localization. The first is map-based approaches where the robot makes use of a user created geometric or topological model of the environment, which is usually referred to as an "occupancy map". The second approach is map-building based, where the robot uses its sensors to create geometric or topological models of the environment and then use these models for navigation. The third approach is map-less navigation where the robot has no explicit representation of the environment but relies on object recognition or object tracking to aid navigation. For indoor environments the map-based approach is the most common technique used due to the structured nature of the environment. In (Pérez et al., 1999), Pérez et al. describe the basic idea of localization within a mapped environment through the following method: At each location of the robot trajectory the localization system perceives the environment with some sensing device. This sensorial information is compared with the expected values (predicted from an *a priori* map), and used to correct the available robot location, to make the perceived data match better with the expected data. The localization system described here is based on Pérez's essential elements to create an accurate and robust localization process using minimalist sensors and processing time, while maintaining reliable accuracy. For a real world application setting, a cost effective solution is essential for universal acceptance and deployment. "LUCAS" was designed to be low cost with its dependence being on basic sensor fusion including calibrated odometry, sonar and monocular vision. As literature has shown, a more precise estimation of the localization parameters may be provided through the fusion of multiple data sources. The authors believe that instead of relying on complex geometric measurements, even with more basic information, a proper combination of different sources can achieve accurate localization. The localization system makes use of the environment features (landmarks), which primarily consist of rows of bookshelves. Horizontal or vertical edges occur frequently in man made environments, which are generally parallelepiped in nature. These edges provide essential information regarding the environment and are generally noted for their simplicity in extraction. The localization system provides an accurate and complete solution, for the specific environment, using low cost off-the-shelf sensing devices. The method proposed is a continuous localization process rather than a single localization step so odometry errors do not have time to accumulate. This allows the robot to apply individual localization procedures for specific localization regions based on odometry alone. The localization system is implemented in a three stage approach. The first stage occurs when the robot is navigating in the *y world* direction, while the robot is locating the correct bookshelf row. The procedure implemented involves initial position estimation from odometry, fused with vertical feature extraction from real time processed images. The output of which is augmented with sonar data that is validated through the use of an Extended Kalman Filter (Welch & Bishop 2004). The second stage occurs on the transition in navigation between the *x world* and *y world* directions and is applied when the robot reaches the aisle it needs to traverse. This localization stage incorporates horizontal feature extraction from real time processed images, using calculated vanishing points for orientation information. Finally the third stage occurs when the robot is traversing the bookshelf aisle, to locate the desired textbook, and motion is in the direction of the *x world* axis. This final stage involves a sonar and odometry data implementation to accurately reach the desired goal location. The locations of the individual localization stages within an environment map may be seen in Fig. 2.
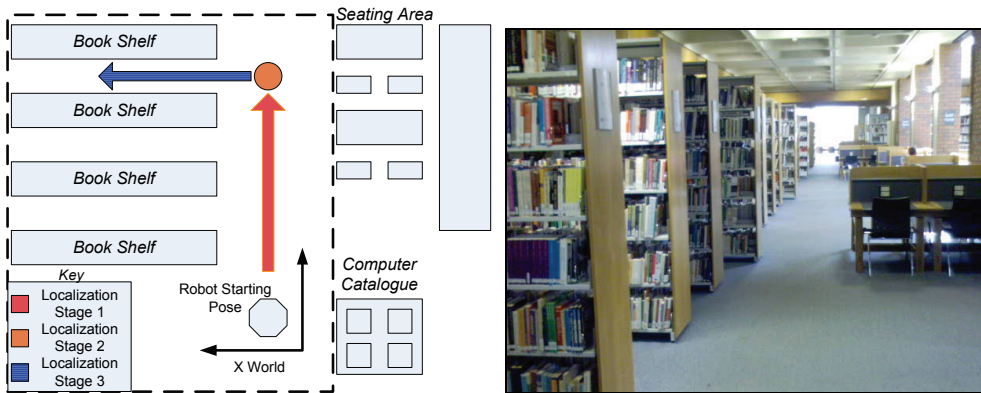
Fig. 2. Environment representation with localization stages

3.4 The implemented localization process

To sense the environment the robot has a suite of nine sonar transducers embedded uniformly as seen in Fig. 3. The robot has two driving wheels and two omni-directional balancing wheels. The origin of the mobile robot's coordinate frame is set at the midpoint between the two driving wheels. As well as using sonar data, a low resolution monocular camera, which points towards the environment features of interest (perpendicular to the direction of motion of the robot), is also used for the localization system. The camera is centrally positioned between sonar sensors seven and eight (Fig. 3). The low resolution of
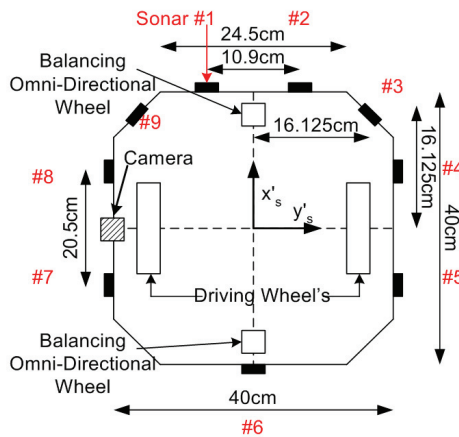
Fig. 3. Robot Sensor System

352 x 288 pixels provides sufficient quality to extract the desired features while reducing the execution time of the image processing algorithms. As the camera coordinate system was chosen to be the same as the world coordinate system and due to the fact that the camera is placed perpendicular to the direction of motion of the robot, the x_r axis coincides with the optical axis and points towards the scene. The image plane is parallel to the (y_r, z_r) plane and

is displaced a distance *f* along the x_r axis. The position and orientation of the robot at time step *k* is represented by the state vector $X(k) = (x_r(k), y_r(k), \theta_r(k))^T$, which corresponds to a cartesian location *(x, y)* and a heading defined with respect to a global coordinate frame *θ* at the kth sampling time. A description of how the vehicles position *X(k)* changes with time in response to a control input *u(k)* and a noise disturbance *w(k)* is known as the Plant Model (Leonard & Durrant-White 1992). The noise source is assumed to have zero mean with a covariance of *Q(k)*. The control input is defined by a translation forward through the distance *T(k)* followed by an anti-clockwise rotation through the *z* axis *Δθ(k)* and is expressed as:

$$u(k) = \left[T(k), \Delta\theta(k)\right] \tag{1}$$

Assuming the pose of the robot at the $(k-1)^{th}$ sampling time is known, the state vector at the kth sampling time is defined as the state transition function $f(X(k-1),u(k))$ plus the noise variable *w(k)* and is expressed as follows:

$$X(k) = f(X(k-1), u(k)) + w(k) \tag{2}$$

or

$$X(k) = \begin{bmatrix} x_r(k-1) + T(k)\cos(\theta_r(k-1)) \\ y_r(k-1) + T(k)\sin(\theta_r(k-1)) \\ \theta_r(k-1) + \Delta\theta(k) \end{bmatrix} + w(k) \tag{3}$$

In this particular application the control input was limited to combinations of three distinct positional movements; a forward motion and 90° left and right turns As the two driving wheels are driven by two independently controlled D.C. motors an additional variable was added to the system state equation Δδ, which is used to compensate for the variability of the individual motors i.e. drift compensation. For this particular application the drift parameter was estimated due to observations of the robot's movement. An adequate drift parameter is quiet difficult to calculate accurately, particularly due to the fact that the two motors are driven independently by two individual power sources. Also different environment floor textures result in different variability in the wheel rotation, i.e. a carpeted floor caused more resistance than a tiled floor. The system state equation now becomes:

$$X(k) = \begin{bmatrix} x_r(k-1) + T(k)\cos(\theta_r(k-1) + \Delta\delta) \\ y_r(k-1) + T(k)\sin(\theta_r(k-1) + \Delta\delta) \\ \theta_r(k-1) + (\Delta\theta(k) + \Delta\delta) \end{bmatrix} \tag{4}$$

Using Eqn. 4 the drive system is said to be calibrated.

According to DeSouza (DeSouza & Kak 2002), it is imperative, either implicitly or explicitly, for a vision system meant for navigation to have a predefined definition of what the camera is supposed to see. After each positional movement, using the *a priori* map augmented with an odometry position estimation, the robot has a preconceived estimate of where it is positioned within the environment. This position estimation is prone to error especially due to drift and wheel slippage. Within the localization procedure, firstly, the robot approximately locates the landmarks using basic odometry. It then determines its exact

position relative to the landmark using a method of observation of vertical or horizontal edges or vanishing points depending on its current pose within the environment. The robot's pose relative to the landmark is extracted using a method of data fusion based on ultrasonic sector scans and image data. The robot's actual pose must be within a solution region, which relates to the individual landmark for localization to be accurately determined. Due to this fact the method of localization occurs approximately every 150cm of movement to ensure that the robot's actual position is always within a solution region and does not allow odometry errors to accumulate. The robot is determined to be within a localization region by the presence of dominant oblique and vertical lines within the cameras field of view. The position of these lines, together with fused sonar data, indicates the robot's current position. Observation of a particular landmark within a localization region is used to correct the robot's pose relative to the landmark. An implementation of the Extended Kalman Filter (EKF) is used to validate the observational sensor values before corrective pose procedures occur. Due to the predictability of the environment and validation through the EKF a single observation is sufficient to determine the robot's pose at each localization region. The localization method is a continuous process carried out throughout the robot's trajectory and is initiated once the robot reaches the first localization region (as predicated by the *a priori* map and the A* algorithm), indicated initially by odometry. The robot's configuration space is divided into localization regions, which are determined by the presence of identifiable features within the environment. As the robot manoeuvres to its goal location, it passes a sequence of landmarks, which primarily consist of rows of bookshelves. Fig. 4, shows an example of the sequence of bookshelves the robot would see as it traverses the library aisles. When odometry determines the robot has reached the first row of bookshelves along its path (based on *a priori* map), the localization process begins and an image is processed from the robot's current location. As the robot progresses through the environment the position of the dominant vertical lines within the image indicate its current position. In Fig. 4 (iii), two dominant bookshelf edges are present telling us that the robot is located directly adjacent the end of a particular bookshelf row. In Fig. 4 (ii) and (iv) a single dominant line is present, indicating that the robot is positioned between the aisle in-between two bookshelf rows and the end of a bookshelf row or vice versa. In Fig. 4 (i), no dominant vertical lines appear indicating the robot is situated perpendicular to the aisle in-between two bookshelf rows. A classic four step approach has been implemented for the image based localization. *1. Image acquisition:* an image is obtained when odometry measurements determine the robot is in the vicinity of a landmark of interest. *2. Image feature extraction:* the image is first processed to enable high quality feature extraction. The Sobel operator is applied to a high thresholded binary image to extract dominant vertical or horizontal edge segments. Once the image edges are obtained, a simple connected component algorithm is used to form distinct line segments. *3: Feature manipulation:* as previously stated the extracted line segments are fitted to known environment features at the robot's estimated position. Within the first localization stage bookshelf edges are the features that are required, so only dominant vertical lines, which are representative of a bookshelf edge, are obtained from the image. The localization method proposes a simple direct fitting of extracted line segments from the image to the estimated feature representation of the environment using a number of techniques. Firstly line segments that do not exceed a minimum length are eliminated. A technique of grouping together split line segments if they are within a threshold vicinity and orientation angle

apart is then implemented. The total length value of grouped lines is examined to ensure that the extracted elements represent an environment variable of interest. As the position of environment landmarks is fixed and the availability of traversable space for the robot is limited the extracted features of interest must also adhere to this restriction. For example, extracted features of interest cannot be within close proximity to each other. The remaining extracted lines are then grouped by the algorithm to determine if suitable as a feature of interest. The modification of the extracted line segments allows for the environment features to be accurately modelled within the image, which results in the variance associated with the robot's current pose to be reduced to zero.

(i)	(ii)
(iii)	(iv)
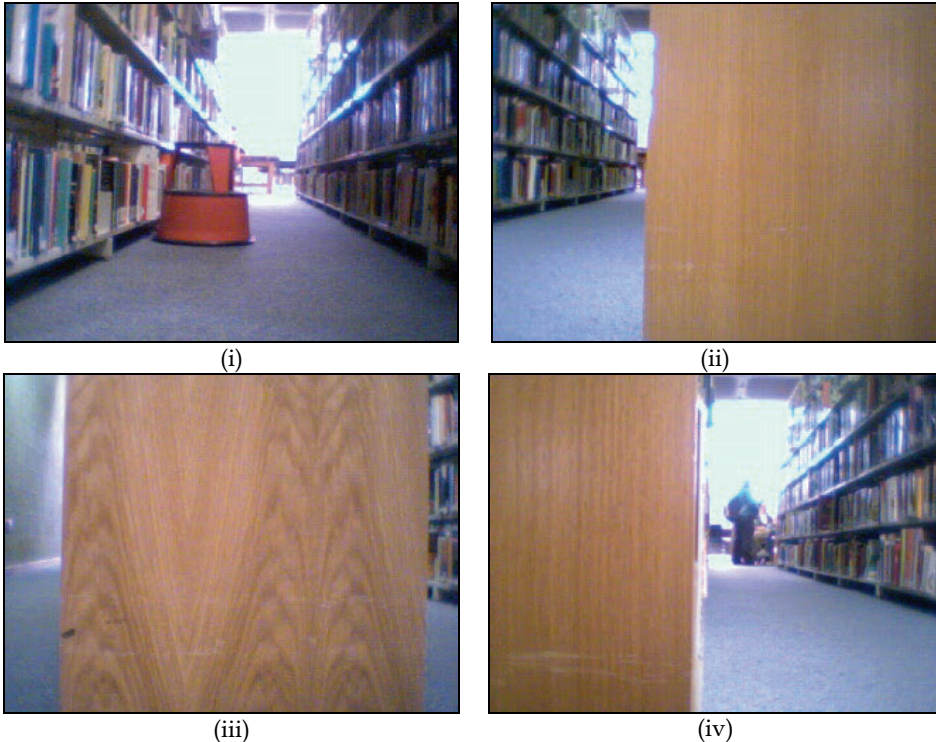
Fig. 4. Sequence of bookshelves

The final step in the localization process is *4: Camera pose computation:* as the cameras pose is fixed relevant (perpendicular) to the robot pose, the information obtained directly from the image is used to calculate the robot's pose. If two dominant bookshelf edges have been obtained as in Fig. 4(iii), the yr coordinate may be determined (*y* coordinate of bookshelf +/- non-centre error of bookshelf within image) and *xr* and *θr,* may be obtained using transmission pulses from the ultrasonic suite of sensors. Two ultrasonic sensors lie uniformly at either side of the camera and as only a single sector scan is used to determine the metric environment measurements at each localization stage, the success of the complete system lies on the accuracy of the readings. In literature sonar has been known to be prone to reflections and false detection. To reduce the probability of erroneous readings, a

validation method incorporating an Extended Kalman Filter (EKF) was utilized. The Kalman filter is a recursive data processing solution to the discrete-data linear filtering problem (Welch & Bishop 2004). It processes all available measurements regardless of their precision, to estimate the current value of the variables of interest, with use of the complete *a priori* knowledge of the system and measurement devices. This includes initial conditions, the statistical description of the system noises, measurement errors and uncertainties in the dynamic model. The output estimate determines the desired variables in such a manner that the error is minimized statistically. The recursive nature means that the Kalman filter does not require all previous data to be kept in storage and reprocessed every time a new measurement is taken. The Kalman Filter is adapted to the non-linear case through the use of the Extended Kalman Filter (EKF) which obtains a suboptimal estimation, using a linear approximation of the system equations.

The EKF relies on two models: a plant model with its associated variance and a measurement model. The plant model is expressed as:

$$\hat{X}_{k/k-1} = f(\hat{X}_{k-1/k-1}, u(k)) + w(k). \tag{5}$$

$\hat{X}_{k/k-1}$ is expressed as the predicted robot location at time step *k* based on *k-1* sensor observations. The zero mean Gaussian noise associated with the plant model is described through *w(k)* with covariance *Q(k)*. The variance of this prediction is defined as:

$$P_{k/k-1} = \nabla f P_{k-1/k-1} \nabla f^T + Q(k) \tag{6}$$

∇f is the Jacobian of the state transition function, $f(\hat{X}_{k-1/k-1}, u(k))$, and is obtained by linearizing about the updated state estimate ($\hat{X}_{k-1/k-1}$) (Leonard & Durrant-White 1992).

$$\nabla f = \begin{bmatrix} 1 & 0 & -T(k)(\sin\hat{\theta}_{k-1/k-1}) \\ 0 & 1 & T(k)(\cos\hat{\theta}_{k-1/k-1}) \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

The measurement model makes explicit the information provided by a sensor measurement in terms of the current vehicle position ($\hat{X}_{k/k-1}$) and the position of geometric features in the environment, with an estimate of measurement noise (Leonard & Durrant-White 1992). It is used to generate predicted observations of each target from the predicted location of each sensor.

The measurement function, $H_{i^{st}}(\hat{X}_{k/k-1}, t)$, expresses a predicted observation $\hat{z}_i(k)$ from sensor *s* to target *t*, as a function of the predicted vehicle position $\hat{X}_{k/k-1}$ and the target geometry:

$$\hat{z}_i(k) = H_{i^{st}}(\hat{X}_{k/k-1}, t) + v(k) \tag{8}$$

where the observation noise measurement is approximated by v(k).

As previously stated each localization target consists of rows of bookshelves which may be expressed as line segments in a 2D representation environment. According to (Leonard & Durrant-White 1992; Meng Quing-Hao et al. 2000), the distance between the jth ($j = 1..9$) sonar and the ith line segment may be expressed as:

$$Hseg_i = Dseg - x_j(k)\cos(\theta_{oi}) - y_j(k)\sin(\theta_{oi}) \qquad (9)$$

where:

(x_j, y_j) are the global coordinates of the jth sonar sensor

$Dseg$ is the distance from the global y_{world} axis to the ith line segment

θ_{oi} is the angle with respect to the x_{world} axis of a perpendicular drawn from the ith line segment to the origin. As all the landmarks of interest lie parallel to the x_{world} axis, θ_{oi} becomes zero.

The environment coordinate system with world and robot axis may be seen in Fig. 5.

The global predicted orientation $\hat{\alpha}_j(k)$ of the jth sonar sensor is used to modify the predicted $Hseg_i$ observation representation of the ith line segment in order to find a true expression of the approximated distance between the jth sensor and the ith line segment. The new approximation becomes:

$$Hseg\_true_i = Hseg_i / \cos(\hat{\alpha}_j(k)) \qquad (10)$$

Fig. 5, graphically displays the difference between the $Hseg_i$ observation representation and the true observation $Hseg_i\_true_i$., as well as the robot's pose at individual time steps.

The predicted state estimate $\hat{X}_{k/k-1}$ is used to compute the measurement Jacobian $\nabla Hseg\_true_i$ for each prediction, which after some mathematical expansion is expressed as:

$$\nabla Hseg\_true_i = \begin{bmatrix} -\cos\theta_{oi} \\ -\sin\theta_{oi} \\ x'_j \sin(\hat{\theta}_{k/k-1} - \theta_{oi}) - y'_j \cos(\hat{\theta}_{k/k-1} - \theta_{oi}) \end{bmatrix}^T \qquad (11)$$

Once an actual observation, $z_j(k)$, is obtained from the sonar sensors a matching process is required between the predicted and actual measurements at each sampling time. For each sonar sensor the innovation variable may be defined as:

$$E_{ij}(k) = z_j(k) - \hat{z}_i(k)$$
$$E_{ij}(k) = z_j(k) - H_{i^{st}}(\hat{X}_{k/k-1}, t) + \hat{v}_i(k) \qquad (12)$$

where $\hat{v}_i(k)$ is the estimated expectation of observation noise.

If the prediction and observation do not correspond to the same sonar sensor the innovation is set to infinity. The innovation covariance is expressed as:

$$S_{ij}(k) = var(E_{ij}(k)) = \nabla Hseg\_true_i P_{k/k-1} \nabla Hseg\_true_i^T + \hat{R}(k) \qquad (13)$$

where $\hat{R}(k)$ is the estimated expected variance of the observation noise $\hat{v}_i(k)$
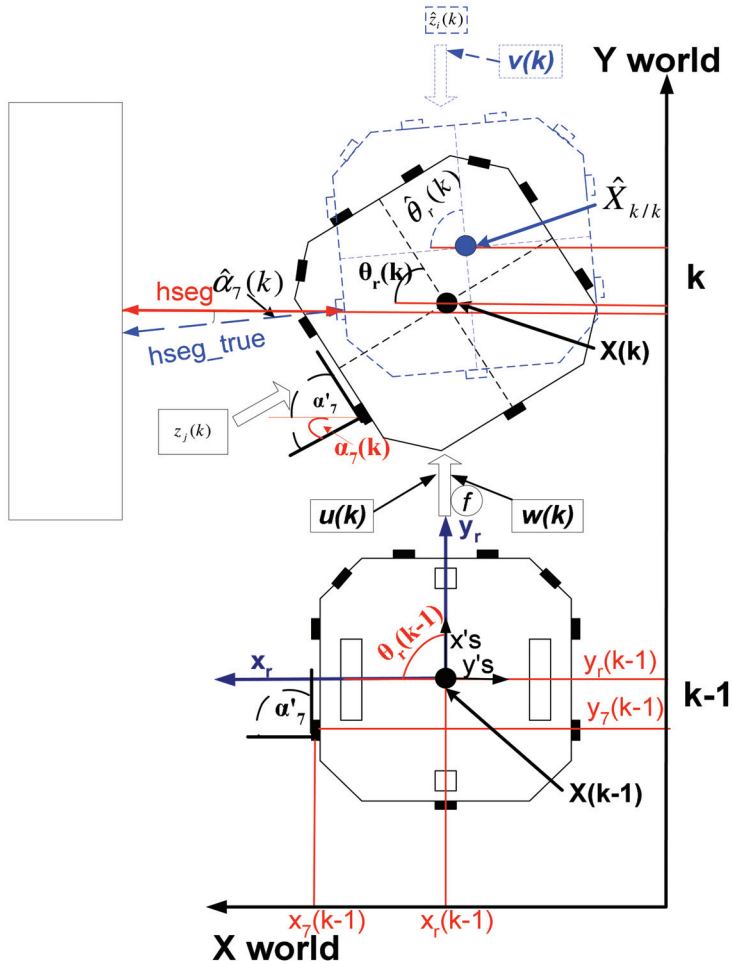
Fig. 5. Environment and Robot Variables at Different Time Steps

If individual sonar readings do not pass through a validation gate, the pose update may be updated using a combination of real readings and estimated readings obtained from the output of the EKF. The xr coordinate is determined from the average distance reading of the two sensors while the orientation of the robot (θr) is determined from the difference between the two readings. If only one vertical line is determined from the image as in Fig. 4(ii) and (iv), the position of the single dominant edge within the image provides essential information about how far the robot is from the centre aisle between the two rows of bookshelves. Using a procedure known as perspective projection, which is approximated by the pinhole camera model, points in the image plane may be mapped to the object plane using simple geometry. Using the pinhole camera model, the distance from the extracted bookshelf edge to the image edge as seen in Fig. 6, may be calculated, which is used to determine the **yr** coordinate for the current position.
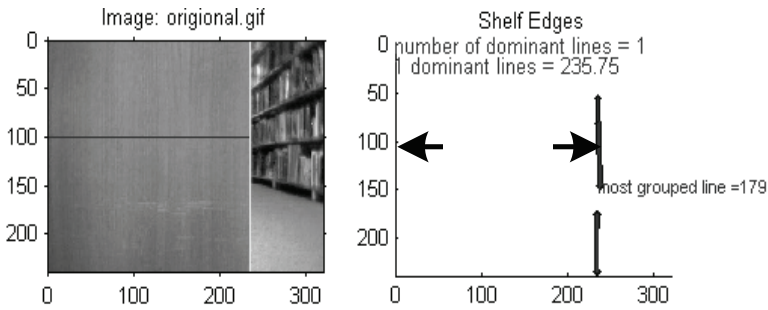
Fig. 6. Single extracted bookshelf edge

The remaining coordinates *xr* and *θr* may be established as before if the detected bookshelf edge lies within the overlap detection region of the adjacent ultrasonic sensors. Both ultrasonic sensors are arranged so that their combined field of view approximately equals the cameras field of view. As both sensor ranges are approximately the same, the position of the bookshelf edge within the image allows us to determine whether the actual bookshelf edge lies in the field of view of both ultrasonic sensors. If both ultrasonic sensors may be used accurately *xr* may be determined by the average of the two readings while *θr* may be calculated using the discrepancies between the two readings. If the bookshelf edge can only be accurately measured by a single ultrasonic sensor, which falls within the EKF validation gate, then the remaining localization parameters are updated using a combination of real and predicted data. The Extended Kalman Filter implementation ensures that a single sonar sector scan is efficient to obtain accurate metric environment information as the measurements are validated, ensuring accuracy, before being passed to the motion control module.

This process of localization continues until the robot reaches the aisle that it has to traverse to obtain the specific textbook. On the last forward positional movement the robot locates itself in-between the rows of bookshelves containing the desired textbook. At this point, the second stage within the localization technique is utilized, which incorporates a technique known as vanishing point detection. With vanishing points, the relationship between 2D line segments in the image plane and the corresponding 3D orientation in the object plane may be established. With a pinhole perspective projection camera mode, a set of parallel lines in 3D space will converge to a single point on the image plane known as the vanishing point. In the case where an image is taken in-between the bookshelves (the aisle the robot has to travel down) each dominant oblique line, which corresponds to an individual bookshelf, corresponds to a parallel line in the real world. Each row of bookshelves also lie parallel to each other, so the dominant oblique lines will all converge to a single vanishing point within the image. At this stage, when the image is processed, only dominant oblique lines are extracted. As vanishing points are invariant to translation and changes in orientation they may be used to determine the orientation of the robot. As the robot's onboard camera is fixed to the robot axis, the vanishing point of the image will tell the orientation of the robot with respect to the orientation of the bookshelves. The most common technique for vanishing point detection is the Gaussian-sphere-based approach

introduced by Barnard (Barnard 1983). The advantage of this method is that it has the ability to represent both finite and infinite vanishing points. In the images taken within the library environment, all the dominant oblique lines will share a common vanishing point due to the parallelepiped nature of the environment. This vanishing point will always be finite and lie within the image plane. In this approach, the vanishing points of the vertical line segments (infinite vanishing points) do not need to be considered, so the complexity of mapping line segments onto a Gaussian sphere is not required. As the dominant oblique lines converge to a single vanishing point, which lies on the image plane, a simple method of the intersection of two line segments will determine the correct location of the vanishing point in pixel coordinates (u, v). The largest dominant extracted oblique line is initially chosen to act as a central line and is intersected with each other extracted line segment, which results in n different vanishing points. The accuracy of the position of the vanishing point depends on the accuracy of the two extracted lines used for intersection. A connected component algorithm is utilized, with components connected if they are within five pixels apart, resulting in groups of related vanishing points. The largest group of connect components is selected, its members averaged, to calculate the exact location of the vanishing point in image coordinates. Using this method erroneous vanishing points are eliminated, as large error points will not have connected partners. If the maximum number of components in the selected list is not greater than three elements, the intersection of lines is not strong enough and a second dominant line is chosen as a central one to intersect with each other extracted line. This ensures that the central dominant line used will actually intersect with the correct vanishing point location. To determine the orientation of the robot from the calculated vanishing point, a method similar to that described in (Jeon & Kim 1999) was implemented. The orientation of the robot is defined by the angle between the optical axis (x_r) and the x_{world} axis of the world coordinate system (θ_r in Fig. 7). Using the method described by Jeon et al., the bookshelf edge D_1 (Fig. 7) may be represented through x_r and y_r, coordinates using the following formula:

$$x_r = \frac{\cos\theta}{\sin\theta} y_r - \frac{y_{r1}}{\sin\theta} \tag{14}$$

z_r is represented by the height of the camera relative to the floor, which is fixed on the robot and known *a priori*. To obtain the u coordinate of the vanishing point, Eqn. 14, is combined with the standard pinhole perspective equation and as x_r goes to infinity u becomes:

$$u = \lim_{x_r \to \infty} y_r \frac{f}{x_r} = f \cdot \tan\theta_r \tag{15}$$

The angle θ_r, which describes the orientation error of the robot, may be described as:

$$\theta_r = \tan^{-1}\frac{u}{f} \text{ , which becomes } \theta_r = \tan^{-1}((u - u_0)/fD_u s_u) \tag{16}$$

u_o is the location of the vanishing point when the robot is orientated exactly perpendicular to the y_r axis, $D_u S_u$ are the camera calibration factors and f is the focal length of the camera.
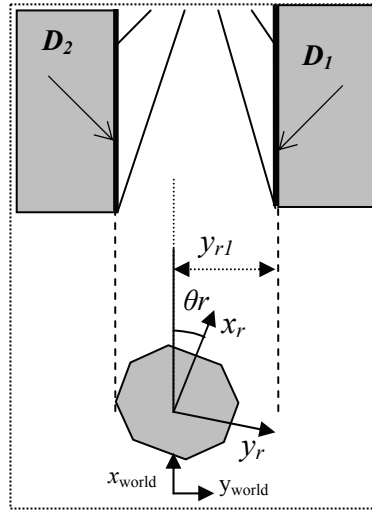
Fig. 7. Vanishing point orientation

When the orientation of the robot is to the right, the vanishing point is located to the left of the image and θ_r is a negative number, when the orientation of the robot is to the left the vanishing point is to the right and θ_r is a positive number. Fig. 8 shows the calculation of the vanishing point when a) the robot's orientation was perpendicular to the bookshelves and the vanishing point is located at the centre of the image and b) when the robot was turned slightly to the right and the vanishing point is located to the left of the image. The use of simple feature extraction (i.e. straight line extraction) in the algorithm implies that even in adverse lighting conditions it is always possible to extract the acquired information. Even if
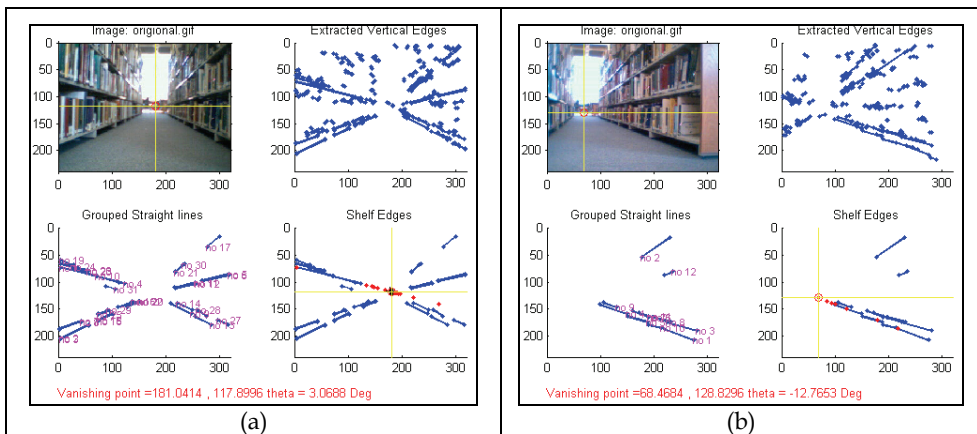
Fig. 8. Vanishing point detection a) robot orientation slightly to left as the vanishing point located slightly to right of image, θ_r calculated to be 3.07° b) robot orientation to right as vanishing point to left of image, θr calculated to be -12.76°

only part of a line gets extracted this is still sufficient for the algorithm to operate correctly. Another advantage of this technique is that when calculating the vanishing point, all the lines extracted converge to a single point so only partial image information is needed for accurate vanishing point calculation. If part of the image becomes occluded (i.e. by a person), it will still be possible to calculate an accurate vanishing point. At this point, the robot uses the orientation information to turn to face in-between the rows of bookshelves and the third stage in the localization process occurs. The robot's orientation is now known and the robot uses its ultrasonic sensor suite to determine the robot's location with respect to the built in *a priori* map. Once the robot enters the row of bookshelves containing the desired textbook, a unique ultrasonic pattern is searched for. Each sensor reading will change sequentially and uniformly on both sides on entering the aisle. Once the ultrasonic pattern has been determined the robot is said to be at the beginning of the aisle and the robot's position with respect to the *a priori* map is known. From this point, as the robot's position and orientation are known, odometry alone is used to traverse the remaining part of the aisle to locate the textbook. Once the robot is travelling down the aisle between the bookshelves the ultrasonic readings are used to keep the position of the robot within the centre of the aisle until it reaches the desired textbook location.

4. Meaningful social Interaction

As previously stated, the success of a service robot is also highly dependent on its ability to interact with its users in a natural and beneficial manner. For a service robot to be successfully implemented within a dynamic real world environment the following described principles must be met.

1. *Create Non Intrusive Devices*

In the development of an assistive system one of the main criteria to be realized is that it is imperative that the system is non-invasive. Giuliani et al. (Giuliani et al. 2005), expressed the term non-invasive as the actions performed by the system as a whole on the environment should occur pro-actively and only when they are beneficial to the assisted person.

2. *Do not Deskill*

The user must be actively involved in the task applied, According to Hoppenot and Colle (Hoppenot & Colle 2000), one of the very principles of aid of an assistive system is that the system must not do for, but compensate the action deficiency of the disabled or elderly person. The person receiving assistance from the technology must be actively involved in the participation of the task.

3. *Build on Existing Ideologies of How Things Work*

Natural communication cues are essential to the successful development of assistive and service robotics. As humans we tend to anthropomorphise our personal belongings and researchers have come to realise that for a service robot to be accepted within a modern socially dynamic environment it is essential that natural human communication methods are supported. This may be achieved through the use of conversation, the ability to collaborate on a task, the use of facial expressions and gestural behaviours and also through tactile approaches. In (DiSalvo et al. 2002), DiSalvo et al. stated that humans prefer to interact with machines in the same way that they interact with people. The human-robot social interaction system must also be initiated and interaction and co-operation must be encouraged between the robot and the user. The robot must also be integrated into the life of the user in a natural and beneficial way.

4. *Simplify Functionality*

An elderly or disabled person interacting with the robot may have reduced physical and/or cognitive capabilities, which can represent a barrier for the use of high-tech instrumentation. Studies carried out as part of the RoboCare project (Scopelliti et al. 2004; Cesta et al. 2005) with the aim of assessing people's attitudes and preferences towards a domestic robot, revealed that elderly people seem to recognise its potential usefulness in the house, but are somewhat afraid of potential damages caused by the robot and of intrusion in their privacy. In addition, older people showed a preference for the robot not to be free to move inside the house and would expect the robot to be programmed in a fixed way. The studies have shown that with respect to younger respondents, older participants did not appreciate the stimulating the intriguing side of the autonomous agent and tended to emphasize the practical benefits. However when asked about the specific tasks the robot could perform in their home the respondent's answers were somewhat vague or unrealistic. The user requirements for the elderly, disabled or children may require very constrained modes of communication and simplistic demonstration capabilities. Any complexity must be hidden and systems must have very acceptable controls and operational modes.

5. *Promote Trust*

An elderly or disabled person may be encouraged to interact with a robot if he/she can relate to the robot's personality through its structural from and the cooperation between the robot and the user. When designing an interacting robot the design must be applicable while functional, yet there is a need to project a certain amount of humanness so that the user will feel comfortable socially engaging the robot. Assurances must also be made so as not to fall into the Uncanny Valley trap. Mashiro Mori developed a theory known as the Uncanny Valley theory, which states that as a robot's appearance increases in humanness the user expects the robot to act in an increasingly human like manner. The user may become disillusioned about the robot's abilities and functionality if it appears to be too human-like. According to psychological studies carried out by Scopelliti et al. (Scopelliti et al. 2004) in order for a robotic application to be successful in a Robot Assisted Activity (RAA) for the elderly it is necessary for the elderly user to perceive the robot as a "friendly creature", which is of some help in their everyday life. One of the principles DiSalvo et al. (DiSalvo et al. 2002) proposed is that when designing a service robot the robot's morphology must match its intended function, enabling the user to feel comfortable with the use of the robot.

6. *Adapt to Changing Needs/Environments*

Successful assistive devices must be customisable to accommodate different levels of task requirements, user abilities and widely different personal environments. They must also have the ability to adapt in-situ to environmental and task changes including error events. Research has shown that robots may have a very strong novelty effect, but experiments carried out by Kanda et al. (Kanda & Ishiguro 2005) with their communication robot, ROBOVIE, have shown that this novelty, which leads to initial eager interaction from participants turns quickly into boredom with the robot. An approach of interactive behavior is required to enable long term positive interaction with the robot and the participant. Researchers have investigated many different possibilities to enable this interaction system and have come up with several solutions, which are investigated through different interaction interfaces and various shapes and forms such as pet robots, life like robots and purely interactive systems. In (Dautenhahn 2004), Dautenhahn proposed that the social

personality of a robot should grow through a socialization process similar to that observed in animals such as dogs. Also, mass produced systems, which can not be specifically configured to each users environment must be easily customisable (by the user or carer) in order to become personalised for each users needs.

7. *Do not Stigmatize*

Particularly elderly individuals in general do not like to be associated with devices coupled with the stigma of ageing. For a robotic system to be implemented as an aid or assistant for elderly individuals within a public environment it must appear to be of universal benefit and not be categorised as a device associated with a specific impairment or weakness.

8. *Enable Reality, Do not try to Substitute it*

Research has shown that social activities and contacts improve dependent elderly person's well-being. Dependent elderly people who are a member of a club, those who often meet their friends and relatives and those who often talk with their neighbours declare a higher satisfaction than the rest (Mette 2005). Assistive systems should be designed to augment realistic situations and to cater for various impairments that impede an individual from completing usual every day tasks rather than replacing these tasks with virtual reality and alternative solutions. One of today's most successful service robots PEARL (Pineau et al. 2003), had a primary objective to create an assistant that augments rather than replaces human interaction.

To incorporate the design principles described above, one of the most important aspects in the creation of a successful assistant robot is the human-robot interaction system. There are three main methods to encourage interactions between the robot and the user. The first is through mechanical actuators. The second method is through virtual reality techniques, where the user may become immersed in the robot's virtual world and the third method is through computer animated agents. Computer animation techniques have the ability to generate a 3D rendered face that has many degrees of freedom to generate facial expressions. To achieve successful human-robot interaction between "LUCAS" and the user, and to incorporate the above principles, a graphical interface displaying a human-like animated character was implemented. The software used in the creation of the character is known as the Rapid Application Developer (RAD), and is part of the CSLU toolkit created by the Oregon Health & Science University (OHSU), Centre for Spoken Language (RAD 2005). This software uses graphical animation techniques to create a 3D face of a human-like character. The face is displayed through a laptop computer screen embedded into the robot's structure as seen in Fig. 1.

The authors acknowledge that human-robot interaction is a very complex and multifaceted area, but wish to provide a simple two-way communication system between the robot and user. This communication system must be both beneficial and natural to the user and adhere to the above design principles, which are critical to the successful application of the robot. To achieve this, the design principles are incorporated into a simple communication application that occurs between the robot and the user as the robot completes its task. The design principles are met in the following manner: 1. *Create Non Intrusive Devices:* The functionality of "LUCAS" is only applicable when the user initiates interaction and does not interfere with library users unless requested. This occurs when the user approaches a specific library catalogue computer containing a graphical user interface (GUI) with the existing library catalogue. The user then selects the desired textbook, which is communicated wirelessly to "LUCAS". An A* (Stentz 1994) algorithm is initiated based on

the textbook selection, which leads the robot to the location of the selected textbook through the utilization of the localization algorithm previously described. 2. *Do not Deskill:* Even though "LUCAS" eliminates the complexity and intricacy of locating a library textbook for the user by physically locating the textbook, the user must still participate in the task by selecting the required textbook and then follow "LUCAS" to the location of the textbook. 3. *Build on Existing Ideologies of How Things Work:* For a service root to be successful a natural method of communication must be established between the robot and the user and the interaction must be similar to the ways in which humans interact with each other. The selection of the CSLU toolkit (RAD 2005) for interaction purposes enables a natural communication method such as speech. The appearance of the character resembles a familiar friendly face but its animated features ensure that the human-robot interaction application does not fall into the Uncanny Valley trap. Once a textbook is selected, the robot reads the title of the textbook to the user using its voice synthesiser, the user then confirms that it is the correct textbook by pressing the button incorporated into the robot's structure (the interaction button). "LUCAS" then encourages the user to follow it to the location of the specific textbook via vocal dialogue and visual text prompts. "LUCAS" again communicates with the user on reaching the desired textbook location, informing the user of the specific book location, before returning to its home-base location. 4. *Simplify Functionality:* Simplicity of use, ease of interaction and avoid over-complication are phrases that are often iterated by researchers involved in assistive technology in relation to robotic functionality. The success of a service robot depends on its perception of usability. In applying "LUCAS" to the implemented environment a single mode of functionality was incorporated, i.e. "LUCAS" guided the user to the location of the user specified textbook. 5. *Promote Trust:* The appearance of the robot through its interacting character, ability to express emotions, its spoken dialogue system, physical structure (the size of the robot is viable in human space) and also its slow movements encourage human participation and help the user to feel comfortable with the use of the robot. 6. *Adapt to Changing Needs/Environments:* If an existing computer library cataloguing system is not already present within a specific library, an alternative implementation may be described as follows. An elderly or disabled person, a new library user or simply a user unfamiliar with the libraries structure may not be searching for a specific individual textbook but for a range of textbooks on a particular topic such as gardening, knitting, computer programming etc. An alternative approach to the implementation of "LUCAS" is that if a user approaches the reception desk within the library requesting a specific range of textbooks, the library attendant may transmit the information regarding the topic of interest wirelessly to "LUCAS" and request the user to follow the robot. The robot may again utilize its database and activate navigation and localization algorithms in a similar fashion to the original application and use its human-robot interaction methods to encourage the user to follow it to the desired location, which holds the textbooks of interest. Using this implementation, a disabled or elderly user may still maintain their independence without totally relying on human intervention. If the robot was equipped with a robotic arm and barcode scanner, the robot may be applied to traverse the library at night and physically record the location of textbooks within the library aisles. This may serve both as a method to continuously update "LUCAS" own database but also simultaneously provide librarians with a method to determine if particular textbooks were placed in incorrect locations according to the Dewi Decimal system or to identify missing textbooks. The design of "LUCAS" allows for portability, which implies that the robot may

be used in variety of settings for assistive purposes. An example of this would be as a store assistant robot, whose functionality would be to guide users throughout aisles within a supermarket to locations of user specified groceries. The robot may also be applied as an interacting warehouse delivery robot or an usher within a public office building. 7. *Do not Stigmatize:* The completed system, even though not designed specifically to assist elderly or disabled individuals may be of important benefit to this increasing pool of potential users. Due to its universal design of being applicable to the population as a whole, i.e. it was not designed to assist an individual with a specific impairment thus it is not fitted with any devices associated with the stigma of ageing or disability. The universal design ensures that users of the system are not categorized by association. 8. *Enable Reality, Do not try to Substitute it:* The complete system supports interaction and usability within a real environment. As previously stated a library may be a place of social interaction and promote a cogitatively challenging hobby for elderly or disabled individuals. The introduction of an assistive system such as "LUCAS" within a public library may facilitate individuals who normally find the task to obtain a textbook a time consuming and arduous process. The system promotes the use of existing human-occupied facilities rather than replace or augment these with virtual reality or on-line techniques.

5. Results

5.1 Implementation of the robotic system

Interactions with the robot, thus initiating its functionality, occurs when the user approaches a specific library catalogue computer. A graphical user interface (GUI) created in Microsoft's Visual Basic contains the existing library catalogue. The user may search the catalogue for the desired book. Once the correct book is selected, interaction with the robot occurs when the user presses the "*Activate Robot*" button inserted at the bottom of the GUI, this initiates a wireless WiFi network transaction of information on the chosen book between "LUCAS" and the catalogue computer. "LUCAS" then utilizes a built in database created in Microsoft Access, which holds a corresponding geographical location coordinate in the form of *x world, y world, z world* (world coordinate system). The *x world* and *y world* information are used to locate the physical location of the book by implementing its path-planning algorithm (A*, Stentz 1994) with its *a priori* map (grid based metric map). The *z world* coordinate corresponds to the bookshelf number of the books location. The robot reads the title of the book to the user using the synthetic voice feature of the human computer interface, then encourages the user to follow it to the location of the specific book via vocal dialogue and visual text prompts.

The following section describes the results of the implemented localization algorithm within the actual environment. To demonstrate the localization process a path of approximately 550cm in each direction with a start node of (1,0) and goal node of (3,9) within a grid based metric map was executed. The path involves the robot passing four different localization regions (three bookshelf rows and a fourth vanishing point region) labelled 1-4 in Fig. 9. Each image and its processed counterpart are seen in Fig. 9, and the corresponding results are displayed in Table 1. In Fig. 9, each white arrow represents a forward positional movement, the variations in arrow length correspond to odometry errors. At point 2, the sonar determined that the robot was too close to the shelf, so its orientation was altered by 10°, this ensured that both sonar sensor readings will overlap, within the next localization region, to obtain accurate results. The shaded arrow between points 3 and 4 indicates a longer forward positional movement to allow the robot to be in a correct position for the

vanishing point implementation. The circles indicate 90° turns. After the vanishing point implementation the robot accurately enters the aisle, and for further traversal of the aisle stage 3 in the localization process occurs. Sonar readings are used to correct orientation and ensure that the robot is centred within the aisle. On reaching the goal location the robot communicates with the user, completes a 180° turn and returns to its home base location. For this particular implementation the localization method was not activated on the return home journey (indicated by dotted arrows). As can be clearly seen, odometry alone with obstacle avoidance and path planning is insufficient for accurate navigation.

Table1: Fig. 9, image #	Sonar 1 cm	Sonar2 cm	EKF Sonar 1 cm	EKF Sonar 2 cm	xr cm	yr error cm	θ °
1	37.9	57.86	37.39	38.46	38.17	-38.0697	0
2	36.6	62.21	43.3	44.38	40.49	-36.4298	0
3	82.82	82.99	67.12	68.2	82.83	4.484	-0.17
4	n/a	n/a			n/a	n/a	3.4951

Table 1. Output results from localization algorithm. For both images 1 and 2, only a single sonar sensor was obtaining accurate range data so the outputted xr location was updated using one measured range reading and one predicted output from the EKF.

From Table 1, it may be seen that in both images 1 and 2, only a single sonar sensor was accurate (sensor left of the camera). This was due to the fact that the shelf edge did not lie within the accurate range of the second (right) sonar sensor. As the first sonar readings fell within the validation gate the second sensor reading was updated using the predicted value resulting from the EKF process. The resulting xr and yr parameters were updated using the validated sonar readings combined with the predicted values. From experimental observations, it was determined that when the pose was extracted from combinations of predicted and actual data, that more reliable results were obtained by setting the orientation to 0° (angle between optical axis and x-world axis) in these situations. This is due to the fact that even very small errors in orientation may lead to the robot becoming lost. As can be seen from image 3 in Fig. 9, using predicted sonar data when real data is unavailable, due to numerous reasons, does not affect the overall localization process. The features of interest (bookshelf), within the next localization stage, seen in image 3 in Fig. 9, were accurately extracted due to the robot being correctly positioned within the localization region. The robot's complete functionality is described in story board from in Fig. 10.

5.2 Human-interaction testing

The final aspect of testing of the developed robot was to test the robot within the target environment with human subjects. Ethical approval was attained from the University of Limerick Ethics Board to carry out a trial with volunteers. As part of its implementation, 7 volunteers aged between 22 and 55 utilized the functionality of "LUCAS" and subsequent interviews were held. Out of the volunteers 100% thought the robot was a success and would use it again. 85% thought the existing process was time consuming but only 14% would ask a librarian for assistance. 85% liked the interaction system and 42% thought a mechanical interface would be "freaky". From the interviews, two main faults of the robotic system were extracted 1: The robot moved too slow, this was due to the fact that an image was processed every 150cm, which resulted in a stop and go motion, with delays for image processing. 2: 57% of the users would have preferred a taller robot, approximately eye height.

Fig. 9: Robots path to goal

Database Confirms Location

shelf	reference	number	title
		books	
2	"TRC 822.33/ROM"	4,2	"Romeo and Juliet"
1	"515.2483/STE"	5,2	"Fourier analysis : an introduction"
2	"821.8/WIL"	5,2	"complete poetry"
3	"842.914/BEC"	6,2	"Beckett radio"
1	"833.912/HES"	7,2	"sttepenwolf 1974"
3	"TRC 823.914/DAH"	4,5	"Marvellous"
1	"823.912/JOY/M"	5,5	"James Joyce"
2	"Popular Reading /COE"	4,6	"The alchemist"
3	"820.994150004/COR"	5,6	"After Yeats and Joyce : reading modern Irish lite

User Initiates Robot

Robot Takes Control

start

User Approaches Catalogue Computer

Robot Communicates with User

A* Implementation

Search found solution path
Node position : (1,0)
Node position : (1,1)
Node position : (1,2)
Node position : (1,3)
Node position : (1,4)
Node position : (1,5)
Node position : (1,6)
Node position : (1,7)
Node position : (1,8)
Node position : (1,9)
Node position : (2,9)
Node position : (3,9)
Solution steps 11
SearchSteps : 24

Goal Path

Return Home Path

Robot Communicates with User on Reaching goal location

Fig. 10: Story board implementation

6. Conclusion

This chapter describes the complete development of a service robot primarily through localization and navigational algorithms and human-robot interaction systems. The service robot was applied as a library assistant robot and its implementation was discussed and evaluated. The chapter was divided into two main topics, the localization system and the human interaction system. The localization system consists of simple modular systems incorporating fusion of odometry, monocular vision and EKF validated sonar readings. The localization method proposed here is a continuous localization process rather than a single localization step and results in fast low cost localization within a specific indoor environment. As the localization process is continuous odometry errors do not have time to accumulate, which implies that the initial position estimation using just basic odometry is relatively accurate. This allows the robot to apply the individual localization procedure for each specific location based on odometry alone. The reduction in image processing techniques such as the use of a monocular vision system rather then a stereo vision system, straight line extraction and simplified vanishing point estimation result in a fast and very effective localization system. The use of simple feature extraction (i.e. straight line extraction) in the algorithm implies that even in adverse lighting conditions it is always possible to extract the acquired information. Even if only partial features are extracted, this is still sufficient for the algorithm to operate correctly. The fact that the robot uses a very simple *a priori* map and does not use pre-recorded images to aid localization, results in faster

execution times and leaves the robot's central processor free to deal with its other tasks such as path planning, obstacle avoidance and human interaction. Through the use of the EKF, feature fitting within obtained images and restricting output errors to contain only valid outliners, the system implemented may accurately localize the robot within the proposed environment.

The second stage of the chapter deals with the human-robot interaction system and the principles required for a robotic application to be a success. "LUCAS" interacts with its users using a spoken dialogue system created using the CSLU toolkit, which allows the user to interact with the robot in a natural manner. The critical design principles required for the application of a successful service robot have been incorporated utilizing the interaction system and the robot's functionality. The design of this particular robot allows for portability, which implies that the robot may be used in a variety of settings for assistive purposes. An example of this would be a store assistant robot, whose functionality is to guide users throughout aisles within a supermarket to locations of user specified groceries or as an interacting warehouse delivery robot.

The application described in this chapter is significant, not just because of the localization methods and applied functionality, but also because its implementation within a real-world environment represents a high level of performance, through navigation, localization and human–interaction, all at low costs. As 100% of interacting volunteers thought the robot was a success when asked about its functionality and also stated that they would use it again shows the acceptance of such a system within the proposed environment. Even though some researchers may believe that locating a textbook is a simple task but if simple applications may successfully be applied in real-world environments, the future of robots cohabitating with humans may become reality, through the initial development of successful simple applications. Some of this research has demonstrated that advanced technology through complete solutions may be achieved through simple and effective modular systems, but to be successful the technology must be easy to use, meet the needs of the specific environment and become an accepted component of daily life. As a result of the implemented testing, several factors have been raised to potentially improve the robot's performance and usefulness, such as increased processing power to minimize the time required for image processing. The human-robot interaction method may be further enhanced by utilizing additional features of the RAD application, specifically the speech recognition and its ability to visualize various gestures and emotions. Also the functionality of the robot may be further enhanced with the addition of a robotic arm, which would allow the robot to physically fetch the desired textbook for the user. This would allow the robot to cater for an increased pool of potential users. The structure of the robot may also be improved by the addition of a walking aid system to support elderly users as they travel to the textbook location.

7. References

Arras K. O. & Tomatis N. (1999). Improving Robustness and Precision in Mobile Robot Localization by Using Laser Range Finding and Monocular Vision. *Proceedings of*

Third European Workshop on Advanced Mobile Robots (EUROBOT '99), Zurich, Switzerland.

Barnard S.T. (1983). Interpreting Perspective Images. *Artificial Intelligence*, 21: 435-462.

Bolmsjo, G., Neveryd H. & Eftring H. (1995). Robotics in Rehabilitation. in *Proceedings of IEEE Transactions on Rehabilitation Engineering.* 3(1). 77-83.

Breazeal C. (2001). Socially Intelligent Robots: Research, Development, and Applications. *Proceedings of IEEE International Conference on Systems, Man and Cybernetics,* pp. 2121-2126, Tucson, Arizona.

Breazeal C. (2002). *Designing Sociable Robots*, MIT Press, Cambridge, MA.

Bruce A., Nourbakhsh I. & Simmons R. (2001). The role of expressiveness and attention in human-robot interaction. *Proceedings of AAAI Fall Symposium Emotional and Intelligent II: The Tangled Knot of Society of Cognition.*

Casals A., Villa R. & Casals D. (1993). A soft assistance arm for tetraplegics. *1st TIDE congress*: pp. 103-107.

Cesta A., Farinelli A., Iocchi L., Leone R., Nardi D., Pecora F. & Rasconi. R. (2005). "Robotically Rich" Environments for Supporting Elderly People at Home: the RoboCare Experience. *Proceedings of AISB '05: Social Intelligence and Interaction in Animals, Robots and Agents,* Hatfield, UK.

Dallaway J.L. & Jackson R.D. (1992). RAID - A vocational robotic workstation. *Proceedings of ICORR 92.*

Dallaway J.L., Jackson R.D. & Timmers P.H.A. (1995). Rehabilitation Robotics in Europe. *IEEE Transactions on Rehabilitation Engineering*, 3: 35-45.

Dario P., Guglielmelli E. & Allotta B. (1995). Mobile Robots aid the disabled. *Service Robot*, 1(1): 14-18.

Dario P., Guglielmelli E. & Allotta B. (1996). Robotics in medicine. *IEEE Robotics and Automation Society Magazine*. **3:** 739 - 752.

Dautenhahn, K. (2004). Robots we like to live with?! A Developmental Perspective on a Personalized, Life - Long Companion. *Proceedings of IEEE Ro-man* Kurashiki, Okayama, Japan.

DeSouza, G.N. & Kak A.C. (2002). Vision for Mobile Robot Navigation: A Survey. *Transactions on Pattern Analysis and Machine Intelligence*, 24(2): 237-267.

DiSalvo C.F., Gemperle F., Forlizzi J. & Kiesler S. (2002). All robots are not created equal: The Design and Perception of Humanoid Robot Heads. *Proceedings of Conference on Designing Interactive Systems*, 321-326, London,England.

Evans J. (1994). Helpmate: an autonomous mobile robot courier for hospitals. *Proceedings of IEEE International Conference on Intelligent Robot and Systems*, pp.1695-1700, Munich, Germany.

Fong T., Nourbakhsh I. & Dautenhahn K. (2003). A survey of socially interactive robots. *Robotics and Autonomous Systems*, 42: 143-166.

Giuliani M.V., Scopelliti M. & Fornara F. (2005). Coping Strategies and Technology in Later Life. *Proceedings of Proceedings of Workshop on Robot Companions, AISB '05* Convention, Hatfield, UK.

Graf B., Hans M. & Schraft R.D. (2004). Care-O-Bot II - Development of a Next Generation Robotic Home Assistant. *Autonomous Robots*, 16: 193-205.

Hoppenot P. & Colle E. (2000). Robotics Assistance to disabled and elderly people. *Proceedings of IMACS'2000*, pp.261, Lausanne, Switzerland.

Jeon S.H. & Kim B.K. (1999). Monocular-based Position Determination for Indoor Navigation of Mobile Robots. *Proceedings of IASTED International Conference on Control and Applications*, 408-413, Banff.

Johnson M.J., Guglielmelli E., DiLauro G.A., Laschi C., Pisetta A., Giachetti G., Suppo C., Perella Y., Carrozza M.C & Dario P. (2003). The Robotic Appliance: The Next Generation Personal Assistant? *Proceedings of The 11th International Conference on Advanced Robotics*, 5 -10, Coimbra, Portugal.

Kanda T. & Ishiguro H. (2005). Communication Robots for Elementary Schools. *Proceedings of Workshop on Robot Companions, AISB 05* Convention, Hatfield, UK.

Kawamura K., Bagchi S., Iskarous M. & Bishay M. (1995). Intelligent Robotic Systems in Service of the Disabled. *IEEE Transactions on Rehabilitation Engineering*, 3(1).

Kawamura K., Bagchi S., Iskarous M., Pack R.T. & Saad A. (1994). An intelligent robotic aid system for human services. *Proceedings of AIAA/NASA Conference on Intelligent Robotics in Fields, Factory, Service and Space*, 413-420.

Krotkov E. (1989). Mobile Robot Localization Using a Single Image *in Proceedings of the IEEE International Conference on Robotics and Automation.* Vol. 2, May 1989, pp. 978-983.

Leonard J.J. & Durrant-White H.F. (1992). *Directed Sonar Sensing for Mobile Robot Navigation*, Kluwer Academic Publishers.

Meng Quing-Hao, Sun Yi-Cai & Cao Zuo-Liang (2000). Adaptive Extended Kalman Filter (AEKF) - Based Localization Using Sonar. *Roboticia*, 18: 459 -473.

Mette C. (2005). Wellbeing and Dependency among European Elderly: The Role of Social Integration. *ENEPRI Research Reports*.

Pérez J.A., Castellanoa J.A., Montiel J.M.M., Neira J. & Tardós J.D. (1999). Continuous Mobile Robot Localization: Vision vs. Laser. *Proceedings of IEEE International Conference on Robotics and Automation,* Detroit, Michigan.

Pineau J., Montemerlo M., Pollack M., Roy N. & Thrun S. (2003). Towards robotic assistants in nursing homes: Challenges and results. *Robotics and Autonomous Systems*, 42: pp.271-281.

RAD. (2005). "http://www.cslu.ogi.edu/toolkit."

Scopelliti M., Giuliani M.V., D'Amico A.M. & Fornara F. (2004). If I had a robot at home…Peoples representation of domestic robots. In: *Design for a more inclusive world* J. C. S. Keates, P. Langdon, & P. Robinson, 257-266. London, Springer-Verlag**:**

Severinson-Eklundh K., Green A. & Huttenrauch H. (2003). Social and collaborative aspects of interaction with a service robot. *Robotics and Autonomous Systems*, 42: 223-234.

Stentz A. (1994). Optimal and Efficient Path Planning for Partially-Known Environments. *Proceedings of IEEE International Conference on Robotics and Automation.*

Thrun S., Schulte J. & Rosenberg C. (2000). Interaction with Mobile Robots in Public Places. *IEEE Intelligent Systems*: 7-11.

Topping M. & Smith J. (1998). An overview of HANDY1, A Rehabilitation Robot for the severely disabled, in Proceedings of CSUN

U.N. (2004). "World Population prospects: The 2002 Revision Population Database, http://www.un.org/esa/population/unpop.htm."

Welch G. & Bishop G. (2004). An Introduction to the Kalman Filter. *Technical Report,* Department of Computer Science, University of North Carolina at Chapel Hill.

3

Human – Robot Interfacing by the Aid of Cognition Based Interaction

Aarne Halme

Helsinki University of Technology
Finland

1. Introduction

There are two challenging technological steps for robots in their way from factories to among people. The first to be taken is to obtain fluent mobility in unstructured, changing environments, and the second is to obtain the capability for intelligent communication with humans together with a fast, effective learning/adaptation to new work tasks. The first step has almost been taken today. The rapid development of sensor technology – especially inertial sensors and laser scanners – with constantly increasing processing power, which allows heavy image processing and techniques for simultaneous localization and mapping (SLAM), have made it possible to allow slowly moving robots to enter in the same areas with humans. However, if we compare the present capability of robots to animals, like our pets in homes, it can be said without no doubt that improvements are still possible and desirable.

The second step is still far away. Traditional industrial robots are mechanically capable to change a tool and perform different work tasks, but due to the nature of factory work need for reprogramming is relatively minor and therefore interactive communication with the user and continuous learning are not needed. The most sophisticated programming methods allow task design, testing, and programming off-line in a simulation tool without any contact to the robot itself. Today's commercial mobile service robots, like vacuum cleaners and lawn mowers, are limited to a single task by their mechanical construction. A multi-task service robot needs both mechanical flexibility and a high level of "intelligence" in order to carry out and learn several different tasks in continuous interaction with the user. Instead of being a "multi-tool" the robot should be capable of using different kinds of tools designed for humans. Due to fast development in mechatronics, hardware is not any more the main problem although the prices can be high. The bottlenecks are the human – robot interface (HRI) and the robot intelligence, which are strongly limiting both the information transfer from the user to the robot as well as the learning of new tasks.

Despite huge efforts in AI and robotics research, the word "intelligence" has to be written today in quotes. Researchers have not been able to either model or imitate the complex functions of human brains or the human communication, thus today's robots hardly have either the creativity or the capacity to think.

The main requirement for a service robot HRI is to provide easy humanlike interaction, which on the one hand does not load the user too much and on the other hand is effective in the sense that the robot can be kept in useful work as much as possible. Note that learning of

new tasks is not counted as useful work! The interface should be natural for human cognition and based on speech and gestures in communication. Because the robot cognition and learning capabilities are still very limited the interface should be optimized between these limits by dividing the cognitive tasks between the human brains and robot "intelligence" in an appropriate way.

The user effort needed for interactive use of robotic machines varies much. Teleoperators need much user effort, because the user controls them directly. Single tasks service robots, like autonomous vacuum cleaners, do not demand too much effort, but complexity and effort needed increase rabidly when general purpose machines are put to work. Figure 1 illustrates this situation. The essential question in the development of next generation intelligent service robots is the complexity of their use. Because being just machines for serving human needs, they are not well designed if they need lot effort either for the preparation of a work or monitoring it.
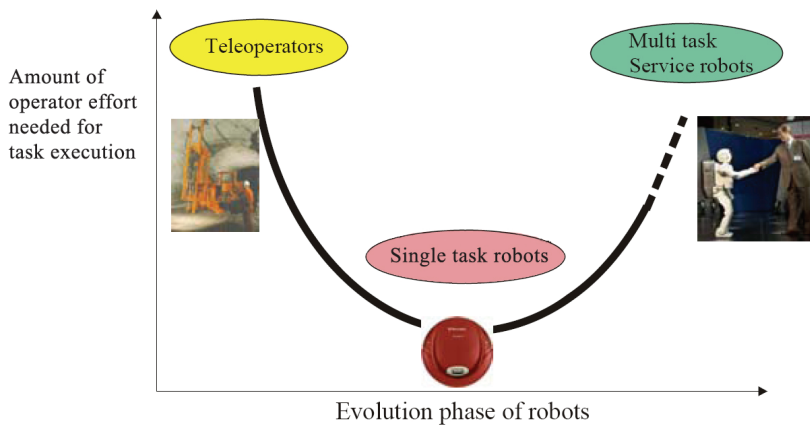
Fig. 1. Illustration of the amount of operator effort needed for task execution during different phases of robot evolution

2. The problem of getting service robots to work efficiently

It is quite clear today that without noticeable progress in the matter, the effort needed for operation of multi-tasking service robots will be even higher than in the case of classical teleoperators, because more data is needed to define the details of the work. The classical teleoperators have evolved greatly since the 1950's and today have reached a high standard of development especially through the development of tele-existence methods and technologies (Tachi, 1999). Teleoperators may be classified into four classes (Fong and Thorpe, 2001) due to their complexity, sensing, and the operator supervision status, but altogether the way these systems are predicted to develop, they will lead to user information loads too large to be practical as a human-robot interface concept for interactive service robots. Thus new concepts are needed.

Intuitively it is clear that such concepts must utilize the superior cognition and reasoning capacity of human brains allowing fusing of different perception information and making conclusions on the basis of insufficient information. This means that controlling the robot must be based mainly on semantic or symbolic information instead of copying motions or

following numeric models, which is the case even in teleoperation commanded in task level. The problem is how to get the exchange of information in such a way that the robot "understand orders" and actively interacts while performing them rather than follow them slavishly. Only in this way is there a possibility of keeping the user load within acceptable limits. The ultimate goal is to let the robots do skilled work, thought at first by the user, but gradually learned by the robot as an independent performance (Halme & all, 2006). By skilled work we mean here work tasks, which are individual in the sense that they need sensing and detail planning each time when executing, although the general plan how to do them has been already taught to the robot. Most of the work tasks in our living environment are such one because of inherent disorder and changes that occur without warning.

3. Interfacing with cooperative robots – the concept of common spatial awareness (CSA)

The approach to be considered in the following is based on the idea that as much as possible of the information needed to make the robot to perform a work task is given in symbolic or schematic form instead of numeric data. The environment itself should be utilized when possible. The human user can create relatively easily such information when using his/her cognition in the work place. The cognitive capacity of the robot is programmed so that transferring the user's will to the robot happens mostly in the form of dialogue, which makes sure that the task is uniquely defined and can be executed by the robot's internal commanding system. The dialogue is based on concepts and objects in a virtual world called Common Spatial Awareness, CSA. The CSA is a model of the working environment, which both entities can understand in the same way through their own cognition system. The detailed meaning of CSA will be explained later, but it is good to note that we use the term "awareness" here in a different meaning than psychologists, who often include feelings and imagination in this concept. In our pragmatic awareness concept the physical working place is the place, where both the human user and the robot try to be "as present as possible". The human user might be also tele-present there, but the robot is supposed to be physically present.

The CSA concept allow to divide the perception, cognition, planning and execution processes between the user and robot brains in a way that utilizes the strong features of both of them. Humans are usually good in cognition i.e. process their perception data in conceptual level and "understand" the environment. In the same way they are good in planning actions when only poor or partial data is available. Their perception is, however, not so good in many cases if compared to available machine perception. Robots can have a very accurate and sensitive perception, a good geometrical navigation system and they can repeat things untiringly in a same way. An optimal division of tasks in human-robot cooperation is to let the human to take care of the challenging (for the robot) cognition and planning tasks and to let the robot do perception and the actual physical tasks following so high autonomy as possible. Asking advice is a normal communication when human do cooperative work and it is also a useful tool in human – robot cooperation to avoid too complex control architectures in robots.

The CSA is not a uniquely defined virtual world, but rather a concept that can be realized in several ways depending on the application. The concept is illustrated in Fig. 2. The CSA describes those details of the working environments, which are necessary to communicate successfully tasks – in this case gardening tasks – to the robot. It includes both geometrical

and other information, which are able to be understood in the same way by human user and the robot. The fact that the information is spatially related is important, because all work tasks are spatial at least in some extent. In the gardening case the corresponding CSA could include a rough geometrical map of the garden, spatial information about planting, mobility restrictions, etc. Not all information needs to be fixed into a virtual model, but some can be also in the real environment as signs or tags readable to the robot. In such cases the CSA is not only a virtual world, but rather a combination of virtual and real worlds.

It is very important that the CSA can be constructed flexibly, quickly if needed and using different type initial data. The condition where the robot is to be used can vary from having almost zero apriori data (e.g. when the robot and user enter a new unmapped place) to data rich environment (e.g. user's well mapped home yard). In all cases the robot should be able to start working after a relative short initialization time, otherwise it is not considered as a useful tool. The underlying idea is to use very simple basic structure of CSA, which can be refined along the robot mission or if the robot is used repeatedly in the same environment. The case considered in the next chapters illustrates this idea.

Fig. 2. Common Spatial Awareness CSA describes those details of the working environments, which are necessary to communicate successfully tasks – in this case gardening tasks – to the robot.

4. Case WorkPartner

The research platform WorkPartner, shown in Fig. 3, is a humanoid service robot, which is designed for light outdoor tasks, like property maintenance, gardening or watching (Halme & all, 2003). The robot was designed as a multi-purpose service robot, which can carry on many different tasks. As the partner to the user it should be capable of performing tasks either alone or in cooperation with its master. In Fig. 3 WorkPartner is cleaning snow on a yard – a very common task in Finland in winter time. The colored beacon shown side of the robot is a part of the user interface equipment by the aid of which he/she can easily crop the area to be cleaned.

Skills for new tasks are initially taught interactively by the operator in the form of state diagrams, which include motion and perception control actions necessary to perform the

task. Design of the user interface has been done so that most of the interaction can be done (not ought to be done) with human like conversation by speech and gestures to minimize the wearable operator hardware. Different interface devices have also been developed to help the mutual understanding between robot and operator, especially in teaching and teleoperation situations.

Fig. 3. WorkPartner service robot cleaning snow from the yard

Although the WorkPartner robot is used here as a reference example, it should be noticed that many of the ideas and results presented are generic in nature and do not depend on the specific robot. Almost any mobile service robot with manipulation capability and with similar subsystem infrastructure could be used as the test robot as well.

4.1 Human-robot interface (HRI) equipment

The main functions of WorkPartner's HRI are

- communication with the robot in all modes of operation
- task supervision, assistance and collaboration
- task definition/teaching
- direct teleoperation
- environment understanding through common awareness
- information management in the home-base (Internet server)

The HRI consists of three main hardware components: operator hardware, robot hardware and home base. The home base component provides additional computing power and a connection to external databases (internet).

The core of the operator hardware is a portable PC including multimodal control interfaces, a map interface and a wireless connection to interface devices as illustrated in Fig. 4 - 5. The whole hardware is wearable and designed in such a way that the user can move easily in the same environment with the robot. The hardware is relatively versatile because it is designed not only for normal commanding and supervision, but also for teaching and teleoperation. Due to the nature of the work these functions might be needed without knowing it beforehand, so it is practical to make the whole system wearable at the same time. In certain cases, however, it is appropriate that the user can control the robot without wearing any operator hardware. Commanding by speech and gestures from close distances is used for this purpose.

The important interface components on board of the robot are camera, laser pointer, microphone, loudspeaker, head LEDs, and the arms. Besides for working, the arms can also

be used for communicating in a dialogue mode, like a human uses his/her hands. The communication network with the operator and the home-base server is based on WLAN.
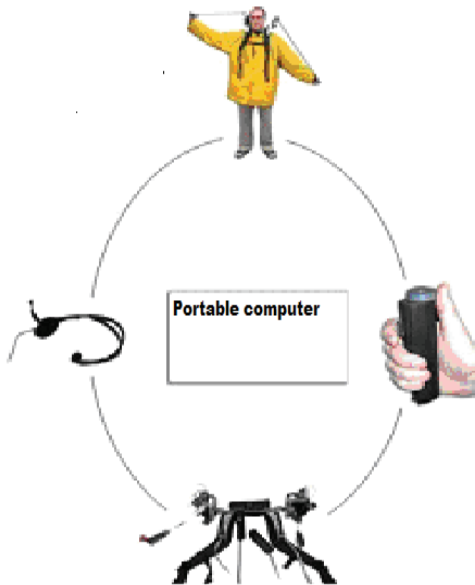
Fig. 4. User wearable hardware including "reins" (see also Fig. 9) and acceleration sensors for transmitting hand and body motions, and microphone connected to portable computer.

Fig. 5. Robot head includes camera, laser pointer and five LEDs.

4.2 Principle of interaction

The core of WorkPartner's interaction and cognition system is a software the main parts of which are the interpreter, planner, manager, and internal executable language (see Fig. 6). The *interpreter* takes care of the communication and receives the commands/information from the user. Information can be spoken commands, gestures, or data from any interface equipment. The data from interface devices and detected gestures are unambiguous and can be forwarded to the manager. Spoken commands are broken into primitives and the syntax of the command sentence is checked. If the syntax is accepted and all the other parameters of the command - such as the objects and their locations - are known, the command is transferred to the *manager*. In the event of shortcomings in the command, the interpreter starts asking questions from the user until the information needed to plan the mission is complete.

The *manager* forwards the interpreted command to the *planner*, which plans execution of the task as a set of subtasks The planner writes the plan automatically in the form of *internal executable language* (ILMR) (Kauppi, 2003), which controls the different subsystems of the robot during execution. ILMR is a XML type language acting as an intermediate link from the user, an intelligent planner, or a HRI to a robot's actions and behaviors. It provides ready features containing sequential and concurrent task execution and response to exceptions.
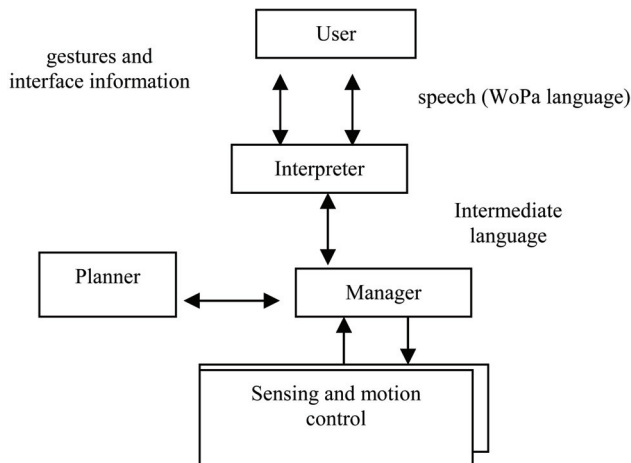
Fig. 6. Interaction principle of WorkPartner

The use of ILMR makes the software development much easier and has an important role when implementing learning capabilities for the robot. The following lines illustrates intermediate language commands

```
obsavoid(on)
speed(0.4)
createpath(myroute,1,1,5,1,5,10,8,13,15,20)
followpath(myroute)
sign1a=findtarget( camera,sign)
```

Due to the poor performance of the commercial speech processing software and the limited speech processing capabilities on board of the robot, the commanding language between

the user and the robot is formulated currently very strictly and the vocabulary is minimized. Language is based on commands starting with an imperative. For example **"Partner, bring box from hall"**. Command processing is executed interactively. The questions to the user are formulated so that they can be answered with one or a maximum of two words. "Partner" is the prefix that starts the command, "bring" is an action verb (go somewhere, take something and bring it back), "box" is an object and "hall" is a location attribute. The object "box" may be a unique object in the common presence or it may be one of the many boxes. In the latter case the robot asks more information. Anyway, the name indicates also form of the object, which is important for gripping process. "Hall" is a known location in common presence, but the location of "box" inside it may be not known. If not, it may be given by the operator by using a more specific definition (e.g. "near door") or the robot may start searching the hall to find the object.

To make actions like above possible a CSA model was developed supporting human task planning and the simple command language. The model is designed to be used in outdoor environment due to the robot design, but many of the principles are generic and applicable to other type of environments, too. The main ideas are explained and illustrated in the next chapters.

5. Spatial awareness in practice – constructing the CSA

In order to cooperate, the operator and robot must have similar understanding of the environment – at least of those elements of it, which are interesting in their mutual division of work. WorkPartner has a set of traditional robotic navigation tools (GPS, dead-reckoning and laser-based map matching), which it uses depending on the present situation. The pose (position and heading) in a fixed world coordinate system is known all the time. Therefore the spatial cognition of the robot can be related in its simplest way to a 2D map with fixed local coordinates and an object database that represents different properties of the environment or tasks to be performed in it. This combination of the map and objects can be visualized together as an occupancy grid to the robot and an object-oriented topographic type of map to the operator.

Making such a map is, however, not a straightforward matter, because connecting sematic information with the geometric one is not a simple task. Humans do not perceive the environment as numerical coordinates, but they can perceive and understand it well without this information. An essential feature in human shaping of the environment is entirety instead of details. Details are considered only after focusing attention to certain aspects motivated by a planned or on-going action. Coordinates are, however, natural for robots. A general problem is how to make both entities to understand the world with semantic information in a similar way. A classical approach to this problem would be to let the robot to recognize objects named by the user using a camera or other perception sensors, and put them on the common map. As well known, the difficulty of this approach is the automatic recognition of objects, which limits strongly utilization of such virtual world as the "common presence". What is meant by an object can be quite a general concept, not only a physical object, but also an abstract object illustrating future actions, like a hole to be drilled. On the other hand, if we allow human interaction the problem can be mostly overcame by letting the user recognize the objects and place them on the map by indicating them to the robot in a way it understands. There are many ways to do this. If the user has geometrically correct map available and he/she knows the position the object can be just be

placed on the map. The robot is not necessarily needed in this operation. Alternatively, when moving together with the robot he/she points the object in a way, which enables the robot to put it on the map. WorkPartner robot has two such pointing devices available, one is the laser pointer in its turning head and the other is "sceptre", which is a stick with colored head used by the user (see Figure 11 below). A third way is to relate a new object to an already known object, e.g. "close to object A", in which case the known object is used as a rough position reference.

It is important to keep the top level representation of CSA as simple as possible to allow human capability of shaping entirety work optimally. As mentioned before, this is done in a natural way by dropping out details until they are needed. One approach for this is a "box-world", where the objects with approximately known location are represented on the global map by "boxes" or " mini – worlds" inside which they exist. The boxes carry the names of the objects shown to the user so that he/she can outline the world both graphically and conceptually. The data base is object oriented so that details of the objects, their form, orientation, etc., can be obtained by clicking the corresponding boxes on the map. The principle is the same used in modern object based digital maps. Not all objects, however, are in boxes. Such objects exist in the database, but have no physical location (or the present location is not known). The objects may exist also without identity, say "ball", in which case it refers to all ball objects in the common presence before identifying more.

The underlying idea when using the CSA is to get an easy to human way to transfer the task related information between the user and the robot. In the "box-world" outlined above the essential information for the user is usually related to the boxes and their mutual relations to the world. For the robot the "box-world" is only the navigation world. Tasks usually require entering "inside boxes" to find detail information needed in execution. Robot perception is supposed to be able to find and recognize the object when it is close to the corresponding box. Task execution can then continue in an autonomous mode using robot's own perception or in cooperative mode, where the user helps using his/her perception. For the user the "box-world" can be represented in several ways graphically, augmented with a real picture from the environment or just as the real world, where positions are marked with signs, lines etc.

The methods to create a new CSA should be easy, quick, and reliable. Using existing may not be practical, because their validity may be a problem, and even if valid, fixing the local co-ordinate system in the right way and positioning the objects might take much time and effort. Using 3D- or 2D- laser scanners for mapping is a potential method, which is shortly described in the following.

Fig. 7 represents a laser range camera view from a parking place in the Helsinki University of Technology campus. Ranges are color coded and objects, like cars, building walls, etc. are easily recognizable for human cognition. In the next picture the user has cropped a car by mouse forming a box around it. He/she can immediate transfer this information to the presence model by giving a name (like "my car") for the box. If the object "my car" is needed during a mission of the robot – e.g. when commanding to wash it - the robot knows that it can be found inside that box in the presence model (provided of course that it has not been moved). The box may include the accurate model of the car as illustrated in the last picture in Fig. 7. Such information has usually no use for the user when commanding the robot, but the robot may need it when doing its job (e.g. washing the car).
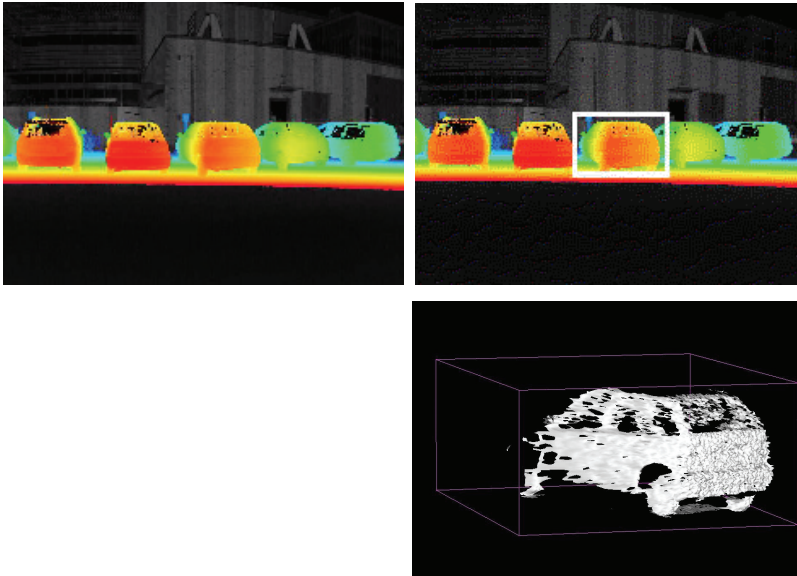
Fig. 7. Building up CSA "box-world" using 3D-range laser camera.

This is a process, where human cognition is used very effectively to create the CSA in a semi-automated way rapidly, reliably and a natural way including only the essential object information. After creation, only a simplified representation of the common presence is usually enough for operational purposes in the HRI.

Mapping of the basic geometry of the CSA can be done by many ways and means. Another possibility is illustrated in Fig. 8. It is a wearable SLAM system, which is based on a personal navigation system and 2D – laser scanner. The personal navigation system (PeNa), developed originally in a European Union PeLoTe project (Saarinen & all, 2004) for use in rescue operations, uses only dead-reckoning instruments, like a stepping odometer and heading gyro, because it is designed to operate without support from beacon systems. The stepping odometer is, however, fused with the laser-based odometer obtained by algorithmic processing of laser range data.
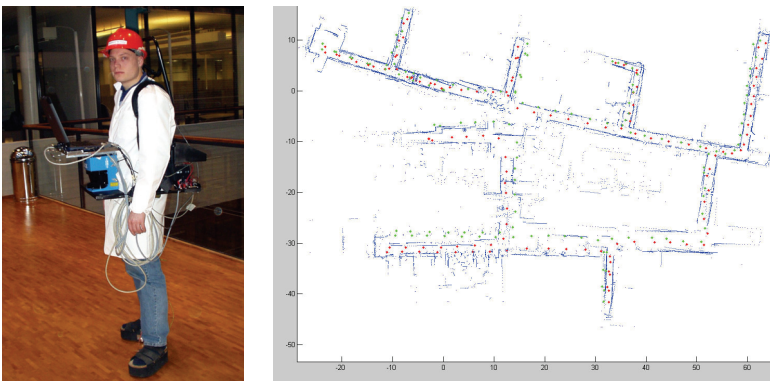
Fig. 8. Personal SLAM system developed in EU PeLoTe project.

Fig. 8 illustrates also the result when mapping of an office corridor environment. The map made by the aid of PeNa is quite correct in proportioning, but the long corridors are slightly bending. The bending effect is due to the dead-reckoning navigation error, mainly caused by gyro drifting and odometric errors. When used as the basic map of a common virtual presence such distortion has no meaning, because a human looks more at the topology of the map when using it and the robot relies on its sensors when moving and working in the environment. Mediating the semantic information between the entities is possible in spite of geometric errors as long as the human entity can understand the main features of the map and their correspondence in the real world.

7. Communication trough the physical environment - devices and means for interaction

When working with a cooperative robot, like WorkPartner, the human user needs effective means to communicate with the robot. When using a CSA model communication utilizes the concepts of this model. From the user point of view the CSA represents the world of robot understanding. From robot's point of view the user is only a special type of object that interacts with it. Interaction can be done directly or indirectly. The direct interaction means that the information goes directly to the awareness of the robot. In the indirect transfer the user leaves some kind of mark and related information to the real environment, where the robot can obtain it when needed. The idea of the common spatial awareness is best utilized when the indirect methods are used as much as possible, because the user uses then maximally his cognition and ability to perceive entireties. The user has then already been able to complete the geometric information with task related conceptual information and the autonomy of the robot actions can be probably increased during the work period. To illustrate this, some of the interaction means developed for WorkPartner robot are explained.

7.1 Direct teleoperation
Direct teleoperation is needed to test and move the robot when an intelligent part of the software cannot take over for one reason or another. This is the primary direct means to interact. In the case of the WorkPartner robot direct teleoperation is used e.g. to teach skilled tasks before they can be done autonomously. Another situation where direct teleoperation is used is when driving the robot from one place to another or testing its functionality. Because of the large number of degree of freedoms the robot has, a joystick alone is not a practical device for teleoperation. A wearable shoulder mounted device, called "Torso controller" was developed (see Fig. 9).

7.2 Gestures
Gestures (and expressions) are very typical and natural way of human communication. They are used both with the speech and without it (Amai et all, 2001; Fong et all, 2000; Heinzmann and Zelinsky, 2001). It is fairly easy to develop a sign language which is simple but rich enough in meaning for interacting with a robot. The problem, as in the case of speech, is reliable recognition of gestures used in the language. In the case of the WorkPartner robot, two different ways to detect gestures have been developed and tested.

One way is to use the camera head of the robot to track the operator, who uses a colored jacket as illustrated in Fig. 10. A colored jacket is used for two purposes, first to mark the user and secondly to facilitate the gesture recognition. The gestures are recognized by a feature extraction algorithm, which first extracts the jacket color from the picture. This method works fairly well within short distances and in moderate illumination conditions. Another method is to use the torso controller explained above. Hand gestures can be recognized on the bases of the wrist positions. This method is not limited by the distance to the robot, but the controller is needed. Using camera based recognition allows communication with the robot without any wearable equipment.

Fig. 9. Torso controller

Fig. 10. Use of gestures in commanding. Gestures are recognized by the robot's camera head, which tracks the colored jacket of the user.

Rich enough sign language can be constructed in most cases by the aid of simple static gestures, but by adding dynamic features to the gestures the language can be made more natural to use. Dynamic features are included in most sign languages used in human to human communication.

7.3 Pointing interfaces

Pointing is an important part of human communication. The purpose is to relate certain special objects with semantic or symbolic information or to give for an object a spatial meaning. Humans naturally use their hands for pointing but also technical means like pointers when the "line of the hand" is not accurate enough. In the case of human to robot communication there are several ways pointing can be realized. Pointing can be done through the virtual CSA by using a normal computer interface provided the accuracy obtained is good enough and spatial association with the object can be done easily (like in Fig. 7).

When pointing in the real environment a pointing device may be used. The main problem with handheld pointing devices, like laser pointers, is how to bring the pointed location to the CSA geometry, usually defined in a local coordinate system. In the case of the WorkPartner robot this problem has been solved by using the robot itself as the reference point. The navigation system of the robot knows all the time the robot pose, i.e. 2-D position and heading angle, in the local coordinate system. Two different systems are in use. In one of them the user uses the laser range finder assembled in the other eye of the camera head on same optical axes as the camera. The user points through the camera image by teleoperating the camera head. The pointed locations can be transferred to the robot base coordinate system immediately and then to the common presence coordinate system if needed. Objects up to 10-15 m distances can be pointed.
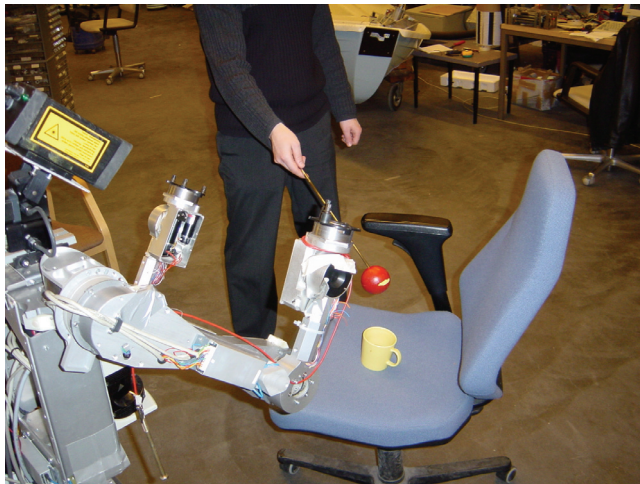
Fig. 11. Use of "scepter" for pointing.

The other system is called "scepter". As illustrated in Fig. 11, the scepter is a stick with a colored ball at the tip. The visual perception with color tracking follows the tip and

measures the coordinates from the image when the operator indicates that the position is right. This system is applicable within close distances only, but it does not require any active devices be with the operator. In the picture the user directs the robots sight to the coffee cup when e.g. naming it as an object in CSA. The same principle is applied when the user uses himself as the pointer. By tracking the colored jacket of the user the robot can measure his/her position and also follow him/her when moving. For example by letting the robot record the trajectory of the motion the user can show the way to travel in a work task at the same time as he/she evaluates if the road is passable.

7.4 Using signs and marks in the real environment

One very old mean of indirect communication between humans is by signs or marks left to the environment. The same principle is also applicable in human to robot communication or even robot to robot communication (Kurabayashi et all, 2001). The signs can be passive but also active, so that they indicate their presence and information actively. Good examples of passive signs are traffic signs. Similar signs can also be used to conduct robot tasks in the working area, as illustrated in Fig. 12. The figure presents a hypothetical case, where a user has marked the home yard for a robot that helps in cleaning and carrying things in this environment. By simple color-coded signs it is easy to mark routes to travel, areas forbidden to cross, dangerous areas (like ditches), areas to collect litter, etc.

Fig. 12. Illustration how task conducting signs could be used in a home yard

One may of course mark them through the CSA model provided such one is available and accurate enough, but in many cases it is easier to just mark this information in the real environment and allow the robot to read it when close enough.

8. Conclusions and discussion

Communication by the aid of symbolic or semantic information between the user and the robot is essential in effective use of future collaborative service robots. This is possibly the only way to avoid the robots becoming masters for th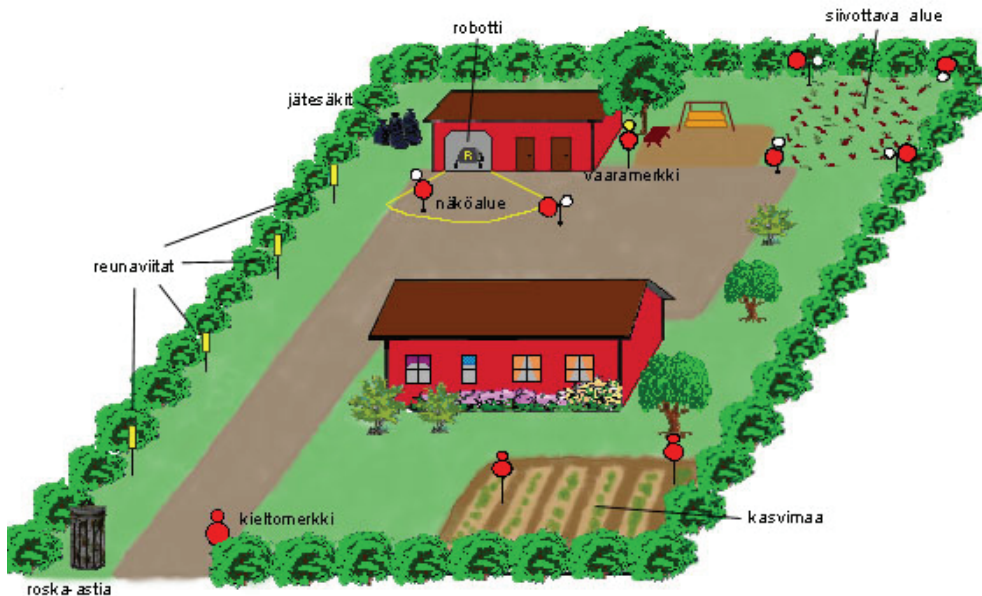e users as long as the autonomy of the robots can be developed highly enough. Really skilled tasks reaching a sufficient level of autonomy is still far away. This way of building up a new type of HRI technology allows the user the possibility of using his/her superior cognition capacity to load the machine instead of it loads him/her.

The underlying idea in the presentation in this chapter is that this can be realized through "common spatial awareness", CSA – a concept of utilizing robot cognition together with human cognition. SCA is a virtual world presentation, which is understood in a similar way by all entities, robots and users, involved in the task. Semantic information related to the task and the real environment can be exchanged trough this world. The essential problems are how to build this world effectively, represent it for the user, and how to associate objects and concepts with it when the real world and /or tasks are changing.

In the presentation the idea behind this has been studied and demonstrated by the aid of the WorkPartner service robot developed at TKK Automation Technology Laboratory. The humanoid-type robot, having a simple command language, and the wearable user interface equipment allow the user to work with the robot in the same outdoor environment. Spatially bound information is essential in cooperative tasks. Various interactive methods and equipment have been developed and demonstrated to bring this information as a functional part of common presence.

9. References

Amai W., and Fahrenholtz J., Leger C. (2001). "Hands-Free Operation of a Small Mobile Robot", Autonomous Robots, Vol. 11, No. 1, July 2001

Fong T., Thorpe C., "Vehicle teleoperation interfaces", Autonomous Robots, Vol. 11, No. 1, July 2001

Fong T., Conti F., Grange S. and Baur C (2000). "Novel Interfaces for Remote Driving: Gesture, Haptic and PDA", SPIE Telemanipulator and Telepresence Technologies VII, Boston, MA, November 2000

Halme A., Sievilä J., Kauppi I., Ylönen S., "Performing skilled work with an interactively operated service robot", In S.Yuta et all (Eds), Field and Service Robotics – Recent Advances in Research and Applications, Springer, 2006 .

Halme, Aarne; Leppänen, Ilkka; Suomela, Jussi; Ylönen, Sami; Kettunen, Ilkka, "WorkPartner: Interactive Human-like Service Robot for Outdoor Applications", The International Journal of Robotics Research , 2003. Vol. 22, nro 7-8, pp. 627-640.

Heinzmann J., Zelinsky A., "Visual human-robot interaction", 3rd conference on Field and Service Robotics (FSR2001), June 11-13, 2001, Helsinki, Finland

Kauppi I., "Intermediate language for mobile robots – A link between the high – level planner and low-level services in robot", Doctoral thesis, VTT Publications 510, Espoo, Finland, 2003

Kurabayashi, D., K. Konishi, H. Asama: "Distributed Guidance Knowledge Management by Intelligent Data Carriers", Int. J. of Robotics and Automation, vol.16, no. 4, pp. 207-216, 2001

Saarinen, Jari; Suomela, Jussi; Heikkilä, Seppo; Elomaa, Mikko; Halme, Aarne: "Personal navigation system". The IEEE/RSJ International Conference on Intelligent Robots and Systems, Sendai, Japan 2004. pp. 212-217.

Tachi, S.: Augmented Telexistence, Mixed Reality, pp.251-260, Published by Springer-Verlag, 1999.

4

The Context-Awareness for the Network-based Intelligent Robot Services

Chung-Seong Hong, Kang-Woo Lee, Hyoung-Sun Kim and Hyun Kim
u-Robot Server Research Team, Electronics and Telecommunications Research Institute
Republic of Korea

1. Introduction

Recently, while the existing industrial robot market is saturated, the research about the service robot is actively progressed. In order that the service robot comes into our daily lives like the electric home appliances or mobile devices, it has to provide the intelligent services to a user.

Generally, the robot has three functional components of sensing, processing and acting. The ability of these components depends on its own hardware performance. Ubiquitous Robotic Companion (hereafter URC) is the new concept of the network-based service robot (Kim et al., 2005). In the concept of URC, a robot expands the ability of these components through the network. The robot can use not only its internal sensors but also external sensors which are embedded in the environment for sensing. A number of robots can share a high performance server to increase the processing power. Also, it can use the external actuators which are deployed in the environment, not only in its body, for acting.

These improvements can bring about the enhancement of context awareness and provide proactive services to the users. When a user requests a service, a robot is able to provide appropriate services by recognizing the user's current situations. Thus, to realize the URC, it is important to develop a context-aware system which can help robots become aware of user's situations.

The context-aware system is comprised of the various kinds of distributed computing entities. Early context-aware systems had the relatively simple structure because of being only comprised of distributed elements which communicate with the local or the remote sensors. But, as the structure of the context-aware system become complicated and various applications appear, the context-aware infrastructure has been needed in order to enhance the maintenance and the reuse of the context-aware application. It can help to remove the complexity of the application development. Therefore, the development of the context-aware framework which can provide reusable components with high-level abstraction is needed.

The aim of this chapter is to introduce the context-awareness for a network-based service robot. For this, we define the functional requirements and implementation issues of the context-aware framework and propose the layered conceptual structure. And then, we describe how proposed conceptual structure was implemented to the CAMUS (Context-Aware Middleware for URC System) which is a context-aware framework for the network-based service robots. Finally, we introduce various kinds of robot services which were developed by using the CAMUS.

2. Related works

As the context-aware computing gets more and more interests, attempts applying it to diverse kinds of research fields have been tried. However, the development of the context-aware intelligent application in the robot field has rarely been carried out. Thus, in this section we review the previous research works in the field of context-aware computing.

Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and application themselves. A system is context-aware if it uses context to provide relevant information and/or services to the user, relevancy depends on the user's task (Dey, 2000).

The context-aware system can be implemented in many ways. The approach depends on special requirements and conditions such as the location of sensors, the amount of possible users, the available resources of the used devices or the facility of a further extension of the system (Baldauf et al., 2007). A number of research groups have developed context-aware systems and applications from 1990s. However, most early researches were remained as the experimental level on developing the prototype, because various context information and its achievement technologies were considered only for specific applications at specific platforms.

Recent context-aware systems use a middleware or context-aware server architecture for separating an acquisition and use of the context. These architectures have a layered structure which has an expandability and reusability.

The architecture of the Context Managing Framework (Korpipaa et al., 2003) uses four main functional entities which comprise the context framework: the context manager, the resource servers, the context recognition services and the application. Whereas the resource servers and the context recognition services are distributed components, the context manager represents a centralized server managing a blackboard. It stores context data and provides this information to the client applications.

Another framework based on a layered architecture was built in the Hydrogen project (Hofer et al., 2002). Its context acquisition approach is specialized for mobile devices. While the availability of a centralized component is essential in the majority of existent distributed content-aware systems, the Hydrogen system tries to avoid this dependency. It distinguishes between a remote and a local context. The remote context is information another device knows about, the local context is knowledge our own device is aware of. When the devices are in physical proximity they are able to exchange these contexts in a peer-to-peer manner via WLAN, Bluetooth etc. This exchange of context information among client devices is called context sharing.

Context Toolkit (Dey et al., 2001) was developed as an intermediate dealing with context between a sensor and an application service. It collects context information from sensors, analyzes it, and delivers it to application services. The toolkits object-oriented API provides a super-class called *BaseObject* which offers generic communication abilities to ease the creation of own components.

CASS (Context-Awareness Sub-Structure) (Fahy & Clarke, 2004) proposed extensible centralized middleware approach designed for context-aware mobile applications. The middleware contains an *Interpreter*, *ContextRetriever*, *Rule Engine* and *SensorListener*. The *SensorListener* listens for updates from sensors which are located on distributed sensor nodes. Then the gathered information is stored in the database by the *Interpreter*. The

ContextRetriever is responsible for retrieving stored context data. Both of these classes may use the services of an *Interpreter*.

RSCM (Reconfigurable Context-Sensitive Middleware for Pervasive Computing) (Yau et al., 2002) separates sensors and application services, and delivers necessary context information to applications through ad-hoc network. RSCM also collects context information, analyzes, and interpret it. When this context information satisfies conditions of registered application services, RSCM delivers it to the application services. It is the client-side to make a decision whether its application services should be executed in it or not

The SOCAM (Service-oriented Context-Aware Middleware) project (Gu et al., 2004; Wang et al., 2004) introduced architecture for the building and the rapid prototyping of context-aware mobile services. It uses a central server called *context interpreter*, which gains context data through distributed *context providers* and offers it in mostly processed form to the clients. The context-aware mobile services are located on top of the architecture, thus, they make use of the different levels of context and adapt their behaviour according to the current context. Also, it proposed a formal context model based on ontology using OWL to address issues about semantic representation, context reasoning, context classification and dependency.

GAIA project (Biegel & Cahill, 2004) also developed the middleware which can obtain and infer different kinds of context information from environments. It aims at supporting the development and execution of portable applications for active spaces. In GAIA, context is represented by 4-ary predicates written in DAML+OIL. The middleware provides the functions to infer high level context information from low level sensor information, and expect the context in advance by using the historical context information.

CoBrA (Context Broker Architecture) (Chen, 2004; Chen et al., 2003) is an agent-based architecture for supporting context-aware computing in intelligent spaces. Intelligent spaces are physical spaces that are populated with intelligent systems that provide pervasive computing services to users. In the centre of CoBrA, there is an intelligent *context broker* that maintains and manages a shared contextual model on the behalf of a community of agents. These agents can be applications hosted by mobile devices that a user carries or wears (e.g., cell phones, PDAs and headphones), services that are provided by devices in a room (e.g., projector service, light controller and room temperature controller) and web services that provide a web presence for people, places and things in the physical world. The *context broker* consists of four functional main components: the *Context Knowledge Base*, *the Context Inference Engine*, the *Context Acquisition Module* and the *Privacy Management Module*.

Though many attempts for developing a context-aware system has been carried out, previous context-aware systems cannot provide a complete solution for all the essential requirements in context-aware computing. Moreover, they do not consider the robot as a computing entity which can interact with and provide services to a user. In this chapter, we describe how we developed the context-aware framework for a network-based intelligent robot.

3. Requirements and issues of context-aware framework

In this section, we define the functional requirements and implementation issues which have to be considered when we develop the context-aware framework based on the previous related researches.

3.1 The functional requirements

The functional requirements should be considered when we design the structure of the context-aware framework. The necessary functions are as follows:

1. Accessing and controlling devices
    The context-aware application can access and control various computing devices which are located in the environment of a user and robot. For this, the context-aware framework has to provide the methods of accessing and controlling these devices. With this function, context-aware application can acquire contexts and provide services with various devices.
2. Processing context
    The context-aware framework has to provide the element which has the responsibility of processing context obtained from various sensors for a context-aware application to use. The functions of this element include time stamping, interpreting, and filtering context.
3. Managing context
    The processed context has to be stored and managed with the unified context model by the context-aware framework. The managed context with the context model can be searched, accessed, and modified by the application when needed.
4. High-level context abstraction
    The context which is obtained from a sensor and managed with the context model can be processed for the high-level abstraction to provide the intelligent service by a context-aware application. This high-level abstraction should be supported by not only the context-aware application, but also context-aware framework.
5. Execution and management of context-aware application
    The context-aware framework can manage the life-cycle of the context-aware applications in order to implement the various applications easily. Moreover, the context-aware framework has to provide the functions of starting, proceeding, and termination a context-aware application.
6. Communications
    Finally, the context-aware framework has to provide the communication methods so that various context-aware elements which are located in the different physical space can be comprised of one context-aware system and they can communicate and deliver the necessary information with each other.

3.2 The implementation issues

The implementation issues are about what should be considered when we implement the context-aware framework. The selected implementation issues are as follows:

1. Support for heterogeneity
    When we develop a context-aware application using the context-aware framework, the computing entities are very various and have different resources, for example, from the resource poor sensor run on the Embedded Linux to the high performance server run on the Windows OS. The context framework has to support the heterogeneity of the OS, programming languages, and so on.
2. Support for mobility
    In the context-aware environment, a user can use not only static devices, but also mobile devices. The context-aware framework provides methods for connecting and controlling these devices dynamically.

3. Support for scalability
   The context-aware application requests and delivers the necessary information with many sensors, the actuators, and various computing entities. According to the development of multimedia, this information could be a large-scaled audio or video streaming. Therefore, the context-aware framework can resolve the scalability problem about reducing the load of the network.

4. Support for privacy protection
   Privacy is about the control of information. In a context-aware computing environment, users worry about their privacy because hidden sensors are embedded in their environment, and the information acquired by these sensors could be often shared by different applications. Users desire privacy protection because they are concerned about the possible misuse of their information. The context-aware framework provides methods for users' privacy protection.

5. Separation of concerns
   The context-aware framework provides the method for using the context which is obtained from a sensor to many context-aware applications. Thus, the acquisition of a context is separated from the use at an application for reusing context.

6. Exceptional Handling
   The context-aware framework is able to make the exception handling like the access failure or disconnection with the computing entity. Particularly, because a robot uses a wireless network in common, the problem about the disconnection and recovery has to be resolved.

4. The layered conceptual structure

The previous context-aware framework has the mutually different hierarchy structure and scope of functionality. In this section, we design the layered conceptual structure of the context-aware framework based on the defined functional requirements in section 3. The modules and operations which are located in each layer are shown in Fig. 1.

4.1 Context source and service layer

In the bottom of the layered conceptual structure, there is the *Context Source and Service Layer* which is consisted of smart devices. In this layer, there exist various kinds of sensors and actuators including robots which are special kinds of devices.

Context sources include not only hardware devices which provide explicit information given by the user or environment, but also computing sources which can be used to collect computing context from information database or web-services. Depending on the type of context, one or multiple sensors may be employed.

4.2 Device access and control layer

In the *Device Access and Control Layer*, there are the Sensor Framework Module and Service Framework Module. The Sensor Framework Module concerns how to acquire the context information from various information sources. Raw context acquisition operation captures the raw data as the value of context features. The Service Framework Module concerns how to provide services and information through actuators. Fig. 2 shows an example of context acquisition from a RFID reader.
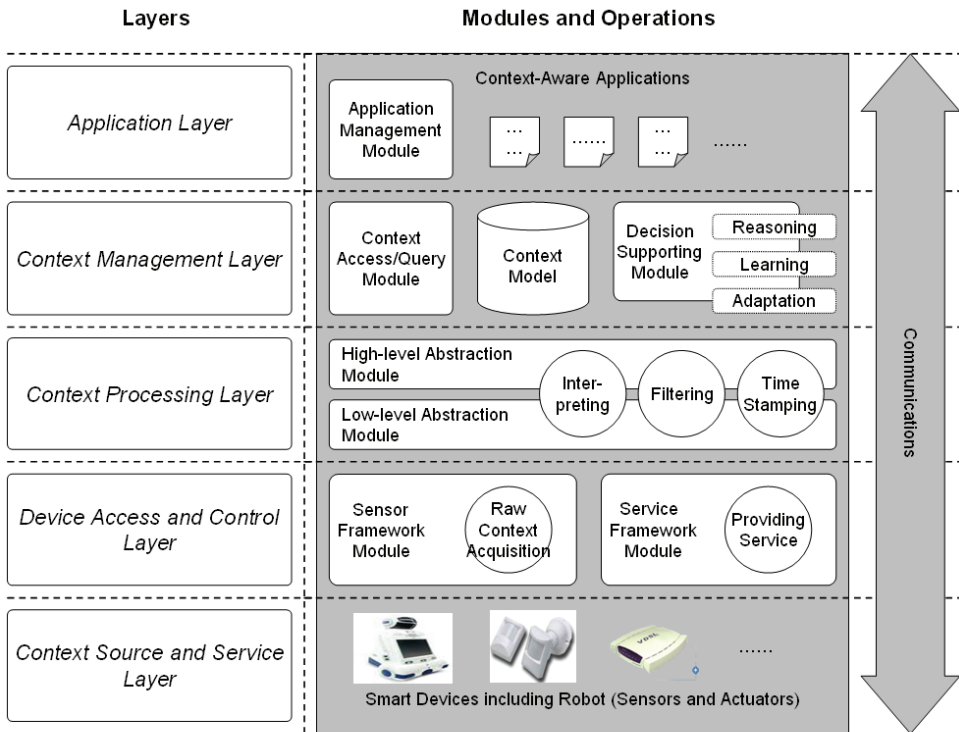
Fig. 1. The layered conceptual structure for the context-aware framework

4.3 Context processing layer

The *Context Processing Layer* concerns how to process the acquired context before storing or delivering it. The context obtained from the *Device Access and Control Layer* may include technical data which is not appropriate to use in the context-aware application directly or include unnecessary information. Usually, context processing operations are performed by the need of the context-aware application. However, the operation which is repeatedly used in many applications has to be separated from the applications and defined in the context-aware framework. After that, an application developer can use the context which is processed by the pre-defined operations without implementing them in the application. Context processing can be divided into following three operations:

1. Interpreting operation
    The Interpreting operation interprets and abstracts context so that it is available to be used in the context-aware application. For example, the current location of a user can be interpreted using the RFID tag information when a user enters the detecting range of a RFID reader which is installed in a specific location as shown in Fig. 3.
2. Time stamping operation
    Time is very important context to all applications. A context is changed while the time passes away. This is due to the fact that context information is dynamic by nature. Time

stamping operation adds time information to the context. By doing this, we can use the information of context history. Fig. 4. shows an example of time stamping operation.

3. Filtering operation

    In the contexts acquired from sensors, there can be unnecessary information. When all contexts are delivered to the components in the context-aware framework, the communication cost will rapidly increase. Moreover, it may cause the system to be down. The filtering operation reduces the amount of data by removing the context which it is not needed or duplicated. Fig. 5. shows an example of filtering operation.

The *Context Processing Layer* is comprised of the Low-level Abstraction Module and High-level Abstraction Module as shown in Fig. 1. The difference of Low-level and High-level Abstraction Modules is according to the referring to the other context which is stored using the Context Model in the *Context Management Layer*.
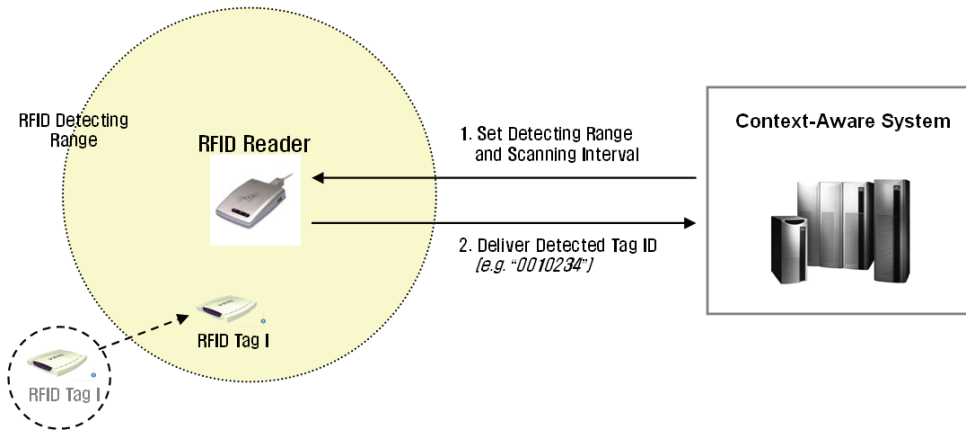
Fig. 2. An example of context acquisition from a RFID reader
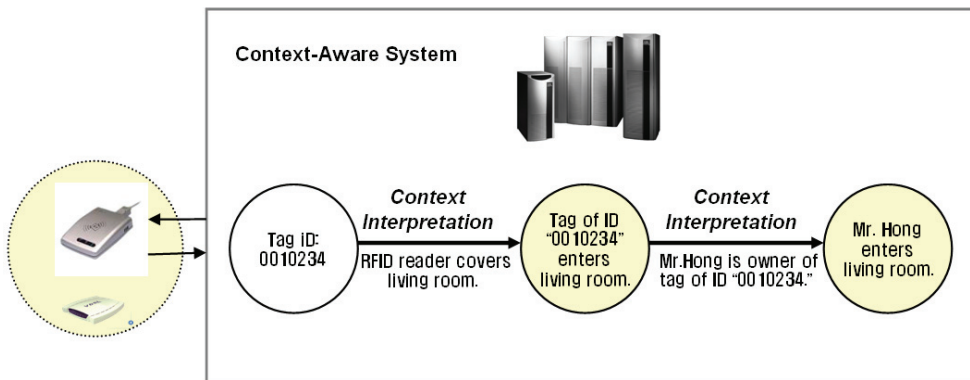
Fig. 3. An example of interpreting operation
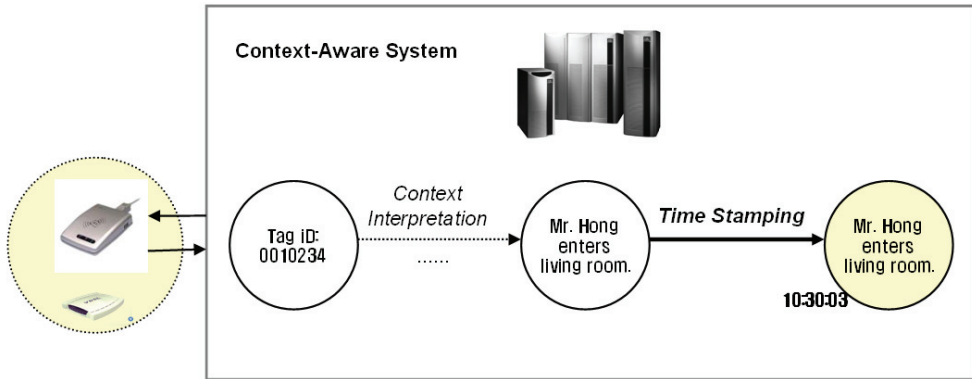
Fig. 4. An example of time stamping operation
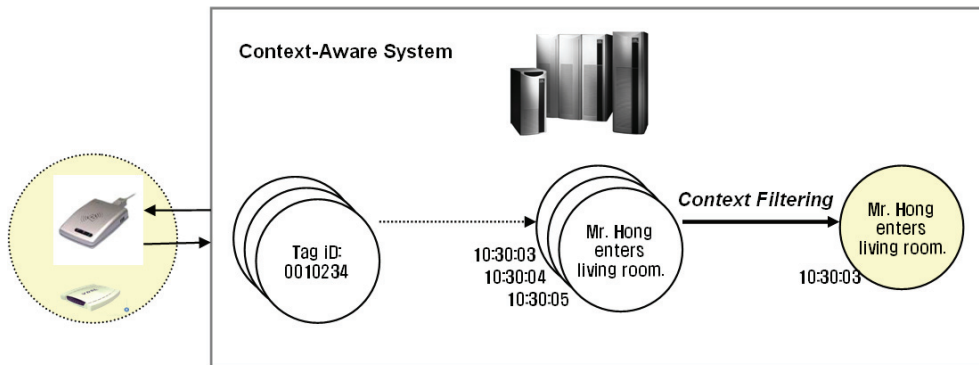
Fig. 5. An example of filtering operation

4.4 Context management layer

The *Context Management Layer* stores and manages the obtained context in order to use lately. The context is stored by using the context model. The *Context Management Layer* provides the Context Access/Query Module for accessing and modifying the context.

In addition, The *Context Management Layer* can provide the Decision Supporting Modules. The Decision Supporting Module includes the supporting tools or engine which can resolve the complex decision problems such as reasoning, dynamic adaptation, learning, and so on for general use.

4.5 Application layer

In the *Application Layer*, there are context-aware applications and the Application Management Module. The Application Management Module manages the life-cycle of an

application. The Application Management Module not only creates and finalizes an instance of application, but also manages the state of it. Moreover, the Application Management Module can provide tools for supporting event handling and rule firing. A context-aware application gets contexts from the Context Model through the Context Access/Query Interface Module or from the sensors directly through the remote-procedure call which is provided by the Sensor Framework Module.

5. The context-aware middleware for URC systems

We developed CAMUS as the context-aware server framework for a network-based robotic system (Kim et al., 2005; Hong et al, 2006). The functional architecture of CAMUS is shown in Fig. 6.

The *service agent* is a software module performed as a proxy to connect various external sensors and actuators to CAMUS. It delivers information of the sensors in environment to the CAMUS *main server*. Also, it receives control commands from CAMUS *main server* and controls devices in the environment, and conducts applications.

The *service agent* is executed in the *service agent manager*. The *service agent manger* makes accesses to adequate *service agent*s out of the currently running *service agent*s using a searching function. The *service agent manager* supports the prerequisite for the proactive context-aware services. Also, it searches, invokes, and executes necessary services for each of CAMUS *task*s. Information obtained by *service agent*s is processed by the *sensor interpreter* as the *low-level abstraction module*. This processed information will be delivered as a type of event to the CAMUS *main server* through the *event system*. Creating events indicates the changes of context in the user's or robots' situation.

The *event system* generates and manages events from physically distributed environments and is responsible for exchanging messages among CAMUS components. Above all, it delivers the events to the *context manager* and *task manage*r so that CAMUS *task*s should be able to recognize the changes of situation and further update the existing *context model*.

The *context manager* infers implicit knowledge based on the context information which is delivered through events, and it also manages this inferred knowledge. When CAMUS tasks are executed, they refer to the context knowledge managed by the *context manager*. Therefore, the *context manager* should be able to provide several main functions for utilizing the context knowledge; (a) the context modelling and the knowledge representation, (b) the management of the context data, (c) the query for the context knowledge, and (d) the inference of the implicit knowledge, and so on.

The *context model* which is represented and managed in CAMUS includes the user context, environment context, and computing device context. The user context includes user profile, user's task information, user preference, and so on. The environment context includes hierarchical location information, time, and so on. The computing device context includes information about available sensors and actuators including robots.

The *task* can be regarded as a set of work items which are required to be taken in a specific context or by the user's command. Each work item, which is a unit of action, is described by the task rules. A task rule is described by the convention of Event-Condition-Action (hereafter ECA). The action part arranges the operations of the *service agent* to be executed

when the incoming event meets conditions. The *service agent* is accessed by Task through the *service agent manager*. The ECA rule is managed by the *ECA rule engine*.

The *task manager* initiates individual tasks and manages on-going task processes. It also controls and coordinates tasks by managing the current state information of each *task*. The state of the *task* is managed by the *Finite State Machine*.

Under this functional architecture, CAMUS is mainly divided as a CAMUS *main server* and *service agent manager*s based on the embedded types in the physical environment as shown in Fig. 7. The *service agent manager* which is deployed in the local server in the home or office environment is called the *local station*. The *service agent manager* which is deployed in the robot is called the *robot station*. And, the *service agent manager* which is deployed in the mobile device is called the *sobot station*. Each type of the *service agent manager* has the different context processing ability, user interface, and program module. The CAMUS *main server* communicates with *service agent manager*s through the communication framework called *PLANET*.
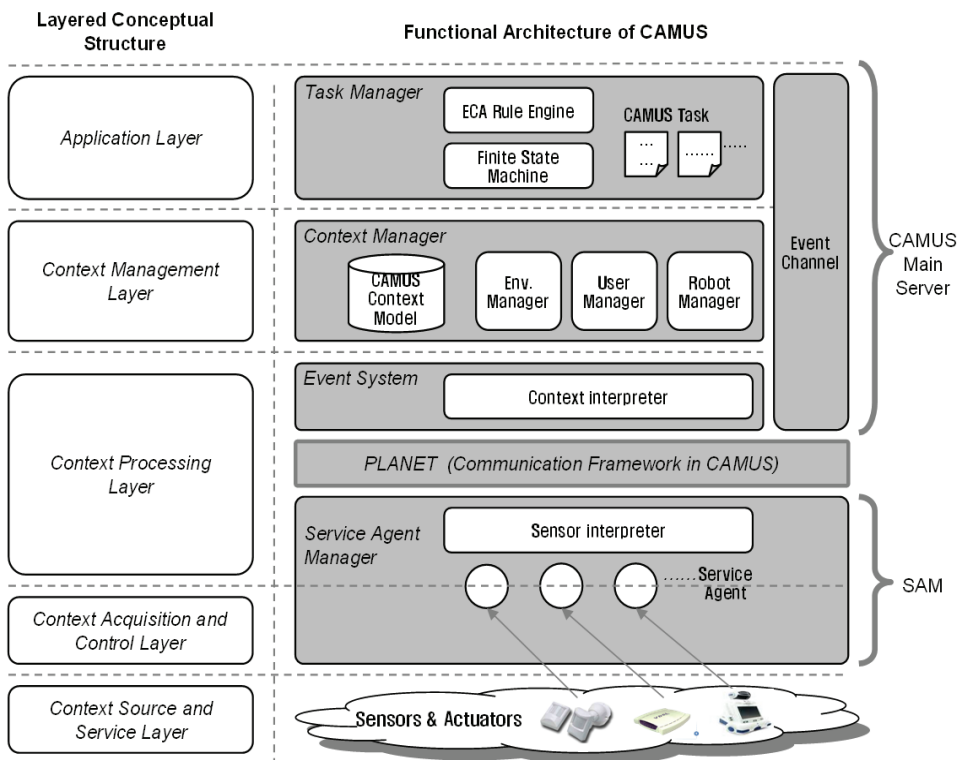
Fig. 6. The functional architecture of CAMUS
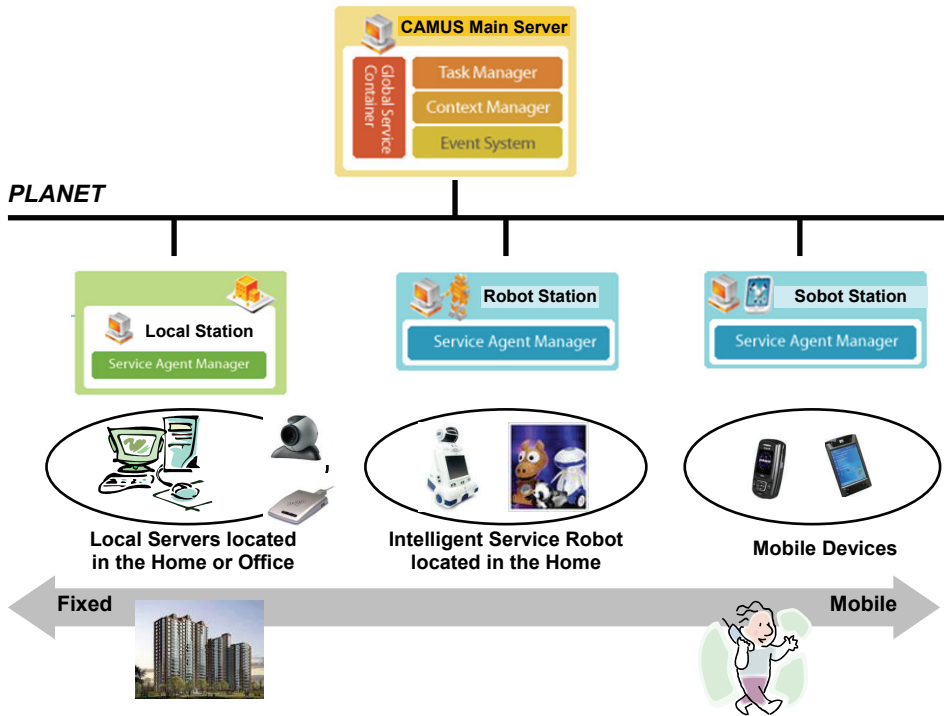
Fig. 7. System deployment in the physical environment

6. Intelligent robot services

In this section, we introduce the URC field test and several applications which are developed by using the CAMUS. Applications are the robot greeting, home monitoring, and follow-me services.

6.1 The URC field test

The first step result of developing CAMUS was validated through URC field test during two months from Oct. 2005. to Dec. 2005 (Hong et al., 2006). For URC field test, 64 households were selected from Seoul and its vicinity, where the broadband convergence network (BcN) was installed and 3 kinds of robots (Jupiter, Nettro and Roboid) were offered to them. Fig. 8 shows the URC field test structure for practical services of the CAMUS. Throughout the URC field test, we verified that the Hardware structure of a robot can be simplified by distributing robot's main internal functionalities (ex. speech recognition, and image recognition) to external servers through the network. And we also found that it is enabled to more effectively provide various kinds of services which are necessary for daily lives under the URC concept.

35 households out of 47 households' respondents showed a high degree of satisfaction, and they felt home monitoring, security, and autonomous cleaning services are most appropriate services for their daily lives. However, they felt a robot is worth only about 500 ~ 1,000

dollars with its performance, which implies that the robot market targeting households needs to make an effort to stabilize the robot price reasonably.
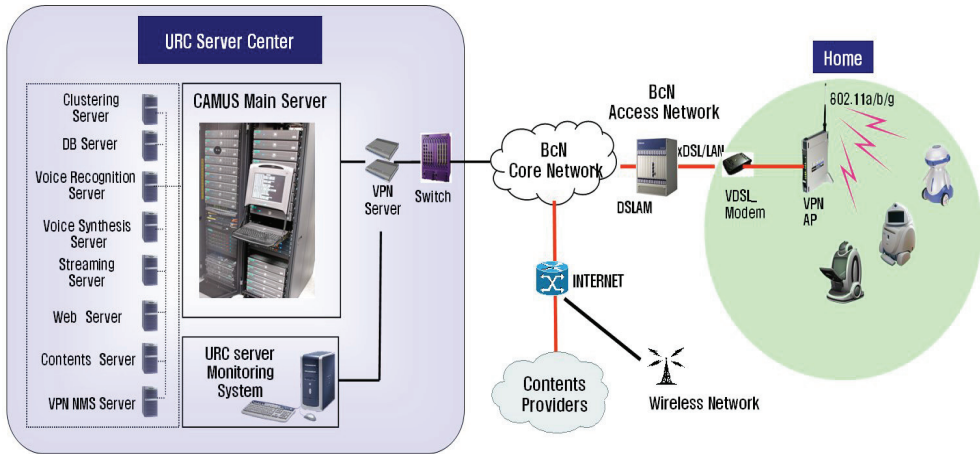
Fig. 8. System structure of the URC field test (Hong et al. 2006)

6.2 The robot greeting service

This is the robot service which was developed for the URC field test (Hong et al., 2006). The used robot called Jupiter was made by the YUJIN ROBOT, Co. and the application was developed by using the CAMUS.

In the scenario of the robot greeting service, a user arrives at home and goes to the living room. The RFID reader deployed in the living room recognizes the user's entrance from outside of the home. The RFID reader delivers the entrance information through the event in the CAMUS. CAMUS *task* orders the robot to move to the living room. The robot delivers an event about navigation completed to the CAMUS *task* through the event channel after moving. The robot says the greeting message using the Text-To-Speech service. The robot shows a message which is left from the user's family. The CAMUS task orders the robot to move to the standby-position and wait for user's command. The snapshots of this service are shown in Fig. 9.

Many kinds of sensors and actuators were used for this service. As a sensor, the RFID reader was used to recognize the location of a user and the microphone which was embedded in the robot were used for the Automatic-Speech-Recognition service. The speaker used by the Text-To-Speech service, robot navigation service, and the LCD monitor were used as the actuators.

6.3 The home monitoring service with the mobile phone

We think that the home monitoring service is one of killer applications in the service robot field. The robot navigation service and video streaming technology enable this service.

In the scenario of the home monitoring service, a robot watches the inside of home. When a thief makes an appearance during the robot's home monitoring service, the robot detects a movement of the thief and records it. A warning message is sent to a user from the robot,

and a user can watches the recorded video streaming by using the mobile phone. The snapshots of the home monitoring service are shown in Fig. 10.

(a) a user arrives at the home

(b) a robot moves to the living room for the robot greeting service

(c) a robot shows a message which is left from the family

(d) Message delivery user interface

Fig. 9. Snapshots of the robot greeting service (Hong et al., 2006)

(a) a robot watches the home

(b) a thief is appeared and a robot detects the movement of the thief

(c) a warning message is sent to a user

(d) a user can watch the recorded video streaming by using a mobile phone
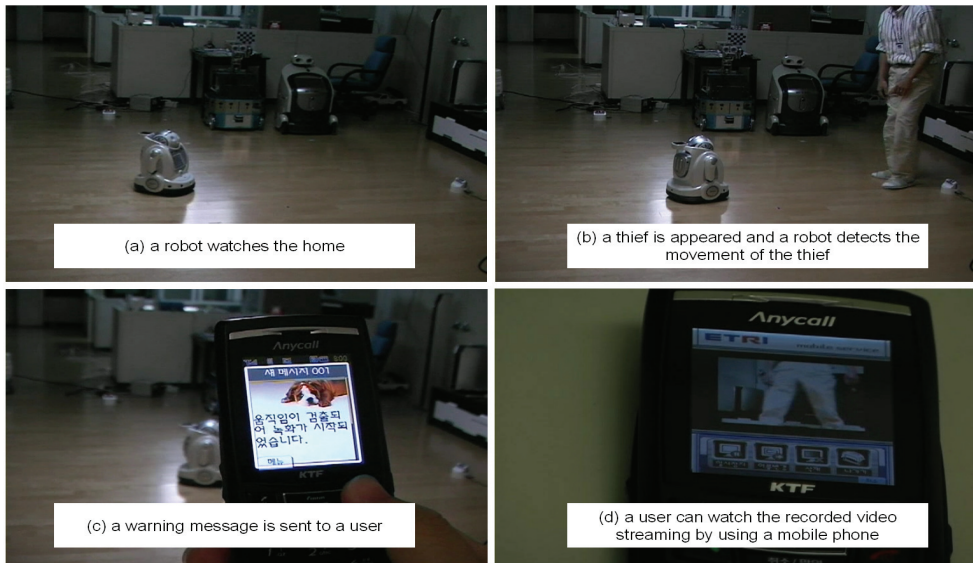
Fig. 10. Snapshots of the home monitoring service

6.4 The follow-me service

The follow-me service is a traditional ubiquitous computing application. By using the CAMUS, we can easily implement the follow-me service. We enhanced the follow-me service with the concept of the software robot (hereafter, sobot). Sobot is a virtual robot which is located in the virtual place. Sobot is responsible for the interaction between the CAMUS and users when they can not be provided services from a real robot. The snapshots of the follow-me service shown in Fig. 11.
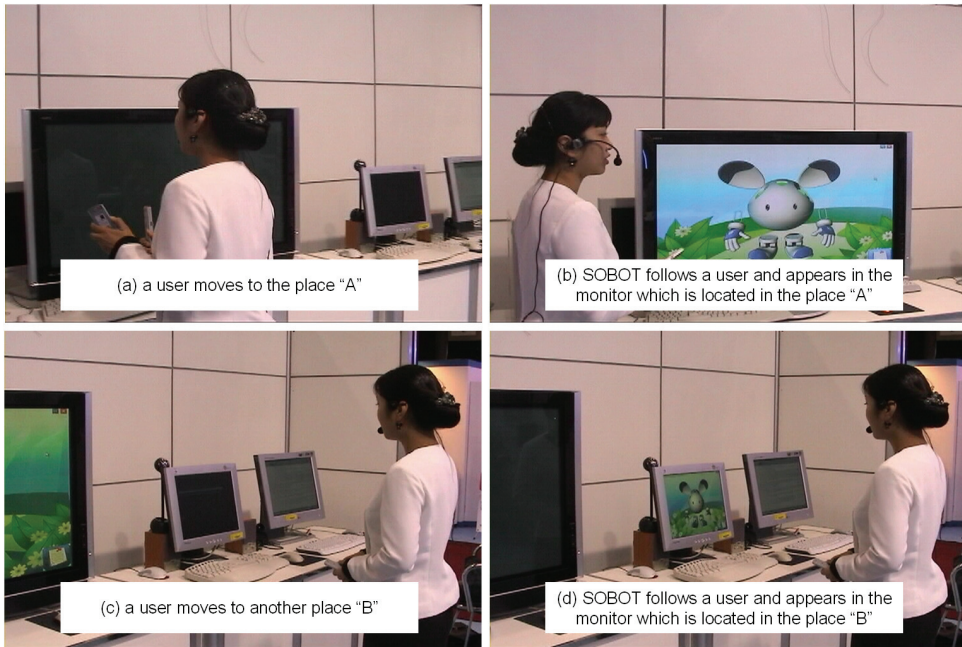
Fig. 11. Snapshots of the follow-me service

7. Conclusions

As the context-aware computing gets more and more interests, attempts applying it to diverse kinds of research fields have been tried. However, the development of the context-aware intelligent application in the robot field has rarely been carried out. In order that the service robot comes into our daily lives like the electric home appliances or mobile devices, it has to provide the context-aware intelligent services to a user.

The aim of this chapter is to introduce the context-awareness for a network-based service robot. For this, we define the functional requirements and implementation issues of the context-aware framework and propose the layered conceptual architecture. And then, we describe how proposed conceptual architecture was implemented to the CAMUS which is a context-aware framework for the network-based service robots. Finally, we introduce various kinds of robot services which were developed by using the CAMUS. As a result of

this work, we can easily develop various kinds of context-aware intelligent services by using the CAMUS.

There are many works that have to be researched. One of the remained works is the navigation of a robot. The technology of the robot navigation is not enough to be applied to a commercial product. Also, we have to consider the implementation issues mentioned in section 3 and should answer how to resolve it. I believe, nevertheless, that the network-based intelligent service robot will be very promising research field and it can improve the life of the human in the future.

8. References

Baldauf, M., Dustdar, S. & Rosenberg, F. (2007). A Survey on Context-Aware Systems, *Int. J. Ad Hoc and Ubiquitous Computing*, Vol. 2, No. 4, pp. 263-277, ISSN 1743-8225

Biegel, G. & Cahill, V. (2004). A Framework for Developing Mobile, Context-aware Applications, *IEEE International Conference on Pervasive Computing and Communications*

Chen, H. (2004). An Intelligent Broker Architecture for Pervasive Context-Aware Systems, *PhD thesis*, University of Maryland, Baltimore County

Chen, H., Finin, T. & Joshi, A. (2003). An ontology for context-aware pervasive computing environments, The Knowledge Engineering Review, Vol. 18. No.3

Dey, A.K. (2000). Providing Architectural Support for Building Context-Aware Applications, *PhD thesis*, Georgia Institute of Technology

Dey, A.K., Abowd, G.D. & Salber, D. (2001). A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications, *Anchor article of a special issue on Context-Aware Computing, Human-Computer Interaction (HCI) Journal*, Vol. 16

Fahy, P. & Clarke, S. (2004). CASS–a Middleware for Mobile Context-Aware Applications. In Workshop on Context Awareness, *MobiSys*

Gu, T., Pung, H. K. & Zhang, D. Q. (2004). A middleware for building context-aware mobile services, *In Proceedings of IEEE Vehicular Technology Conference (VTC)*, Milan, Italy

Hofer, T., Schwinger, W., Pichler, M., Leonhartsberger, G. & Altmann, J. (2002), Context-Awareness on Mobile Devices – the Hydrogen Approach, *In Proceedings of the 36th Annual Hawaii International Conference on System Sciences*

Hong, C-S., Cho, J., Lee, K-W., Suh, Y-H., Kim, H. & Lee, H-C. (2006). Developing Context-Aware System for Providing Intelligent Robot Services, *Proceedings of European Conference on Smart Sensing and Context (EUROSSC2006)*, LNCS 4272, pp. 174–189, ISBN 3-540-47842-6, Enschede, Netherlands, October 2006, Springer

Kim, H., Cho., Y.J. & Oh, S.R. (2005). CAMUS - A middleware supporting context-aware services for network-based robots, *IEEE Workshop on Advanced Robotics and Its Social Impacts*, Nagoya, Japan

Korpipaa, P., Mantyjarvi, J., Kela, J., Keranen, H. & Malm, E.-J. (2003). Managing Context Information in Mobile Devices, *IEEE Pervasive Computing*, July, pp. 42-51

Wang, X.H., Zhang, D.Q., Gu, T. & Pung, H.K. (2004). Ontology based Context Modeling and Reasoning using OWL, *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*

Yau, S.S., Karim, F., Wang, Y., Wang, B. & Gupta, S. (2002). Reconfigurable Context-
     Sensitive Middleware for Pervasive Computing, *IEEE Pervasive Computing joint
     special issue with IEEE Personal Communications*, Vol. 1, No. 3

Development of Intelligent Service Robotic System Based on Robot Technology Middleware

Songmin Jia, Yoshiro HADA, Takayuki Ohnishi,
Harunori Gakuhari and Kunikatsu Takase
University of Electro-Communications
1-5-1 Chofugaoka, Chofu-City, Tokyo 182-8585,
Japan

1. Introduction

Many service robotic systems have been developed in order to improve care cost and the QoL (Quality of Life) of the elderly people in the population-aging society (Nagi, N. Newman et al., 2002; Schulz, D. et al., 2002; Stein, M. R. Stein et al. 2000). Our HARSP (Human-Assistance Robotic System Project) project has been developing a network distributed Human-Assistance Robotic System since 2000 (Jia et al., 2001; Jia et al., 2004). We developed hardware base, key technologies, and implemented CORBA (Common Object Request Broker Architecture) application servers to provide some basic services to aid the aged or disabled. Improvements of the developed system have been doing from practical viewpoint. In order to enable remote users to get a better understanding of the state of robots working and the state of the aged or disabled, we developed network distributed monitoring system to transmit media streams in real time. Home robot integration system and network monitoring system using QuickCam Orbit cameras were developed and demonstrated from June 9 to June 19 at the 2005 World Exposition, Aichi, Japan. The omnidirectional wheelchair and maneuvering system have been developed to enable the disabled person to operate it skilfully. The developed powered wheelchair gives users the same degree of mobility that healthy individuals enjoy (Ohnishi et al., 2002). This new wheelchair can be operated by a disabled wheelchair user using either a joystick or by simple body actions with two hands free. Force sensors arranged under the seat detect the dynamic changes in the user's posture in the wheelchair. We also developed the omnidirectional intelligent service robot which can bring the necessary objects such as a canned drink, a towel or a newspaper to wheelchair user. In order for the service robot to perform a task autonomously, iGPS (indoor Global Position System, Hada et al., 2004) was developed to localize the mobile robot. In addition, a network distributed monitoring system was developed in order to observe the state of the robotic system, as well as the state of the user. In the developed system, Robot Technology Middleware framework was used to develop robotic functional elements as RT components that can facilitate network-distributed software and sharing. AIST (Agency of Industrial Science and Technology, Japan, http://www.is.aist.go.jp/rt/) developed RTM to promote application of Robot

Technology (RT) in various fields (Ando et al., 2004). Modularization of robot components is a key technology for the efficient construction of robot systems. RTM is based on CORBA (omniORB), so the components of the system can be implemented by different programming languages, run in different operating system, or connected in different networks to allow interoperation. We develop robotic functional elements as "RT component", which makes application and system integration easier. It is very easy for the user to create new application system by re-using existing RT components, thus lowers the cost of development of new robotic system. This paper introduces the architecture of the developed system and the results of experiments demonstrated from June 9 to June 19 at the 2005 World Exposition, Aichi, Japan.

The rest of the paper consists of 5 sections. Section 2 describes concepts of Robot Technology Middleware (RTM) developed by AIST. Section 3 presents the developed home integration robot system. Section 4 introduces the developed network monitoring system using QuickCam Orbit cameras. The experimental results are given in Section 5. Section 6 concludes the paper.

2. Robot technology middleware

Robot Technology Middleware (RTM, Kitagaki et al., 2004) was developed by Agency of Industrial Science and Technology of Japan (AIST) to promote application of Robot Technology (RT) in various fields. RTM aims to modularize robotic functional elements as "RT software component", which enables the system to be extended and integrated easier for a new system or new applications. In order to enable system to be language independence, operating system independence, RTM has been developed based on CORBA. CORBA (Common Object Request Broker Architecture, Object Management Group, http://www.omg.org) is a distributed computing technology. There are the other distributed middleware like RMI (Remote Method Invocation, (Java remote method invocation, http://java.sun.com/products/jdk/rmi/index.html), DCOM (Distributed Component Object Model), MOM (Messages Oriented Middleware, Message-orientated middleware: http://sims.berkeley.edu/courses/is206/f97/GroupB/mom/). In contrast to all of these, CORBA uses an Object Request Broker (ORB, Object Oriented Concepts, Inc., http://www.omg.org) as the middleware that establishes a client-server relationship between objects, and it is an object-oriented extension of Remote Procedure Calls (RPCs). CORBA uses GIOPs (General Inter-ORB Protocols), ESIOPs (Environment Specific Inter-ORB Protocols) and IIOP (Internet Inter-ORB Protocols) to implement a truly heterogeneous distributed system. This heterogeneity enables CORBA to inter-operate ORBs purchased from different vendors and supported on different platforms. RTM used OminORB 4.0.5 to implement framework. OmniORB is a robust, high-performance CORBA 2 ORB, developed by AT & T. It is one of only three ORBs to be awarded the Open Group's Open Brand for CORBA. This means that omniORB has been tested and certified CORBA 2.1 compliant. OmniORB implements the specification 2.3 of the Common Object Request Broker Architecture (CORBA).

2.1 Structure of RTM
RTM includes Standard RT Components, Standard RT Services, RT Components Framework and RT Library (RTM::RtcBase, RTM::InPortBase, RTM::OutPortBase and so on). Figure 1 shows the structure of RT middleware.
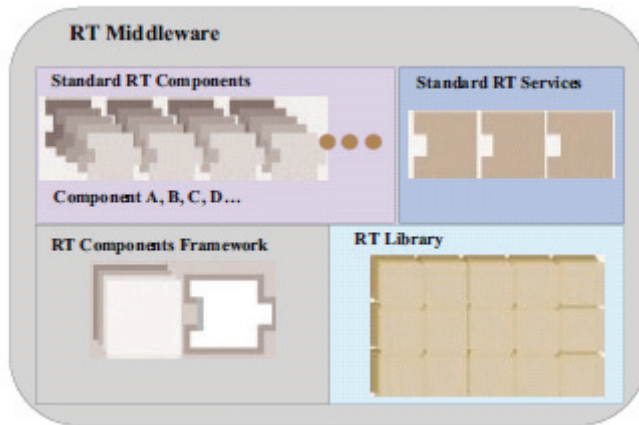
Fig. 1. Structure of Robot Technology Middleware.

2.2 RT Component

RT Component includes RT body object, Inport object and Outport object. RT Component's features are such as it has activity, interface for connecting output and input data stream and administrative function of component-objects. Figure 2 illustrates the structure of RT component.
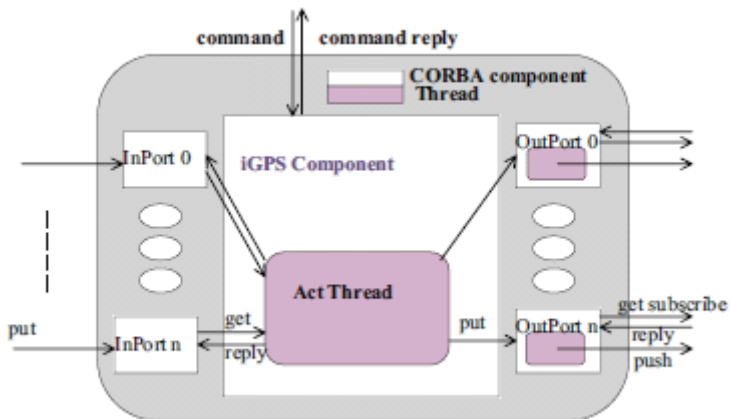
Fig.2. Structure of RT Component.

2.3 RT component manager

Figure 3 illustrates the relationship among RT Component, Manager and name service. RT Component Manager manages all RT Component, controls activity thread, adds or deletes the object of RT Component and does instance of Component. Registering all RT Components' names and object references to name server using CORBA name service enable the other host to access the them in different network.
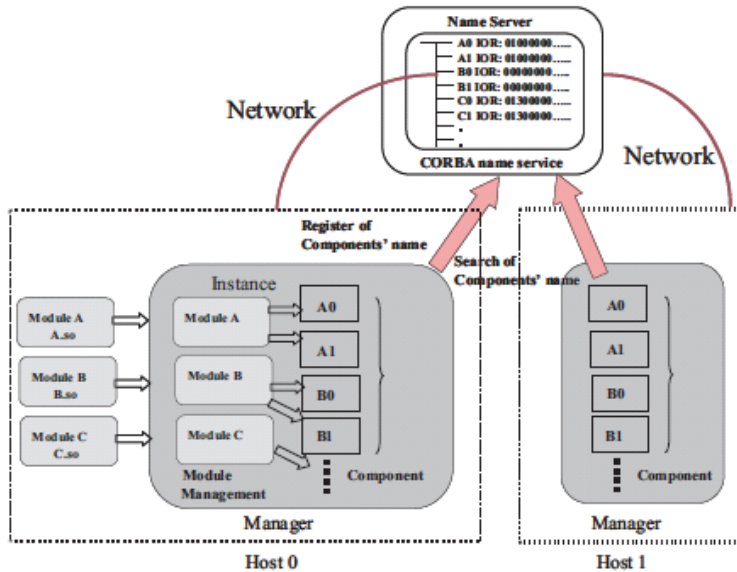
Fig. 3. The relationship among RT Component, Manager and name service.

3. System demonstration at the 2005 World Exposition, Aichi, Japan

Home robot integration system and network monitoring system using QuickCam Orbit cameras were developed and demonstrated from June 9 to June 19 at the 2005 World Exposition, Aichi, Japan. Figure 4 shows the architecture of the developed intelligent home service robotic system.
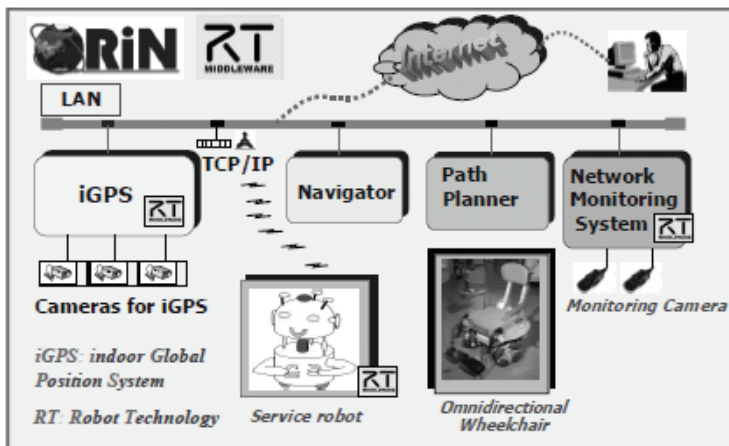
Fig. 4. Structure of the developed intelligent service robotic system.

3.1 Multi-robot systems

The omnidirectional wheelchair was developed to give the aged or disabled the same degree of mobility that healthy people enjoy. The weight of the developed wheelchair is about 80 kg, the maximum acceleration is about 0.41 G, the maximum velocity is about 1.3 m/s and the time of continuous operation time is about 0.5 hour. The user can control wheelchair by a joystick that is applied mainly as user interface. In addition, a novel steering interface was also developed for holonomic omnidirectional power wheelchair to observe user's body action such as tilting an upper body or twisting a waist in order to get user's intention. The observation of changing user's posture on wheelchair is implemented by force sensors arranged under the seat. The user can maneuver the wheelchair by simple body actions with free both hand, so that he or she can enjoy playing a sport such as tennis. Figure 5 illustrates the developed omnidirectional power wheelchair and its specification.

Specifications

Weight	80kg
Maximum acceleration	4.1 m/s^2
Maximum velocity	1.3 m/s
Continuous operating time	0.5 hours

Fig. 5. The developed omnidirectional wheelchair

An intelligent omnidirectional errand robot was developed to help the disabled wheelchair user. The weight of the developed errand robot is about 50 kg, the maximum velocity is about 1.2 m/s and the time of continuous operation time is about 2 hour. When the aged or disabled wheelchair user sends a command such as "juice, please" via PDA (Personal Digital Assistance), the errand robot can autonomously move to the place where the juice is, and take the juice, then confirm the position and orientation of the wheelchair, move to the place where the aged or disabled wheelchair user is, and hand off the juice. Even if the wheelchair user changed the position or orientation while the errand robot was executing a task, the robot can recognize the changes and perform the task autonomously because the robot can get the information of the wheelchair user's position via iGPS. All robots are equipped with infrared (IR) LED units. An IR pass filter, which removes light in the visible spectrum, is attached to the TV camera lens. These TV cameras detect the position and orientation of mobile robot accurately. Figure 6 illustrates the developed errand robot and its specifications.

Specifications

Weight	50kg
Maximum acceleration	0.4 m/s^2
Maximum velocity	1.2 m/s
Continuous operating time	2 hours
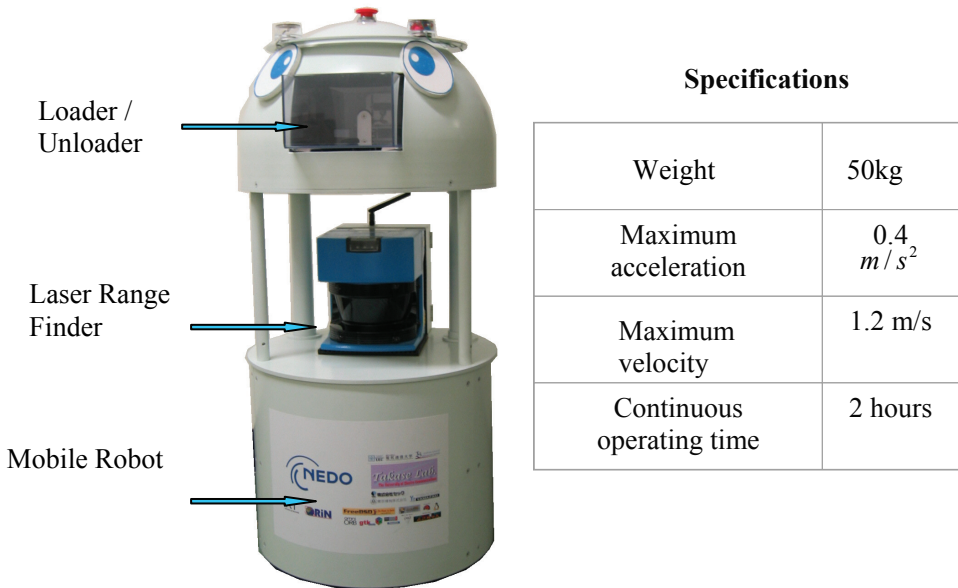
Fig. 6. Developed errand robot system.

3.2 Task management server

We developed task management server for home integration robotic system. The task management server accepts commands from the robot operation management server and manages the behavior of each robot. The task management server has two functions. The first function is to manage the behavior of the errand robot. The server accepts the the commands given in Robot Action Command (RAC) format from the robot operation management server. Figure 7 illustrates some samples of Robot Action Commands divided into a sequence of robot behaviors. For example, Robot Action Command "bring a canned drink" can be divided into sequence of robot behavious such as: "go to the place where the canned drink is, pick up the container, go to the place where the wheel chair user is , give the object and so on". These commands are sequentially transmitted to the navigation server via CORBA communication. When the navigation server notifies the Task Management Server that it finished executing the given command, the next command is transmitted to the navigation server from the task management server.

The second function is to manage the state of the robot. The state of robot is periodically transmitted from the navigator server, and stored in its memory. The task management server was developed by ORiN (Open Resource interface for the Network/Open Robot interface for the Network, OriN", http://www.orin.jp/) provides integrated interface to access to the devices on the network. One of them is Application Program Interface (API) and the other is communication interface which absorbs the differences between the various specifications of the data storage or the communication protocols between personal computers and devices. Furthermore, ORiN provides protocols to connect to the Internet.
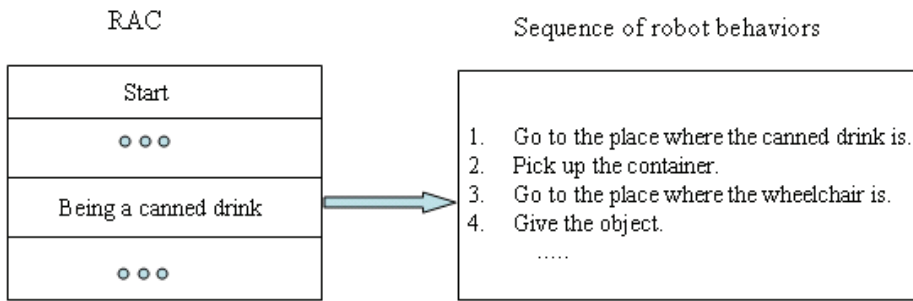
Fig. 7. RAC into a sequence of robot behaviors

3.3 Robot navigation server

The Robot Navigation Server receives the task level directive which was constructed by the Task Management Server, gathers information of robots and environments, generates the operation level instructions in real time, and sends the instructions to all robots. Figure 8 illustrates the architecture of developed Robot Navigation Server.

The navigation server is implemented on a desktop PC (AthlonTM64 3500+) running FreeBSD 4.11-STABLE. This server is designed as a part of the distributed system, and CORBA (implementation of omniORB4) is employed as middleware.
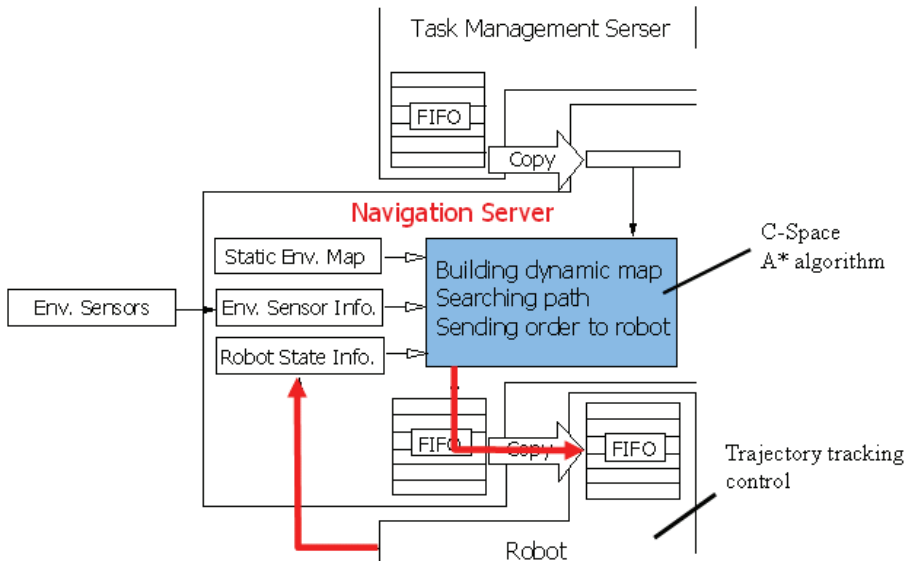
Fig. 8. Developed Errand mobile Robot navigation server.

3.3.1 Task level directive

The following seven task directives are issued from the Task Management Server to the Errand Robot Navigation Server: (1) Canceling a directive, (2) Moving a robot absolutely, (3)

Moving a robot relatively, (4) Loading a container, (5) Unloading a container, (6) Pushing a container halfway, and (7) Pulling a container. Directive (2) is used to move the robot toward a target position given with respect to the global coordinate system. When the robot receives Directive (3), it moves to the relatively designated position. Directives (4) and (5) are sent when it is necessary for the robot to load/unload a container. Directives (7) and (8) are issued to have the robot give an object to the user. The navigation server handles only one task level directive per robot at the any time. If two or more directives are received at the same time, the extra directives are rejected. The user can also send directives using the PDA, or another command terminal, manually. If the received directive is impracticability, then the navigation server rejects the directive and informs the task management server that the navigation server cannot executing a directive. When the directive is finished, the navigation server notifies the work management server and then waits for a new directive.

3.3.2 Operation level instructions

The environment map used by the robot navigation server consists of a static map and a dynamic map. The static map includes geometric information, such as the shape of the robot workspace and stationary objects like desks or shelves. The static map is built beforehand. After the robot finishes an operation level instruction, the navigation server gathers information about uncharted dynamic bjects such as other robots and chairs, sensed by the sensors arranged on the robots or in the environment. And the navigation server refreshes the environment maps using this information. Using the update map, the navigation server builds Configuration space (C-Space) as a search space to find a collision free path. The A∗ algorithm (Gakuhari, et al., 2004) is used to determine the shortest path in the approximate C-Space. The obtained path is represented as a sequence of cells.

One-step motion from one cell to a neighboring cell in the approximate C-Space and manipulation of the container are treated as one operation level instruction. Therefore, one task level directive is divided into a sequence of operation level instructions. The navigation server stores these operation level instructions in a multistage pipeline. The instructions are copied to the multistage pipeline of the robot immediately via CORBA communication. After the robot finishes executing all of the instructions, in other words, when the pipeline is emptied, the navigation server issues the next instructions until the given task level directive will be finished.

As shown in Figure 9, the robot generates a trajectory from instructions that represent translation and rotation motion. The current pose of the robot can be determined by iGPS and an odometer, and a smooth trajectory from the current pose to a target pose can be generated using the third-order B-spine interpolation algorithm.

The navigation server runs on i386 architecture PC (CPU: Pentium 4 2.40B GHz, Memory: PC2100 512 MB) with FreeBSD 4.11-RELEASE. According to the experimental results, the server can dynamically navigate the robots in real time without a collision or a dead-lock in an environment. When the robot finishes one instruction, the robot notifies the navigation server and deletes the corresponding instruction from the pipe-line. When the server received the notification, it also deletes the corresponding instruction from the pipe-line, and checks the number of the instructions remaining in pipe-line. If the number is less than the configured value, then the navigation server generates new instructions.
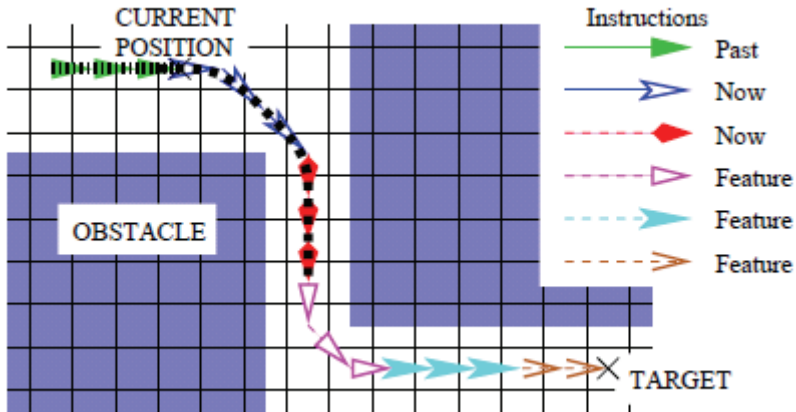
Fig. 9. Trajectory generation from s sequence of the task level instructions.

3.3.3 Dealing with abnormal circumstances

When the robot enters an abnormal state, all servers and the robot process stop immediately. The typical examples of abnormal circumstances are given as following:

- When the user gives the stop directive by using a PDA or something commanding terminal, the task management server gives the stop directive to the robot navigation server. When the navigation server receives the stop directive, it sends the cancel instruction to the robot, and removes all of the operation level instruction from the pipeline. The navigation server then waits for the next directive from the task management server.
- When the emergency button of the robot is pushed, the robot stops moving and notifies the navigation server. The navigation server then dequeues all of the operation level instructions from the pipeline. The navigation server then waits for the next directive from the task management server.
- If disconnection of the communication channel between the navigation server and the robot is detected, the navigation server and the robot try to recover the connection. If the disconnection is fatal, autonomous recovery is impossible and the connection must be fixed manually.

3.4 iGPS RT functional component

We developed iGPS RT functional component in order to enable system integration easier. An indoor Global Positioning System (iGPS) has been developed to localize the omnidirectional mobile robot. IEEE 1394 cameras, are mounted on the ceiling so that the cameras overlook the robot's moving area (Hada at el., 2005). We evaluated accuracy of iGPS by experiments. We selected 24 points distributed in the Lab about 7000 mm×7000 mm area, and measured them using iGPS and the maximum value of measurement error is 38 mm. This result verified that the accuracy of iGPS is enough for navigation of mobile robot.

4. Network distributed monitoring system using QuickCam Orbit cameras

In order to enable a remote user to get a better understanding of the local environment, media streams must be received and transmitted in real-time in order to improve interaction in home integration robot system. We implemented video/audio RT component based on RT Middleware, and OmniCORBA IIOP is employed as message communication protocol between RT component and requester. The QuickCam Orbit (Logitech Co.) cameras were used in our system with high-quality videos at true CCD 640×480 resolution, automatic face-tracking and mechanical Pan, Tilt and face tracking feature. This camera has a maximum video frame rate is 30 fps (frames per second) and works with both USB 2.0 and 1.1. The area of the booth used to demonstrate the developed robotic system was approximately 4.5×5 m², so two cameras were set up in the environment. The cameras were able to view the area in which the omnidirectional wheelchair and errand robot move by adjusting the mechanical Pan and Tilt of the cameras. The structure of the developed RT video stream functional component is shown in Figure 10. This RT component has one Inport for camera property setting and Outport 1 for video data and Outport 2 for status data of camera control.

- Inport: camera property for camera's setting
- Outport1: video data
- Outport2: status data for camera control
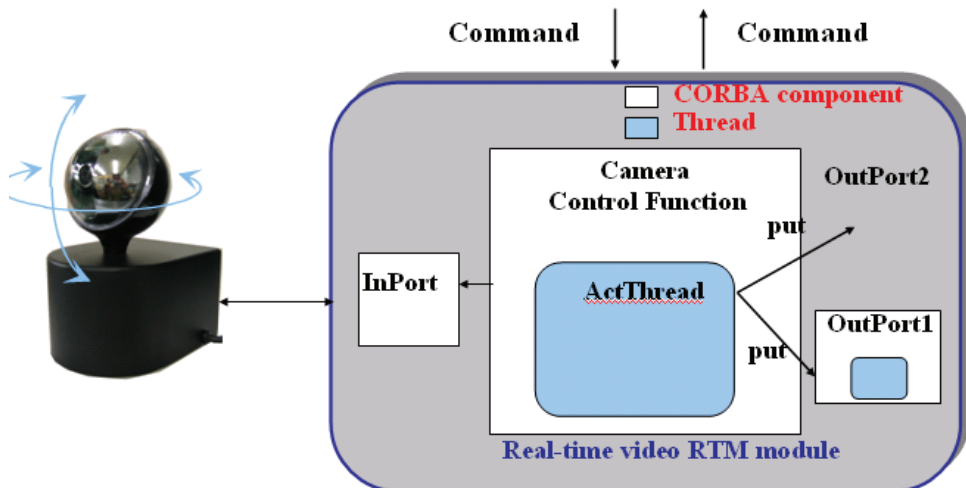
Fig. 10. Video/audio RT component developed based on RTM.

Figure 11 illustrates the class structure of the developed video RT component. The camera's control function classes includes:

- RtcBase: OpenRTM-aist-0.2.0 component base class.
- InPortBase: OpenRTM-aist-0.2.0 InPort base class.
- OutPortBase: OpenRTM-aist-0.2.0 OutPort base class.

- InPortAny<TimedUShortSeq>: InPort template class.
- OutPortAny<TimedUShortSeq>: OutPort template class.
- RtcManager: RT component management class.
- CameraRTC: camera control RT component.
- CameraComp: camera control RT component main class.
- CameraControl: camera operation class.
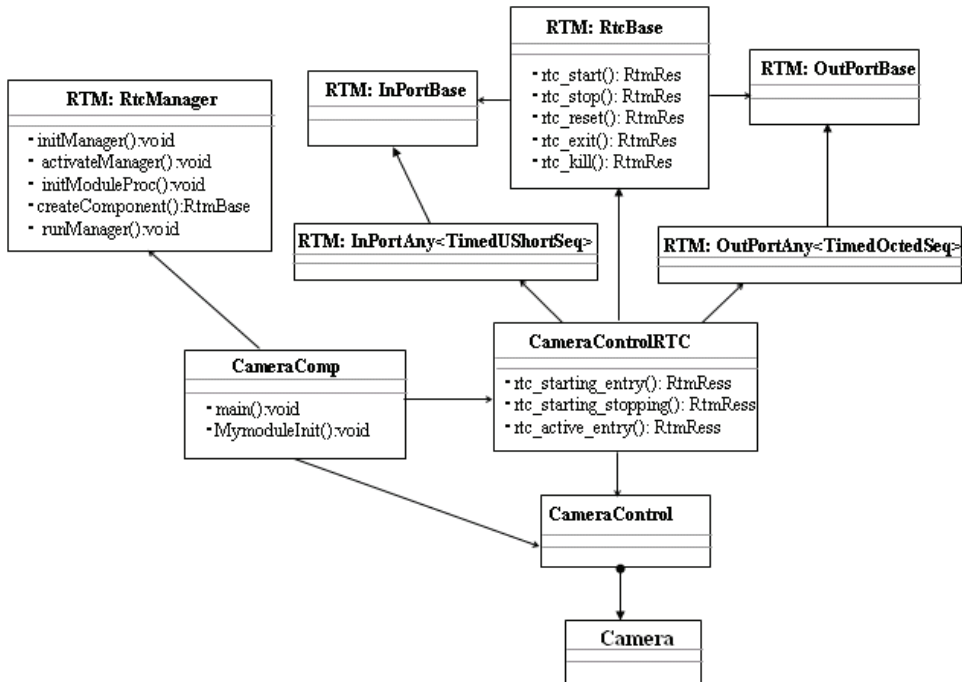
Fig. 11. Class structure of the developed RT component.

 In addition, we developed a graphic user interface (GUI) for the video stream system that provides a remote video stream camera zoom and pan-tilt adjustment, and a chat function that allows a remote user to communicate with a local user. When the user sends a request for video, the system will autonomously display the GUI. The user can click "Connect" and input the IP address of the computer on which the RT video component is running to view a real-time video feed.

The RT video stream component was implemented by Visual C++, Microsoft visual studio.net 2003. A performance test of the developed real-time video stream was conducted to examine the possibility of using a live video feed to monitor the state of the elderly or disabled wheelchair user. The video server is run on Windows 2000 Professional (1.9 GHz, Pentium4), and the video client is run on Windows XP (2.4 GHz, Pentium4). The average frame rate is approximately 16.5 fps (video format 320×288). Figure 12 illustrates the architecture of the developed network monitoring system based on RTM.
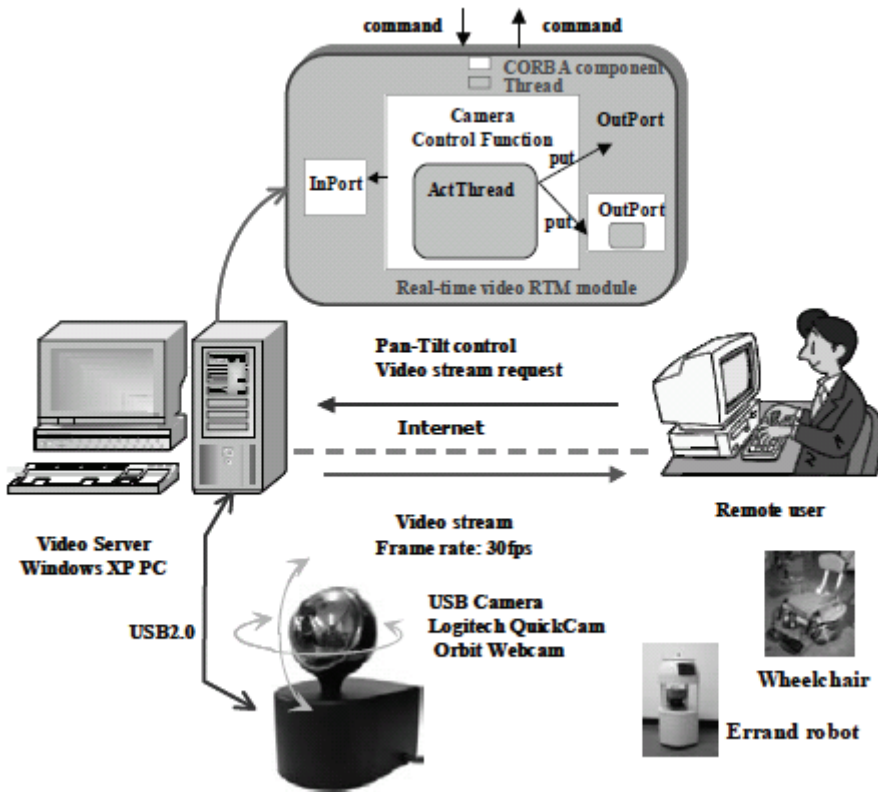
Fig. 12. Structure of RT video stream functional component.

5. Experimental results

Home integration robotic system was demonstrated from June 9 to June 19 at the 2005 World Exposition, Aichi, Japan. Figure 13 illustrates the scenery of demonstration in the 2005 World Exposition, Aichi, Japan. Figure 13(a) is a modelled living room at the prototype robot exhibition and 13(b) is the booth for our developed system demonstration. Figure 13(c)-(f) illustrates some images of task performance demonstration of robotic system performing a service task. The wheelchair user can issue an order to the robot to bring objects such a canned drink via PDA. Then the errand robot starts to move toward the front of the shelf where the container holding the target canned drink is placed and loads the container. The errand robot can offer the canned drink to the wheelchair user because the robot can obtain position information of the wheelchair via iGPS. Even if the wheelchair user changed the position or orientation while the errand robot was executing a task, the robot can recognize the changes and perform the task autonomously. Fig. 13(g)-(i) illustrates the video stream for monitoring the state of robotic systems working. The developed network distributed monitoring system can monitor the state of robotic system's working

and the state of the aged or disabled in demonstration. Cameras for monitoring the environment were connected to the computer running on Windows XP (2.4 GHz, Pentium4), and GUI is run on the other same specification Windows XP (2.4 GHz, Pentium4). Two computers are connected in a LAN. The average frame rate is approximately 18.5 fps. Figure 14(a)-(h) shows the performance demonstration of the omnidirectional powered wheelchair. The user operates the wheelchair through the joystick skilfully (Figure 14(a)-(d)). The user can also operate the wheelchair via a body action control interface which enables hands-free maneuvering of the wheelchair, so that he or she can enjoy playing a ball with two hands (Figure 14(e)-(h)). The demonstration time was approximately held twice a day. A total of 22 demonstrations were performed and the errand robot failed to execute its task three times. The success rate is about 86%. The cause of the failure was that the angle of Camera 2 changed over time so that the calibrated camera parameters differed from the original parameters, causing an error in the measurement of the robot position. When Camera 2 was neglected, the robot did not fail to execute its task. The demonstration verified that the developed system can support the aged or disabled to a certain degree in daily life.
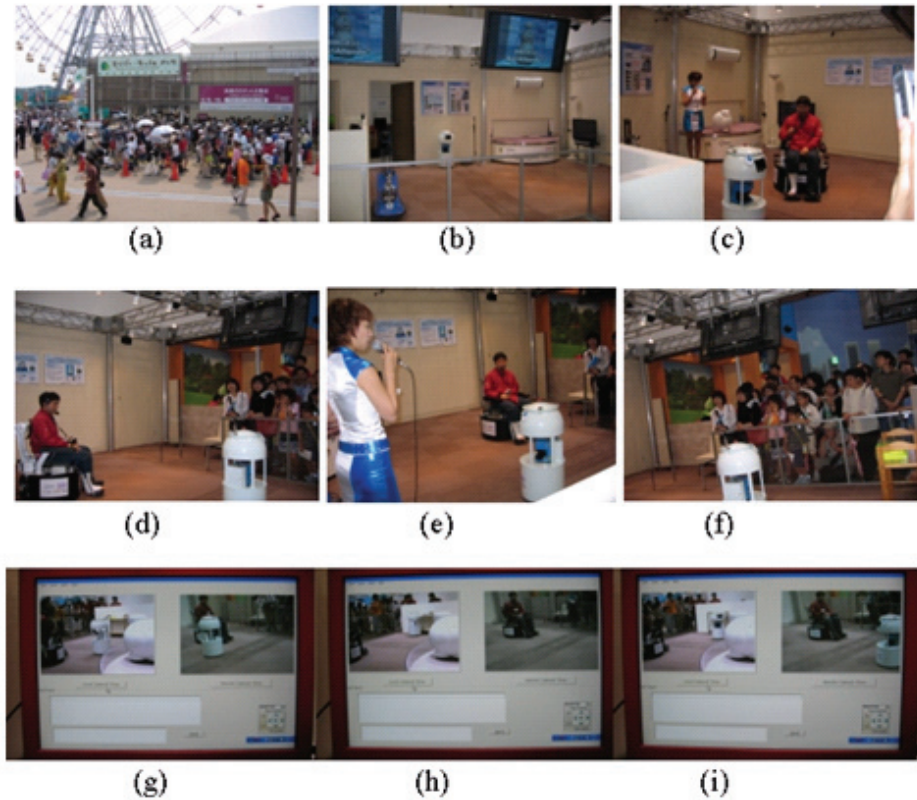
Fig. 13. Some images of task performance demonstration of robotic system performing the task

(a)   (b)   (c)   (d)

(e)   (f)   (g)   (h)
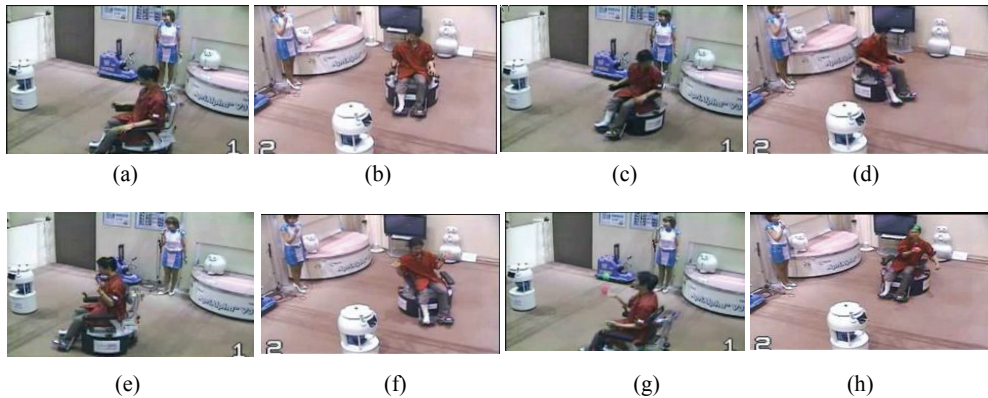
Fig. 14. The demonstration of the omnidirectional powered wheelchair.

6. Conclusion

This paper presented the developed service robotic system supporting elderly or disabled wheelchair users. Home integration system was demonstrated at the prototype robot exhibition from June 9 to June 19 June at the 2005 World Exposition, Aichi, Japan. We developed an omnidirectional wheelchair and its maneuvering system to enable skilful operation by disabled wheelchair user. Since the user can maneuver the wheelchair intuitively by simple body actions and with both hands free, they are able to enjoy activities such as tennis. We also developed an errand robot that can deliver objects such as newspaper or canned drink to disabled wheelchair users. Even if the wheelchair user changed the position or orientation while the errand robot was executing a task, the robot can recognize the changes and perform the task autonomously because the   robot can get the information of the wheelchair user's position via iGPS. Network monitoring system using QuickCam Orbit cameras was implemented to monitor the state of robotic systems working.

Because Robot Technology Middleware (RTM) was used in the developed system, we can develop the functional module as RT component, which makes the system has high scaling and inter-operating ability, facilitating network-distributed software and sharing, and makes application and system integration easier. It is also very easy for the user to create new application system by re-using existing RT components, thus lowers the cost of development of new robotic system. For future work, we will develop the other functional robot system components as RT components such as RFID RT object recognition component for object recognition or RT localization component for localizing mobile robot in order to improve the flexibility of the home integration robotic system.

7. Acknowledgements

8. References

Ando, N., Suehiro, T., Kitagaki, K., Kotoku, T. and Yoon, W., 2004, Implementation of RT composite components and a component manager, The 22nd Annual conference of the Robotic Society of Japan, IC26.

Kitagaki, K., Suehiro, N., Ando, N., Kotoku, T., and Yoon, W., 2004, GUI components for system development using RT components," The 22nd Annual conference of the Robotic Society of Japan, IC23,2004.

Jia, S., and Takase, K., 2001, Internet-based robotic system using CORBA as communication architecture, Journal of Intelligent and Robotic System, 34(2), pp. 121-134, 2001.

Jia, S., Hada, Y. and Takase, K., 2004, Distributed Telerobotics System Based on Common Object Request Broker Architecture, The International Journal of Intelligent and Robotic Systems, No.39, pp. 89-103, 2004.

Gakuhari, H., Jia, S., Hada, Y., and Takase, K., 2004, Real-Time Navigation for Multiple Mobile Robots in a Dynamic Environment, Proceedings of the 2004 IEEE Conference on Robotics, Automation and Mechatronics, Singapore, pp. 113-118.

Hada, Y., Gakuhari, H., Takase, K.and Hemeldan, E.I., 2004, Delivery Service Robot Using Distributed Acquisition, Actuators and Intelligence, Proceeding of 2004 IEEE/RSJ International Conference on Intelligent Robots and System (IROS'2004), pp. 2997-3002.

Hada, Y., Jia, S., Takase, K., Gakuhari, H., Nakamoto, H., and Ohnishi, T., 2005, Development of Home Robot Integration System Based on Robot Technology Middleware, The 36th International Symposium on Robotics (ISR 2005), TU4H6, Japan.

Message-orientated middleware: http://sims.berkeley.edu/ courses/is206/f97/ GroupB /mom/.

Ohnishi, T., and Takase, K., 2002, The maneuvering system of omnidirectional wheelchair by changing user's posture, Proceeding of 2002 international Conference on Control, Automation and System (ICCAS2002), pp. 1438-1443, 2002. http://www.orin.jp/.

Object Management Group, http://www.omg.org.

Object Oriented Concepts, Inc., http://www.omg.org.

Java remote method invocation:

http://java.sun.com/products/jdk/rmi/index.html.

http://www.is.aist.go.jp/rt/.

Nagi, N. Newman, W.S., Liberatore, V. (2002), An experiment in Internet-based, human-assisted robotics, Proc. of IEEE Int. Conference on Robotics and Automation (ICRA'2002), Washington, DC, USA, pp.2190-2195.

Schulz, D., Burgard, W., Fox, D. et al.: (2002), Web Interface for Mobile Robots in Public Places, IEEE Robotics and Automation Magazine, 7(1), pp. 48-56.

Stein, M. R. Stein, (2000), Interactive Internet Artistry, *IEEE Robotics and Automation Magazine*, 7(1) (2000), pp. 28-32.

An ITER Relevant Robot for Remote Handling: On the Road to Operation on Tore Supra

Keller Delphine, Friconneau Jean-Pierre and Perrot Yann
CEA LIST Interactive Robotics Unit
France

1. Introduction

In the context of Fusion, several experimental reactors (such as the International Thermonuclear Experimental Reactor (ITER)), research aims to demonstrate the feasibility to produce, on earth, the plasma that occurs on the sun or stars. Fusion using magnetic confinement consists in trapping and maintaining the plasma in a magnetic container with torus shape (Tokamak), under Ultra High Vacuum (10-6 Pa) and high temperature (100 millions °K).

During plasma burning, the severe operating conditions inside the vacuum vessel apply high thermal loads on the first wall Plasma Facing Components (PFCs). Therefore, regular inspections and maintenance of 100% of the first wall surface is highly required. When considering the maintenance between two plasma shots, the conditions to perform maintenance tasks, without breaking the vacuum, exclude human intervention and require use of remote means based on robotic technologies that enable extension of human capabilities into the machine.

The technologic research on robotics and remote operations is called the Remote Handling (R.H.) activity. The Interactive Robotics Unit of CEA-LIST has been working on Remote Handling for Fusion for more than ten years. Experience on JET reactor maintenance has proven the feasibility to maintain an installation with robots controlled by distant operators (A.C. Rolfe et al., 2006), (O. David et al., 2000).

When considering generic Tokamak relevant conditions such as we can find in the CEA Tore Supra Tokamak, the set of major challenges we selected for the Remote Equipment is to sustain the following severe operating conditions: ultra high vacuum (10-6 Pa), temperature (120°C), baking (200°C). The limited number of machine access ports and the very constrained environment complicate the introduction of a robot into the machine. These issues impose an major step in term of technologic research for R.H.: innovation in robot conception, new kinematics, new actuator technologies, hardened electronic components were designed, simulated and tested to cope with the ultra high vacuum and the temperature constraints.

Since 2000, under EFDA (European Fusion Development Agreement) support, the Interactive Robotics Unit of CEA-LIST and the CEA-DRFC of Cadarache collaborate on a potential ITER relevant Remote Handling Equipment (RHE). The main challenge of the project is to demonstrate the feasibility of close inspection of a plasma chamber In Vessel

first wall with a long reach robotic equipment, under some ITER requirements: Ultra High Vacuum (10-6 Pa), temperature 120°C and 200°C during the outgassing phase to avoid pollution chamber. The proof of feasibility is performed on the existing CEA facilities called Tore Supra (TS), which is an experimental fusion machine using superconducting coils and water cooled plasma facing component (like ITER) located in Cadarache facilities (R=2.3m, r=0.8m for torus dimensions).

The Remote Handling Equipment (RHE) designed for this application is composed of a Robotic Equipment called Articulated Inspection Arm (AIA), a video process and a Tokamak Equipment which enables conditioning and a precise guiding of the robot. (Fig. 1)
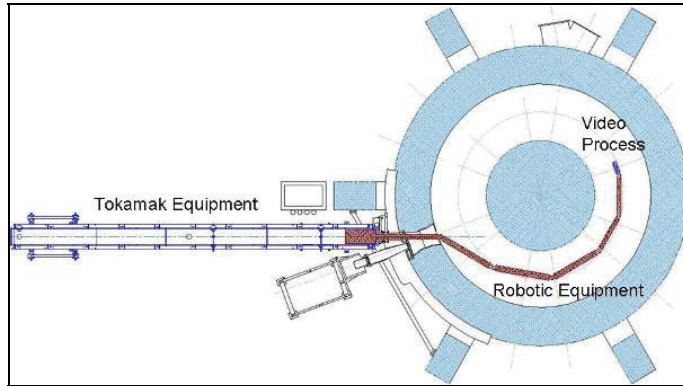
Fig. 1. View of the Remote Handling Equipment (RHE) in Tore Supra

Since the first conceptual design in 2000, succession of mock up, tests campaigns, tuning and design enhancements lead, in 2007, to the prototype module qualification under real operating conditions, Ultra High Vacuum (10^{-6} Pa) and temperature (120°C). The full robot is then manufactured, assembled and tested under atmospheric conditions on a scale one mock up in Cadarache facilities. The robotic equipment is assembled to the Tokamak Equipment for the complete qualification of the RHE connection on Vacuum Vessel.

 In September 2007, 12th the successful feasibility demonstration of close inspection with a long reach poly-articulated robot carrier in Tore Supra is proved under atmospheric conditions.

Next milestone is the complete robot qualification under real operating conditions. At this step of the project, the robot prototype needs or could need further developments to meet 100% of the ITER operational requirements.

The RHE has to be used in real operating conditions to collect knowledge on the system behaviour. The design and command control has to be enhanced toward robustness and reliability. Further developments on command control and modelling taking into consideration the structure deformation are still necessary to have good confidence on the robot position in the 3D environment. Reliability of the complete RHE and control modes will have to be proved before the final RHE could be qualified as operational on Tore Supra.

This chapter presents the complete RHE including the Robotic Equipment (RE), the Tokamak Equipment (TE) and the Video Process. An overview of the mechanical and control design principles is presented. Then, technologies selected for the robot to sustain vacuum and temperature are detailed and a presentation of the prototype module and full

RHE qualification tests and successful deployment demonstration in Tore Supra are depicted. The last part presents further developments that could be done in order to enhance the robot performances and manoeuvrability.

2. The AIA RHE design

2.1 Summary of the requirements

Toward the final objective to use the AIA Robotic Equipment on Tore Supra as an inspection tool, and with respect of the ITER relevant conditions, several requirements have to be met and taken into consideration during the robot design:

- Small penetration hole: equatorial port dedicated not larger than 250mm
- Operational full extension, able to reach any point inside the Tokamak, high mobility in the environment.
- Payload: Possibility to plug various processes (up to 10kg); the first process developed is a video camera for inspection.
- Functioning conditions: Ultra High Vacuum (10^{-6} Pa) and temperature: 120°C in use (baking phase 200°C for vacuum conditioning)
- In-Vessel requirement: do not pollute the Tokamak Equipment.

2.2 General design and control

First conceptual designs started in 2000. Simulation results and first computations converge toward the following kinematics structure: a poly articulated robot formed by 5 identical segments and one precise guiding and pushing system at the base, called "Deployer", able to push the robot into the machine. Each module includes up to two degrees of freedom, two rotary joints (one in the horizontal plane and one in the vertical plane) (Y. Perrot et al., 2004).

Main characteristics:

- Cantilever length: 9.5 meters.
- Weight: ~300 kg (5 modules + Deployer).
- Payload: 10 kg.
- 6 modules Ø160 mm, up to 11 degrees of freedom (d.o.f.), (10 rotary joints, 1 prismatic joint at the base).
- Rotary joint (vertical axis): +/- 90°.
- Rotary joint (horizontal axis): +/- 45°.
- Prismatic joint at the base: 10m range. (Fig. 2)
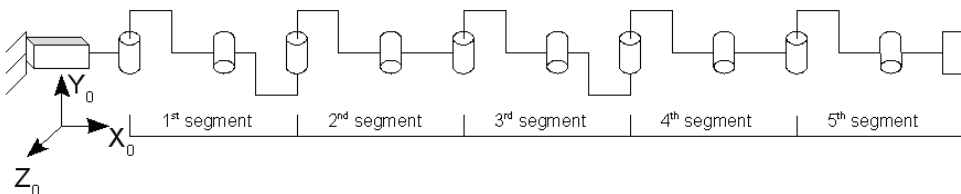
Fig. 2. Simplified AIA kinematics model with 11 d.o.f.

In Fig. 3, the elevation axes are represented with a simple revolution axis whereas, in fact, it is a parallelogram structure that performs the elevation motion in order to minimize the impact of the cantilever structure and keep the axis vertical.
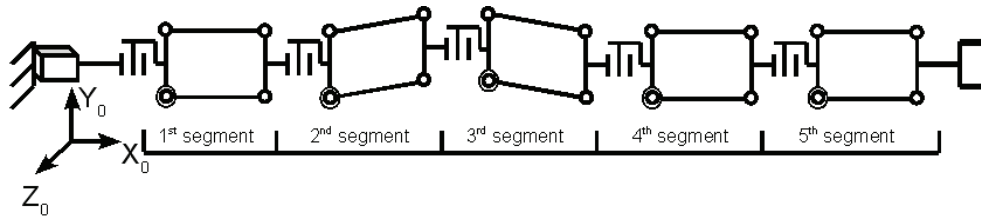
Fig. 3. AIA kinematics model with parallelogram structure

The AIA articulations are actuated by electrical motors. Each module includes on-board temperature hardened control electronics qualified up to 120°C in use and 200°C switched off. The robot can carry a payload of 10 kg at its end effector.

At the moment, the AIA can be piloted by programming the desired angles of the robot's joints (articular control mode).

Limited access of viewing in the Vacuum Vessel requires developing assistance to steering that could be developed in the next phase of the project.

2.3 Mechanical design

The AIA robot carrier is composed of a set of 5 modules and a pushing system (Deployer). The payload is supported by the end effector. Because of the high cantilever structure (9.5 m), the robot elements are submitted to high forces and torques. Tubes and clevis are made of titanium for its mechanical properties even under high temperature, rods are made of bearing steel for its high mechanical resistance in traction.

Each module is a two DOF mechanism: 2 rotary joints (horizontal and vertical axis) with a four-bar mechanism (the parallelogram) composed of the rods, the base clevis, the tube and the head clevis (Fig. 4).
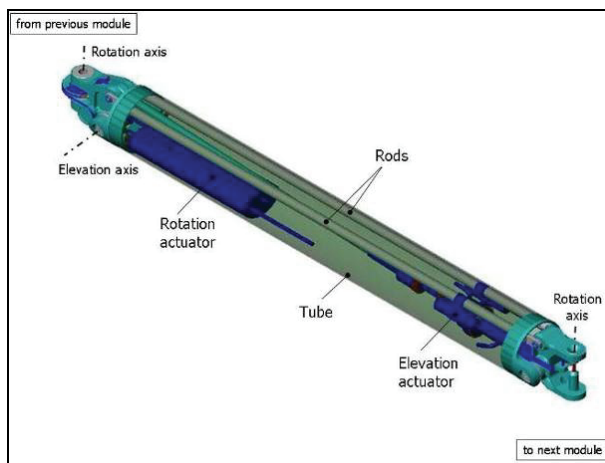
Fig. 4. View of a AIA module

The parallelogram plays a major role in reducing the gravity effect over the joints of the structure by keeping the clevis vertical. Thus, if the parts deformations are neglected, the rotation axis between two modules will also be kept in a vertical position for any given configuration. This property is an advantage for the design because it tends to reduce the

size of the electrical actuators that provide the modules rotation motion. As far as no dynamic motion is required, high ratio gearbox with D.C. motors are satisfactory. The angular displacement in the horizontal plane is set in motion by the actuators through a cable and pulleys system as shown in (Fig. 5).
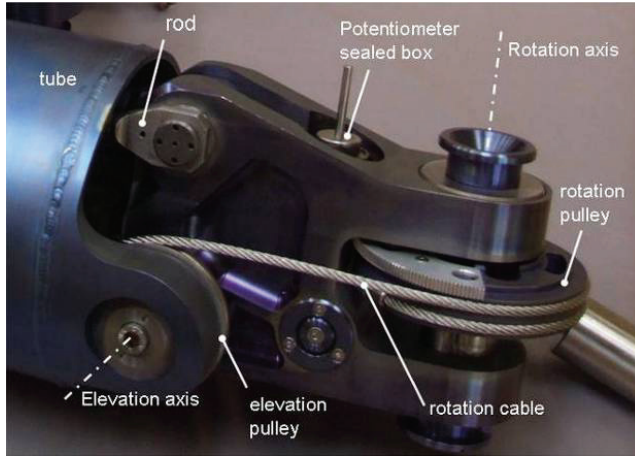
Fig. 5. Rotation cable system

The forces and tensions due to gravity over one of the modules of the AIA robot are represented in Fig. 6. The segments of the robot that follow the current one are modeled by a weight $\vec{P}$ and a moment $\vec{C}$ applied over the clevis. When performing a structural analysis to define the tensions created in the parallelogram structure, we find that:

$$T_1 = \frac{C}{\ell \cos\alpha}$$
$$T_2 = P.\frac{d}{\ell_1}$$
$$T_3 = \frac{C}{\ell \cos\alpha} + P.\frac{L}{\ell_1}$$

(1)
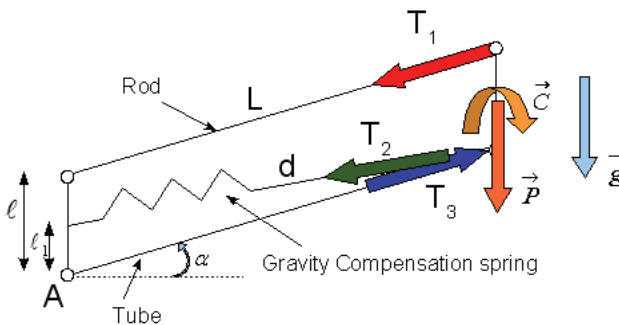
Fig. 6. Gravity forces repartition in the parallelogram structure

Since l = l1 in (1), and considering the basis module, the maximal forces supported by the elements are 64000N in the tube, 40000N in the rods and 25000N in the jack.

The issues of the final Robotic Equipment design are represented in Fig. 7.
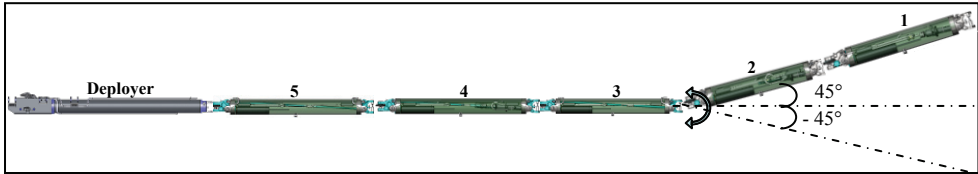
Fig. 7. View of the complete Robotic Equipment design

2.4 Tokamak Equipment design and integration on Tore Supra

As the final objective is to demonstrate the feasibility to use a Robotic Equipment has an inspection tool between two plasmas, the equatorial port of Tore Supra is dedicated to receive the AIA RHE. It means that the robot must be already conditioned under the same vacuum and temperature level as the Tokamak ones to avoid any perturbation during the robot introduction into the machine. In this context, a long storage cask has been designed. It is provided to allow the robot conditioning and precise guiding of the Deployer (Fig. 8).

This large structure: 11m long, 3m height and about 5 tonnes in operating mode is carried by 2 rolling wagons operated by winches and guide rails on the ground. One of the initial integration objectives is to connect or fold up the entire device in about 1 hour. For this purpose, all electro-technical equipment is embedded to realise a compact and an autonomous system. In particular the head wagon integrates the vacuum pump group, the heating and temperature regulation components while the second one includes the robot and process drivers. The cask is also equipped with a double valve that allows disconnection of the vessel without loss of the vacuum condition (L. Gargiulo et al., 2006, 2007).
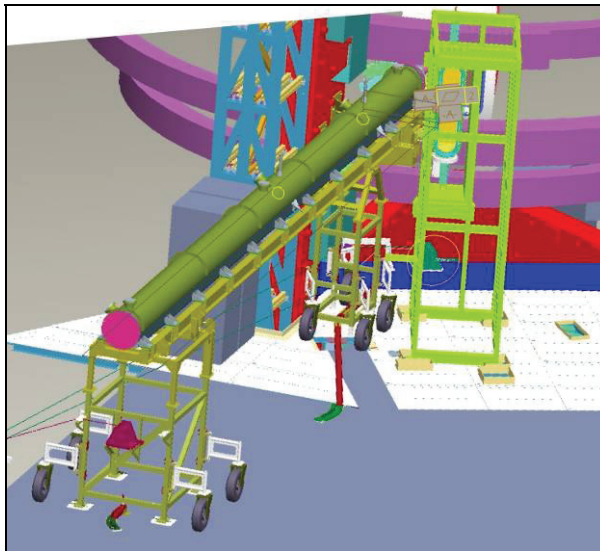
Fig. 8. Schematic view of the AIA integration on Tore Supra (side view)

2.5 Video process design

The initial objective of the project is to demonstrate the feasibility of close inspection task. Therefore, the first process developed and integrated in 2007 on the AIA robot is a video process. It was designed and assembled by ECA/HYTEC.

Principal characteristics:

- CCD color sensor with zoom and LEDs lights
- gas cooled system (temperature below 60°)
- 3 degrees of freedom (1 body rotation + 2 camera rotations)

The video process is designed with a fixed CCD camera embedded in a tight box made in stainless steel with a bright coating. This box is linked to the head of the robot through a vertical joint actuated from inside with the same system as the yaw joint of the robot. All the components and more particularly the CCD sensor located inside this box are actively cooled by the means of a small diameter flexible umbilical.

The AIA is designed to allow accurate displacements of the head, close to the first wall. A water loop leak testing process could be performed under dry nitrogen atmosphere. It will use a specific sensor able to sniff helium placed at the head of the AIA carrier or positioned at the extremity of a 20m sniffer umbilical.

3. The AIA RHE prototype and proof of feasibility

3.1 Vacuum and temperature technologies – Experimental measurements

As the robot is dedicated to be introduced in a Tokamak without breaking the vacuum, several technologies were selected to be used under ultra high vacuum:

- The structure materials in metallic alloys, like titanium.
- Some other non organic materials could also be used like Vespel and Viton.
- No organic materials.
- Use of welding for assembly of the structure parts.
- Gearbox will use standard reducers. Roller screw and gearbox should be lubricated and embedded in tight sealed boxes with the motors.
- Use of needle bearings with dry lubricant but significant regular maintenance will be required.
- Electronics will be embedded in tight boxes with tight connectors and linked by flexible tubes. (Fig. 9)

Fig. 9. Rotation tight boxes embedded on the Robotic Equipment

In parallel of the Robotic Equipment design, tests campaigns, components characterizations under temperature and, if needed, specific developments have been performed at CEA-LIST laboratory facilities. For instance, no manufacturer's standards actuators cope with the vacuum and temperature requirements. Specific temperature hardened motors and gold coated electronic boards using HCMOS military electronic boards have been developed and tested.

Fig. 10. Temperature tests on motors
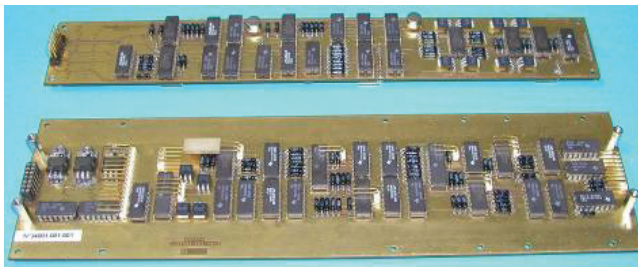
Fig. 11. Gold coated electronic boards qualified under temperature (baking 200°C and 120°C in use)

3.2 First prototype module

Since overall vacuum and temperature technologies are selected and qualified, the prototype module can be manufactured and tested.

In 2004, a vacuum and temperature test campaign was performed in CEA DRFC test facility ME60. This test has shown a good functioning under ITER representative conditions 120°C temperature and vacuum. The baking of the robot at 200°C was performed during one week and the final spectrum has shown a good behaviour of the system (Fig. 12).

These first results give confidence in the technologies developed for the AIA to be used in a Tokamak conditioning. An endurance testing was also performed at room conditions to qualify the module performances under representative payload (Fig.13).
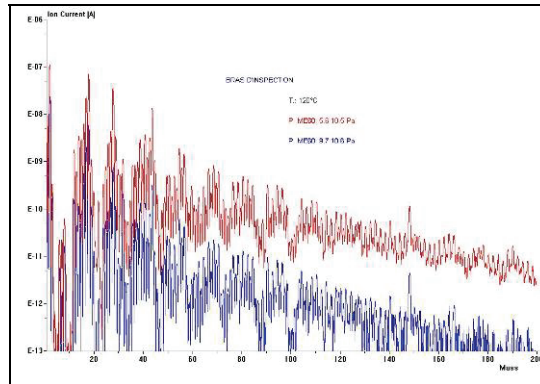
Fig. 12. Final outgassing spectrum before and after baking

The final pressure obtained at the end of the test campaign was 9.7 10-6 Pa which are good results to use the AIA robot in a Tokamak vessel.

Fig. 13. Endurance tests on the prototype module in CEA-LIST facilities

The experience collected during these tests campaigns pointed out the necessity to upgrade some mechanical parts toward a new prototype design. On that base, this module is considered as the first of series constituting the whole robot.

3.2 First prototype module qualification under Vacuum & Temperature

The February 2007 test campaign was a significant milestone of the project as it represents the final design of the prototype module qualification under real operating conditions in ME60 facility (Fig. 14):

- Ultra High Vacuum (10^{-6} Pa)
- Cycles between 200°C for outgassing, 120°C and 20°C.

The successful results of this test campaign qualify the entire prototype module and also all the components and technologies developed since the beginning of the project to cope with the severe operating requirements: Ultra High Vacuum (10-6 Pa) and temperature (200°C for outgassing, 120°C in use).
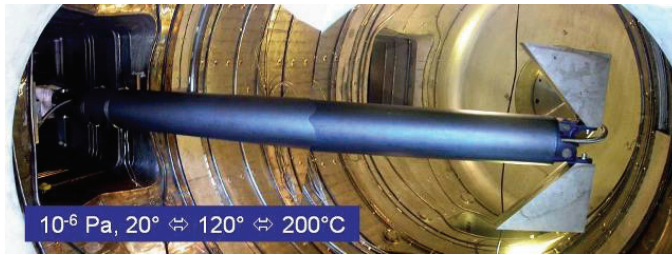
Fig. 14. Test campaign on the ITER relevant module in real functioning conditions (February 2007)

3.3 Complete assembly of the Robotic Equipment and preliminary tests

The full AIA robot manufacture was based on the same design than the upgraded module. All these parts were delivered before the end of 2006 and were integrated beginning 2007. The command control of the robot is deployed on the system and tuned.

Fig. 15. Robotic Equipment preliminary tests at CEA-LIST robotics laboratory facilities

The video process assembly is achieved in June 2007 and integrated on the AIA Robotic Equipment (Fig.16).

In August 2007, the complete AIA robotic equipment is achieved and tested at CEA – LIST Robotics laboratory facilities (Fig. 17) under ambient pressure and temperature conditions for:

- Robot mechanical and electronic functional validation,
- Command control tuning and validation,
- Robot integration on the deployment system and functional validation,
- Video process integration and functional validation
- Performances measurements,
- AIA robot qualification under atmospheric conditions in free environment,
- Definition of deployment trajectories in free environment
- Tuning of security modes
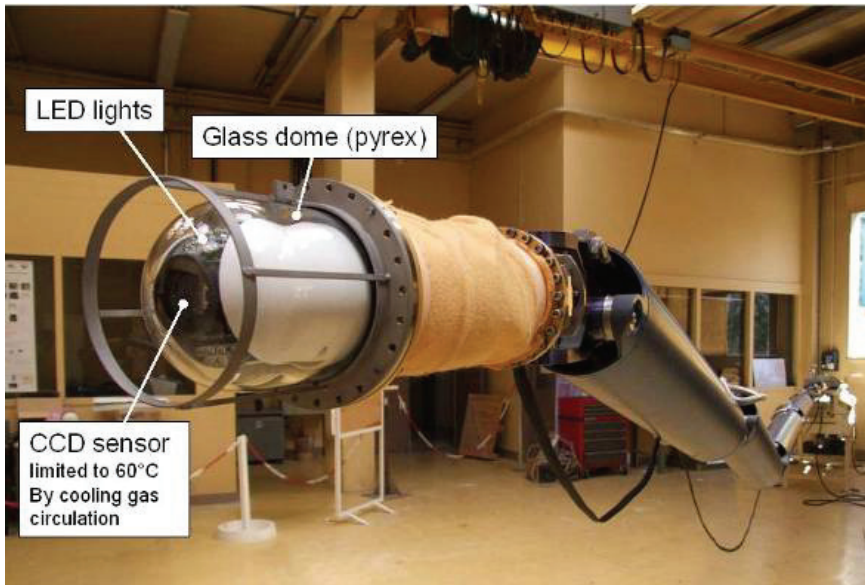
Fig. 16. Video process integration on the AIA robot carrier
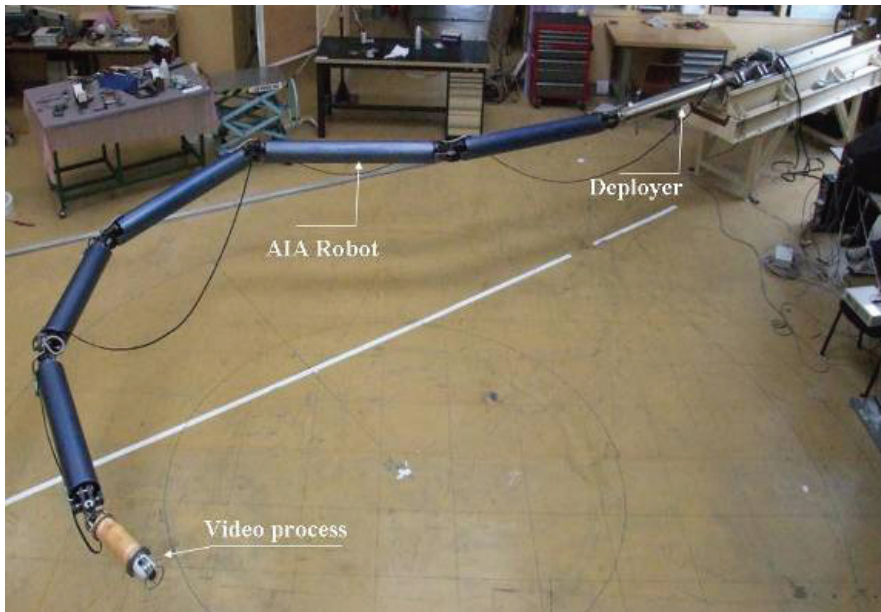
Fig. 17. AIA robot and Deployer tuning in CEA-LIST laboratories.

In the meantime, the Tokamak Equipment is assembled and tested at CEA-DRFC facilities for the functional validation and its integration on the Tokamak.

Qualification tests of the guiding system and leak tests are performed under high temperature. Connection to Tore Supra is validated (Fig. 18).
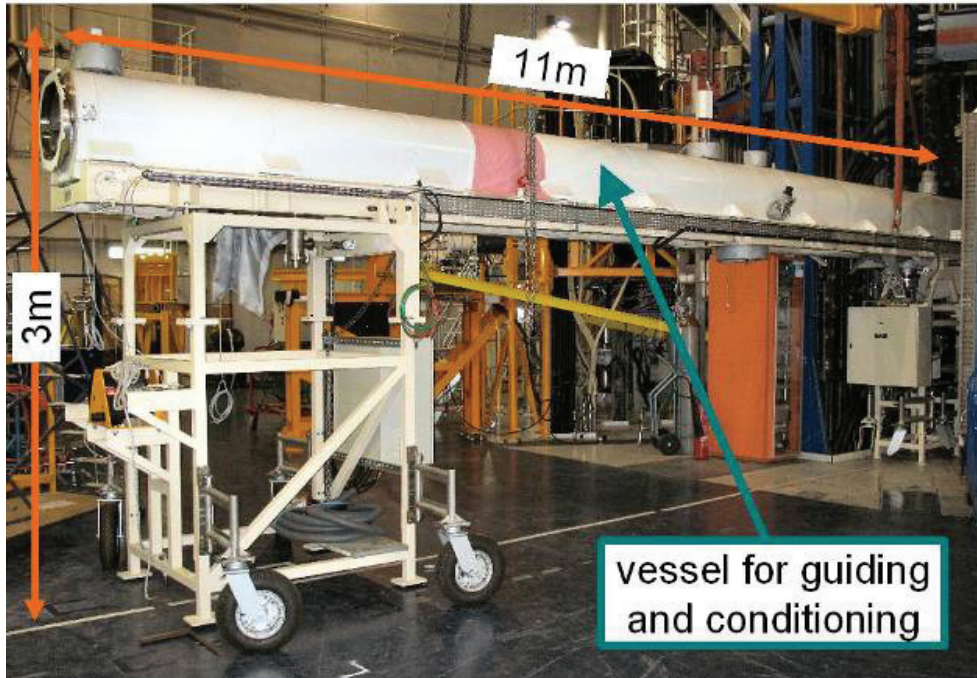
Fig. 18. Tokamak Equipment integration on Tore Supra

3.4 Tests on a Tore Supra scale one mock up

Qualification campaigns with a deployment on the Tore Supra scale 1 mock-up of the port and slices of the vessel (Fig. 19) were carried on for:

- AIA robot qualification rehearsal under atmospheric conditions in free environment
- Deployment trajectories rehearsal under atmospheric conditions in free environment (as defined in CEA-LIST labs)
- Deployment trajectories rehearsal under atmospheric conditions on the scale one mock up for :
  - trajectory validation,
  - distance measurements between the robot and the Vacuum Vessel mock up
  - Deployment scenario validation before the robot introduction in Tore Supra.
- Security modes rehearsal and validation of safety procedures.
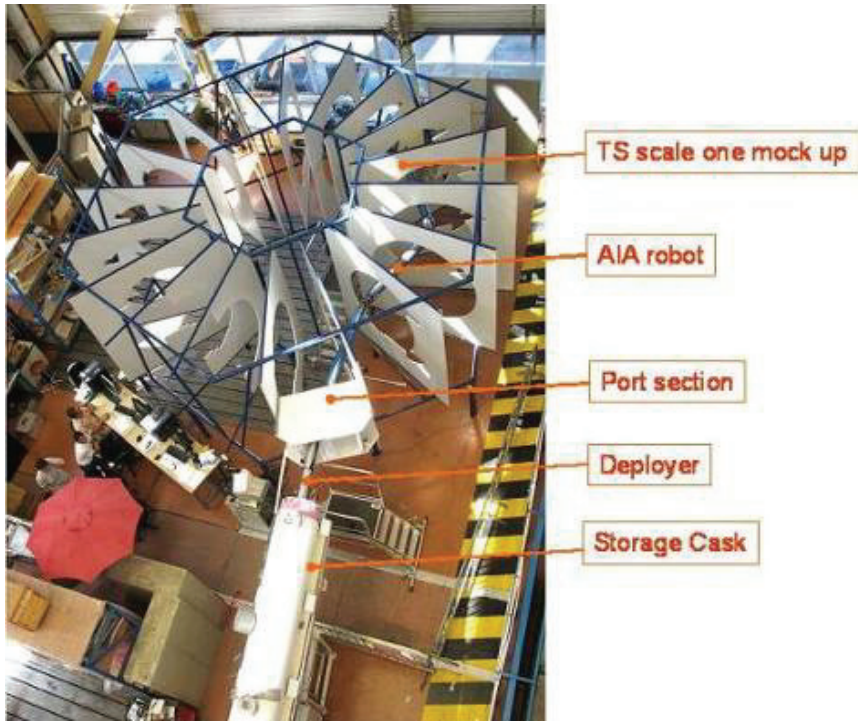
Fig. 19. AIA introduction scenario rehearsal in Tore Supra mock up

3.5 Deployment on Tore Supra under atmospheric conditions – Proof of feasibility

In September,12th 2007, the Remote Handling Equipment is connected on Tore Supra and deployed into the machine by following the same scenario as rehearsed on the mock up (Fig. 20).

Fig. 20. AIA deployment scenario rehearsal in TORE SUPRA under atmospheric conditions

The tests enabled to validate the operational performances of visual inspection process. Close inspection of the machine first wall components gives sufficient quality images to accurately analyze the state of the installation. These tests bring confidence to draw sustainable analysis of the inner surface state and diagnostic of possible events. The test has still to be rehearsed under real operating conditions. Thus, the robot reliability and controllability must be enhanced to have confidence in close teleoperation trajectories.

3. Development to maturity toward In-Service use on Tore Supra

This RHE tool from its prototypical nature has still to prove its reliability, robustness and controllability for its In-Service use. Therefore, several tests campaigns are planned on the Tore Supra scale one mock up and ME60 test bed facility to collect knowledge on the system behaviour. Some mechanical upgrades could be necessary to enhance the RHE reliability.

Also, due to its large structure and flexibilities the accuracy of the AIA robot is perfectible. Therefore, a calibration method of the flexible model taking into consideration the flexibilities of the structure and geometric imperfections could improve the robot performances (J. Chalfoun et al., 2007). Once calibrated, the robot model could be computed in real time by the controller in order to locate the robot position in the 3D environment.

Limited access of viewing in the Vacuum Vessel requires on line monitoring of the robot. Next developments concern a teleoperation control system enabling steering thanks to a passive 3D device (space mouse), a 3D graphical interface for 3D realistic display of the manipulator. The robot position in the 3D environment is computed by the real time flexible model. On line collision avoidance and real time dynamic simulation are functionalities provided by the graphical interface tool.

The on line monitoring could also be used for fault detection while it could detect robot deviation or component fault in real time. This steering interface will provide all the functions necessary to remotely control manipulators. (Fig. 21).
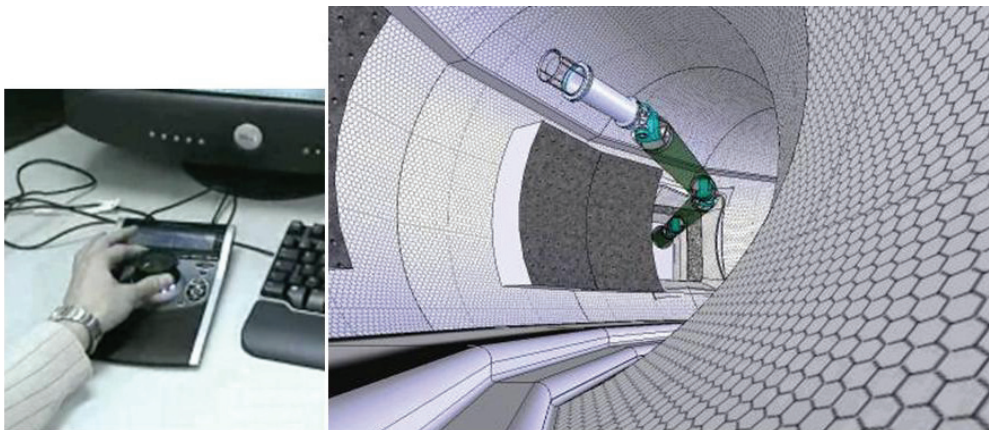
Fig. 21. AIA's control through on line monitoring system

Several tools to be installed on the AIA for various in vessel operations are being studied. In particular water loop leak testing, laser ablation for wall detritiation and surface characterization are foreseen as utilities to be placed at the AIA end effector. All these various systems are currently in development in different laboratories of the CEA.

4. Conclusion

The first lessons learned on the preliminary results on testing a scale one RHE prototype in a scale one in service Tokamak facility is extremely important to show real results on Remote Handling which stands for 50% robotics and 50% tokamak integration technology. The demonstration on Tore Supra helps in the understanding of operation issues that could occur in the tokamak vacuum vessel equipped with actively cooled components.

Tore Supra operates with similar vacuum and temperature conditions as ITER (120°C to 200°C). Integration and rehearsal operation of the AIA demonstrator in this tokamak facility will give essential results when considering in-vessel inspection routine capabilities and reliability of the system.

In-Service use of a RHE as a routine inspection tool on a Tokamak will provide a lot of lessons very precious to follow up the demand for high performance remote maintenance required to operate the next Tokamaks expected to support the future Fusion experiments.

5. Acknowledgements

6. References

A.C. Rolfe et al. The assessment and improvement of JET remote handling equipment availability. 19th Symposium on Fusion Technology. Lisbon Portugal. 1996

O. David & al, Operational Experience Feedback in JET Remote Handling **,** 21th Symposium on Fusion Technology, Madrid

Y. Perrot et al., Scale One Field Test of a Long Reach Articulated Carrier for Inspection in Spent Fuel Management Facilities, 10th ANS, 28-31 March 2004, Florida (USA).

Y. Perrot et al., The articulated inspection arm for ITER, design of the demonstration in tore supra, 23rd SOFT, 20-24 September 2004, Venice (Italy).

L. Gargiulo et al., Towards operations on Tore Supra of an ITER relevant inspection robot and associated processes, 24th SOFT, 11-15 September 2006, Warsaw (Poland).

L. Gargiulo et al., Development of an ITER relevant inspection robot, ISFNT8, September, 30th - October, 5th 2007, Heidelberg (Germany).

J. Chalfoun et al., Design and flexible modelling of a long reach articulated carrier for inspection, IROS 2007, Oct. 29-Nov. 2 2007, San Diego (USA).

Local and Global Isotropy Analysis of Mobile Robots with Three Active Caster Wheels

Sungbok Kim and Sanghyup Lee
Department of Digital Information Engineering
Hankuk University of Foreign Studies
Korea

1. Introduction

In the near future, personal service robots are expected to come into human daily life as supporters in education, leisure, house care, health care, and so on. Most of them are built on an omnidirectional mobile robot so as to navigate in an environment restricted in space and cluttered with obstacles. Caster wheels have been chosen for the development of an omnidirectional mobile robot, as reported in (Holmberg, 2000), among several omnidirectional wheel mechanisms, including universal wheels, Swedish wheels, orthogonal wheels, and ball wheels. This is because caster wheels can operate without additional peripheral rollers or support structure, and maintain good performance even though payload or ground condition changes.

There have been several works on the kinematics of a caster wheeled omnidirectional mobile robot (COMR), including the kinematic modeling, the design and control, the isotropy analysis, as reported in (Holmberg, 2000; Muir & Neuman, 1987; Campion et al., *1996*; Kim & Kim, 2004; Park et al., 2002; Kim & Moon, 2005; Oetomo et al., 2005; Kim & Jung, 2007). Previous isotropy analysis, as reported in (Kim & Kim, 2004; Park et al., 2002; Kim & Moon, 2005; Oetomo et al., 2005), has been made only for a COMR in which the steering link offset is equal to the wheel radius, except our recent work, as reported in (Kim & Jung, 2007). It was found that such a restriction is necessary to have globally optimal isotropic characteristics of a COMR, as reported in (Park et al., 2002; Kim & Moon, 2005; Oetomo et al., 2005). Nevertheless, many practical COMR's in use take advantage of the steering link offset which is different from the wheel radius, mainly for improved tipover stability, as reported in (McGhee & Frank, 1968; Papadopoulos & Rey, 1996). The tipover stability becomes critical when a COMR makes a rapid turn or sudden external forces are applied to a COMR.

The accuracy of the velocity kinematics of a robotic system depends on the condition number of the Jacobian matrix involved. Based on the Euclidean norm, the condition number of a matrix can be defined as the ratio of the largest to smallest singular values of the matrix, as reported in (Strang, 1988), whose value ranges from unity to infinity. For a given linear system $\mathbf{Ax} = \mathbf{b}$, the condition number of $\mathbf{A}$ represents the amplification of the

relative error of $\mathbf{x}$, given by $\|\delta\mathbf{x}\|/\|\mathbf{x}\|$, with respect to the relative error of $\mathbf{b}$, given by $\|\delta\mathbf{b}\|/\|\mathbf{b}\|$. Smaller condition numbers are preferred to larger condition numbers with regard to error amplification in solving $\mathbf{Ax} = \mathbf{b}$. Furthermore, the condition number equal to unity is the best situation that can be achieved when a robotic system is said to be in isotropy.

Since the Jacobian matrix of a COMR is a function of caster wheel configurations, the condition number is subject to change during task execution. For reduction in aforementioned error amplification, it is important to prevent a COMR away from the isotropy or keep a COMR close to the isotropy, as much as possible. In the light of this, this paper aims to investigate the local and global isotropic characteristics of a COMR. The isotropic configurations of a COMR which can be identified through the local isotropy analysis can be used as a reference in trajectory planning to avoid excessive error amplification throughout task execution. On the other hand, the design parameter of a COMR, such as wheel radius and steering link offset, can be optimized for the global isotropic characteristics by minimizing the averaged value of the condition number over the whole configuration space.

The purpose of this paper is to present both local and global isotropy analysis of a fully actuated COMR with the steering link offset different from the wheel radius. This paper is organized as follows. Based on the kinematic model, Section 2 derives the necessary and sufficient conditions for the isotropy of a COMR. Section 3 identifies four different sets of all possible isotropic configurations, along with the isotropic steering link offsets and the isotropic characteristic lengths. Using the local isotropy index, Section 4 examines the number of the isotropic configurations and the role of the isotropic characteristic length. Using global isotropy index, Section 5 determines the optimal characteristic length and the optimal steering link offset for maximal global isotropy. Finally, the conclusion is made in Section 6.

2. Isotropy conditions

Consider a COMR with three identical caster wheels attached to a regular triangular platform moving on the xy–plane, as shown in Fig. 1. Let l be the side length of the platform; let $d(\geq 0)$ and $r(> 0)$ be the steering link offset and the wheel radius, respectively; let φ_i and θ_i be the steering and the rotating angles, respectively; let $\mathbf{u}_i$ and $\mathbf{v}_i$ be the orthogonal unit vectors along the steering link and the wheel axis, respectively, such that $\mathbf{u}_i = [-\cos\varphi_i \ -\sin\varphi_i]^t$ and $\mathbf{v}_i = [-\sin\varphi_i \ \cos\varphi_i]^t$; let $\mathbf{p}_i$ be the vector from the center of the platform to the center of the wheel, and $\mathbf{q}_i$ be the rotation of $\mathbf{p}_i$ by $90°$ counterclockwise. For each wheel, it is assumed that the steering link offset can be different from the wheel radius, that is, $d \neq r$.

With the introduction of the characteristic length, as reported in (Strang, 1988), $L(> 0)$, the kinematic model of a COMR under full actuation is obtained by

$$\mathbf{A}\dot{x} = \mathbf{B}\dot{\Theta} \qquad\qquad (1)$$

where $\dot{\mathbf{x}} = [\mathbf{v}\ L\omega]^t \in \mathbf{R}^{3\times1}$ is the task velocity vector, and $\dot{\Theta} = [\dot{\theta}_1\ \dot{\theta}_2\ \dot{\theta}_3\ \dot{\varphi}_1\ \dot{\varphi}_2\ \dot{\varphi}_3]^t \in \mathbf{R}^{6\times1}$ is the joint velocity vector, and

$$\mathbf{A} = \begin{bmatrix} \mathbf{u}_1{}^t & \dfrac{1}{L}\ \mathbf{u}_1{}^t & \mathbf{q}_1 \\[4pt] \mathbf{u}_2{}^t & \dfrac{1}{L}\ \mathbf{u}_2{}^t & \mathbf{q}_2 \\[4pt] \mathbf{u}_3{}^t & \dfrac{1}{L}\ \mathbf{u}_3{}^t & \mathbf{q}_3 \\[4pt] \mathbf{v}_1{}^t & \dfrac{1}{L}\ \mathbf{v}_1{}^t & \mathbf{q}_1 \\[4pt] \mathbf{v}_2{}^t & \dfrac{1}{L}\ \mathbf{v}_2{}^t & \mathbf{q}_2 \\[4pt] \mathbf{v}_3{}^t & \dfrac{1}{L}\ \mathbf{v}_3{}^t & \mathbf{q}_3 \end{bmatrix} \in \mathbf{R}^{6\times3} \tag{2}$$

$$\mathbf{B} = \begin{bmatrix} r\mathbf{I}_3 & 0_3 \\ 0_3 & d\mathbf{I}_3 \end{bmatrix} \in \mathbf{R}^{6\times6} \tag{3}$$

are the Jacobian matrices. Notice that the introduction of L makes all three columns of $\mathbf{A}$ to be consistent in physical unit.
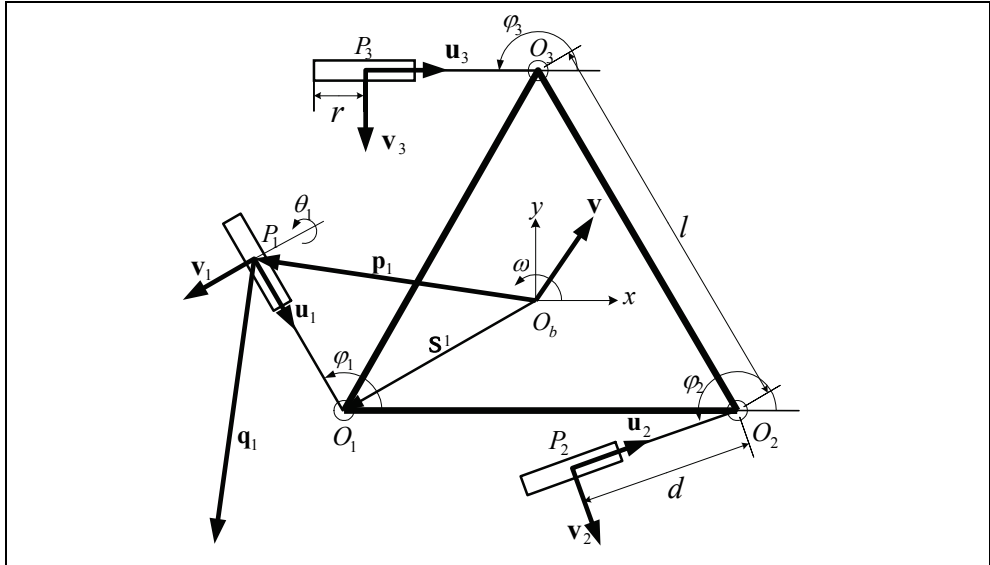
Fig. 1. A caster wheeled omnidirectional mobile robot

Now, from (1), the necessary and sufficient condition for the kinematic isotropy of a COMR can be expressed as

$$\mathbf{Z}^t\mathbf{Z} = \sigma\mathbf{I}_3 \tag{4}$$

where

$$\mathbf{Z} = \mathbf{B}^{-1}\mathbf{A} \tag{5}$$

$$\sigma = \frac{3}{2}\left(\frac{1}{r^2} + \frac{1}{d^2}\right) \tag{6}$$

Using (2), (3), (5), and (6), from (4), the three isotropy conditions of a COMR can be obtained as follows:

$$\sum_{i=1}^{3}[\,\mu\,(\,\mathbf{u}_i\,\mathbf{u}_i^{\,t}\,) + (\,\mathbf{v}_i\,\mathbf{v}_i^{\,t}\,)\,] = \frac{3}{2}(\,\mu+1)\,\mathbf{I}_2 \tag{7}$$

$$\sum_{i=1}^{3}[\,\mu\,(\,\mathbf{u}_i^{\,t}\,\mathbf{q}_i\,)\,\mathbf{u}_i + (\,\mathbf{v}_i^{\,t}\,\mathbf{q}_i\,)\,\mathbf{v}_i\,] = \mathbf{0} \tag{8}$$

$$\frac{1}{L^2}\sum_{i=1}^{3}[\,\mu\,(\,\mathbf{u}_i^{\,t}\,\mathbf{q}_i\,)^2 + (\,\mathbf{v}_i^{\,t}\,\mathbf{q}_i\,)^2\,] = \frac{3}{2}(\,\mu+1) \tag{9}$$

where

$$\mu = \left(\frac{d}{r}\right)^2 > 0 \tag{10}$$

which represents the square of the ratio of the steering link offset d to the wheel radius r. Note that $\mu = 1$ corresponds to the case of the steering link offset d equal to the wheel radius r, as reported in (Kim & Kim, 2004).

3. Isotropic configurations

The first and the second isotropy conditions, given by (7) and (8), are a function of the steering joint angles, $(\varphi_1, \varphi_2, \varphi_3)$, from which the *isotropic configurations*, denoted by Θ_{iso}, can be identified. For a given wheel radius r, the specific value of steering link offset, called as the *isotropic steering link offset*, d_{iso}, is required for the isotropy of a COMR. With the isotropic configuration known, the third isotropy condition, given by (9), determines the specific value of the characteristic length, called the *isotropic characteristic length, L_{iso}*, is required for the isotropy of a COMR. The detailed procedure to obtain the isotropic configurations Θ_{iso}, the isotropic steering link offset d_{iso}, and the isotropic characteristic length L_{iso} can be found our previous work, as reported in (Kim & Jung, 2007).

All possible isotropic configurations Θ_{iso} of a COMR can be categorized into four different sets according to the restriction imposed on the isotropic steering link offset d_{iso}, for a given

wheel radius r. Table 1 lists four different sets of Θ_{iso}, denoted by S1, S2, S3, and S4, and the corresponding value of d_{iso}.

Set	Θ_{iso}	d_{iso}	L_{iso}
S1	$\varphi_1,$ $\varphi_1 + \dfrac{2\pi}{3},$ $\varphi_1 - \dfrac{2\pi}{3}$	No restriction $\mu = \left(\dfrac{d}{r}\right)^2$	$\sqrt{\dfrac{2}{\mu+1}\left\{\left(\dfrac{1}{\sqrt{3}}\sin(\varphi_1 - \dfrac{\pi}{6})\right)^2 \mu + \left(d - \dfrac{1}{\sqrt{3}}\cos(\varphi_1 - \dfrac{\pi}{6})\right)^2\right\}}$
S2	$\varphi_1,$ $\varphi_1 - \dfrac{2\pi}{3},$ $\varphi_1 + \dfrac{2\pi}{3}$	r	$\sqrt{d_{iso}{}^2 + \dfrac{1}{3}}$
S3	$\left(\dfrac{\pi}{6}, \dfrac{\pi}{2}, -\dfrac{\pi}{6}\right),$ $\left(-\dfrac{\pi}{6}, -\dfrac{5\pi}{6}, -\dfrac{\pi}{2}\right),$ $\left(\dfrac{\pi}{2}, \dfrac{5\pi}{6}, -\dfrac{5\pi}{6}\right)$	$\sqrt{r^2 + \dfrac{4}{3}r^4 - \dfrac{2}{\sqrt{3}}r^2}$	$\dfrac{1}{\sqrt{1 - \dfrac{2}{\sqrt{3}}d_{\mathrm{iso}}}}\left(\dfrac{1}{\sqrt{3}} - d_{\mathrm{iso}}\right)$
S4	$\left(-\dfrac{5\pi}{6}, -\dfrac{\pi}{2}, \dfrac{5\pi}{6}\right),$ $\left(\dfrac{5\pi}{6}, \dfrac{\pi}{6}, \dfrac{\pi}{2}\right),$ $\left(-\dfrac{\pi}{2}, -\dfrac{\pi}{6}, \dfrac{\pi}{6}\right)$	$\sqrt{r^2 + \dfrac{4}{3}r^4 + \dfrac{2}{\sqrt{3}}r^2}$	$\dfrac{1}{\sqrt{1 + \dfrac{2}{\sqrt{3}}d_{\mathrm{iso}}}}\left(\dfrac{1}{\sqrt{3}} + d_{\mathrm{iso}}\right)$

Table 1. Four different sets of all isotropic configurations

It should be noted that S1 places no restriction on d_{iso}, unlike the other three sets, S2, S3, and S4. Fig. 2 illustrates four different sets of Θ_{iso}, characterized by the tuple of $(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3)$.

It is interesting to observe that there exist certain geometrical symmetries among four sets: the symmetry between S1 and S2, shown in Fig. 2a) and 2b), and the symmetry between S3 and S4, shown in Fig. 2c) and 2d).

Once the isotropic configuration has been identified under the conditions of (7) and (8), the isotropic characteristic length L_{iso} can be determined under the condition of (9). For four different sets of the isotropic configurations, the expression of L_{iso} can be elaborated as

listed in Table 1. Note that the isotropy of a COMR cannot be achieved unless the characteristic length is chosen as the isotropic characteristic length, that is, $L = L_{\text{iso}}$.
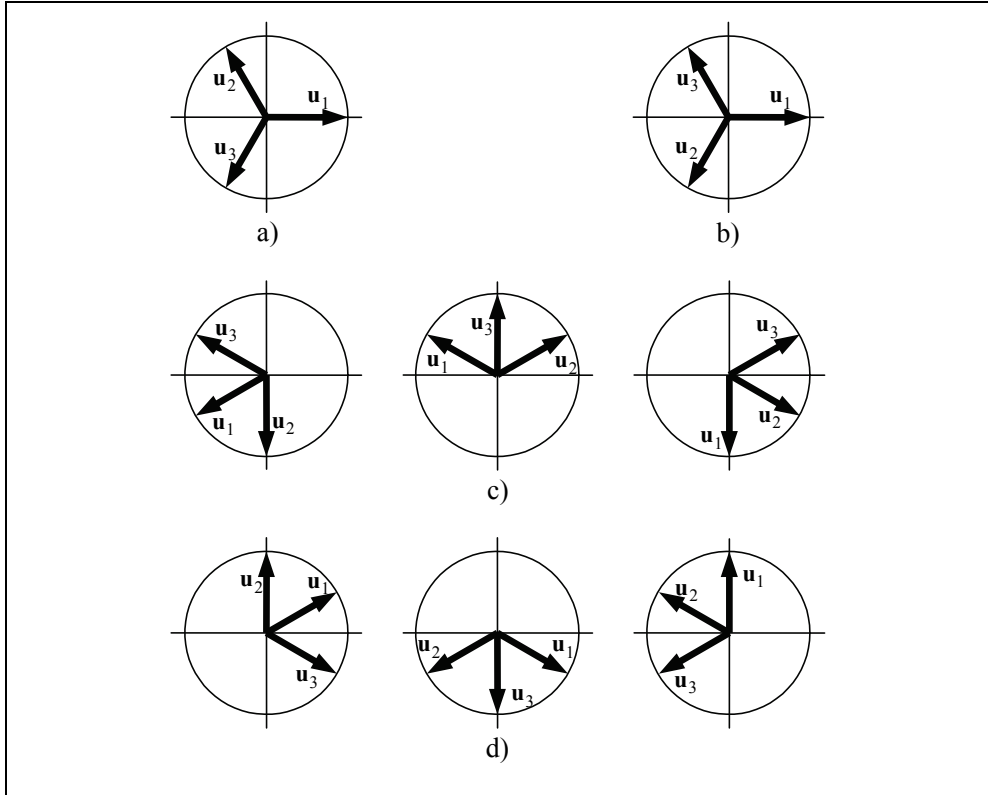
Fig. 2. Four different sets of the isotropic configurations: a) S1 with $\varphi_1 = \pi$, b) S2 with $\varphi_1 = \pi$, c) S3 and d) S4

4. Local isotropy analysis

Let $\lambda_i, i = 1, 2, 3$, be the eigenvalues of $\mathbf{Z}^t \mathbf{Z}$, whose square roots are the same as the singular values of $\mathbf{Z}$. We define the local isotropy index of a COMR, denoted by σ, as

$$0.0 \leq \sigma = \sqrt{\frac{\min \lambda_i}{\max \lambda_i}} \leq 1.0 \tag{11}$$

whose value ranges between 0 and 1. Note that the local isotropy index σ is the inverse of the well-known condition number of $\mathbf{Z}$. In general, σ is a function of the wheel configuration $\mathbf{\Theta} = (\varphi_1, \varphi_2, \varphi_3)$, the characteristic length L, the wheel radius r, and the steering link offset d:

$$\sigma = \sigma\left(\Theta, L, r, d\right) \qquad (12)$$

To examine the isotropic characteristics of a COMR, extensive simulation has been performed for various combinations of characteristic length L, the wheel radius r, and the steering link offset d. However, we present only the simulation results obtained from two different situations, denoted by SM1 and SM3, for which the values of the key parameters, including r, d_{iso}, Θ_{iso}, and L_{iso}, are listed in Table 2. Note that all the values of r, d, and L represent the relative scales to the platform side length l, which is assumed to be unity, that is, $l = 1.0[m]$.

Situation	r	d_{iso}	Θ_{iso}	L_{iso}
SM1	0.2	0.2	$\left(\dfrac{\pi}{6}, \dfrac{5\pi}{6}, -\dfrac{\pi}{2}\right)$	0.3774
SM3	0.2	0.1591	$\left(\dfrac{\pi}{6}, \dfrac{\pi}{2}, -\dfrac{\pi}{6}\right)$	0.4629

Table 2. Simulation environment

First, let us examine how the value of the local isotropy index σ changes over the entire configuration space $\Theta = (\varphi_1, \varphi_2, \varphi_3)$. With the values of r, d_{iso}, and L_{iso} given in Table 2, Fig. 3 shows the plots of $\sigma\,(\varphi_1 = \pi/6)$ for $-\pi \le \varphi_2, \varphi_3 \le \pi$ in the cases of SM1 and SM3.
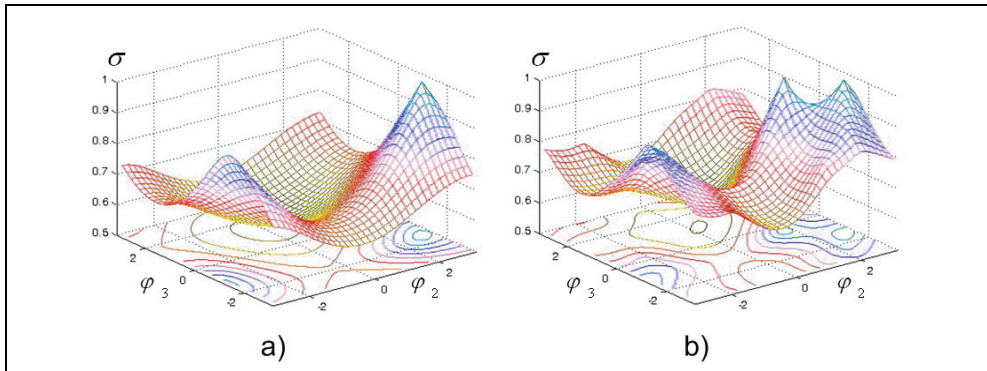
Fig. 3. The plots of $\sigma\,(\varphi_1 = \pi/6)$ for $-\pi \le \varphi_2, \varphi_3 \le \pi$: a) SM1 and b) SM3

The ranges of σ are obtained as $0.5336 \le \sigma\,(\varphi_1 = \pi/6) \le 1.0$ for SM1 and $0.5972 \le \sigma\,(\varphi_1 = \pi/6) \le 1.0$ for SM3. For both cases, it can be observed that the value of σ changes significantly depending on the wheel configurations and also that the isotropic configurations with $\sigma = 1.0$ appear as the result of $d = d_{iso}$ and $L = L_{iso}$. Note that SM1 has a single isotropic configuration, $(\pi/6, 5\pi/6, -\pi/2)$ which belongs to S1, whereas SM3 has

two isotropic configurations: $(\pi/6, \pi/2, -\pi/6)$ which belongs to S3 and $(\pi/6, 5\pi/6, -\pi/2)$ which belongs to both S1 and S3.

Next, for a given isotropic configuration, let us examine how the choice of the characteristic length L affects the values of the local isotropy index σ. With the values of r, d_{iso} and Θ_{iso} given in Table 2, Fig. 4 shows the plots of $\sigma(\Theta_{\text{iso}})$ for $0 < L \leq 1.0$, in the cases of SM1 and SM3. For both cases, it can be observed that the value of σ decreases significantly as the choice of L is away from the isotropic characteristic length L_{iso}: $L_{\text{iso}} = 0.377$ for SM1, and $L_{\text{iso}} = 0.463$ for SM3. This demonstrates the importance of $L = L_{\text{iso}}$ for the isotropy of a COMR.
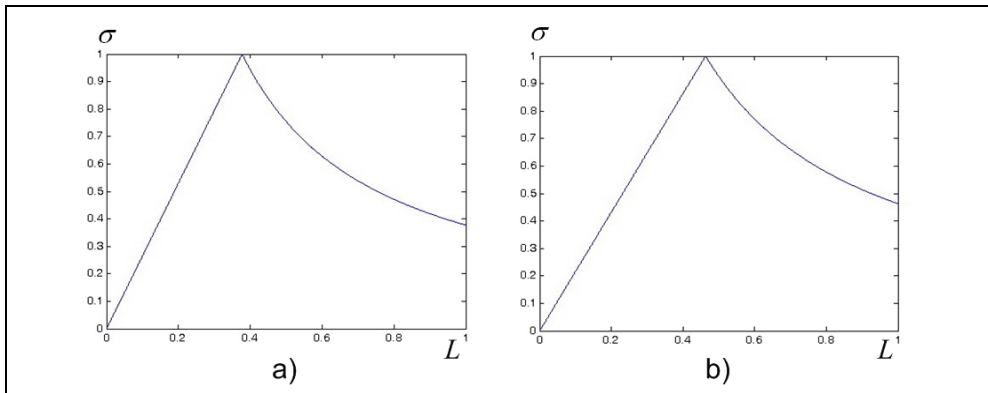
Fig. 4. The plots of $\sigma(\Theta_{\text{iso}})$ for $0 < L \leq 1.0$: a) SM1 and b) SM3

5. Global isotropy analysis

The local isotropy index represents the local isotropic characteristics of a COMR at a specific instance of the wheel configurations. To characterize the global isotropic characteristics of a COMR, we define the global isotropy index of a COMR, denoted by $\bar{\sigma}$, as the average of the local isotropy index σ taken over the entire configuration space, $-\pi \leq \varphi_1, \varphi_2, \varphi_3 \leq \pi$. Now, $\bar{\sigma}$ is a function of the characteristic length L, the wheel radius r, and the steering link offset d:

$$\bar{\sigma} = \bar{\sigma}(L, r, d) \qquad (13)$$

Let us examine how the choice of the characteristic length L affects the values of the global isotropy index $\bar{\sigma}$. With the values of r and d_{iso} given in Table 2, Fig. 5 shows the plots of $\bar{\sigma}$ for $0 < L \leq 1.0$, in the cases of SM1 and SM3. For both cases, it can be observed that the value of $\bar{\sigma}$ reaches its maximum, which is called as the *optimal global isotropy index*, $\bar{\sigma}_{\max}$, at the specific value of L, which is called as the *optimal characteristic length*, L_{opt}: $\bar{\sigma}_{\max} = 0.8017$ at $L_{\text{opt}} = 0.614$ for SM1, and $\bar{\sigma}_{\max} = 0.7538$ at $L_{\text{opt}} = 0.588$ for SM3.
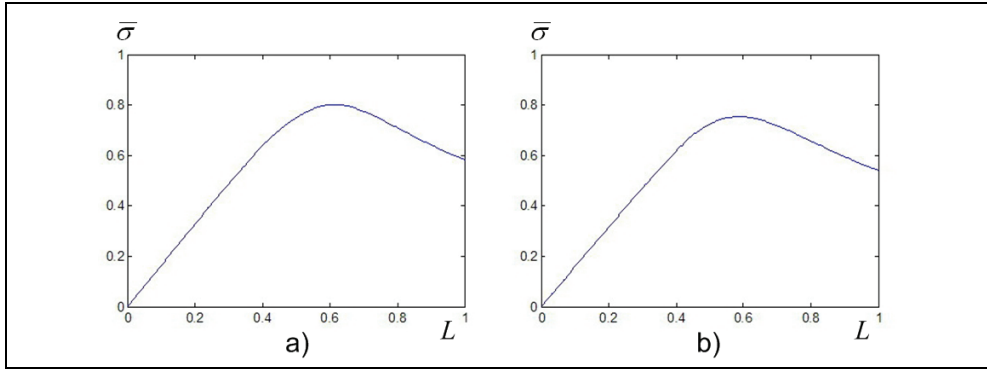
Fig. 5. The plots of $\overline{\sigma}$ for $0 < L \leq 1.0$: a) SM1 and b) SM3

Next, let us examine how the ratio of the steering link offset d to the wheel radius r affects the values of the optimal global isotropy index $\overline{\sigma}_{\max}$ and the corresponding optimal characteristic length L_{opt}. With the value of r given in Table 2, Fig. 6 shows the plots of $\overline{\sigma}_{\max}$ and L_{opt} for $0 < d \leq 0.3$ The ranges of $\overline{\sigma}_{\max}$ and L_{opt} are obtained as $0.2760 \leq \overline{\sigma}_{\max} \leq 0.8016$ and $0.58 \leq L_{\text{opt}} \leq 0.64$. It can be observed that the optimal value of d is found to be 0.2 so that $d/r = 1.0$, which results in $\overline{\sigma}_{\max} = 0.8016$ at $L_{\text{opt}} = 0.62$.
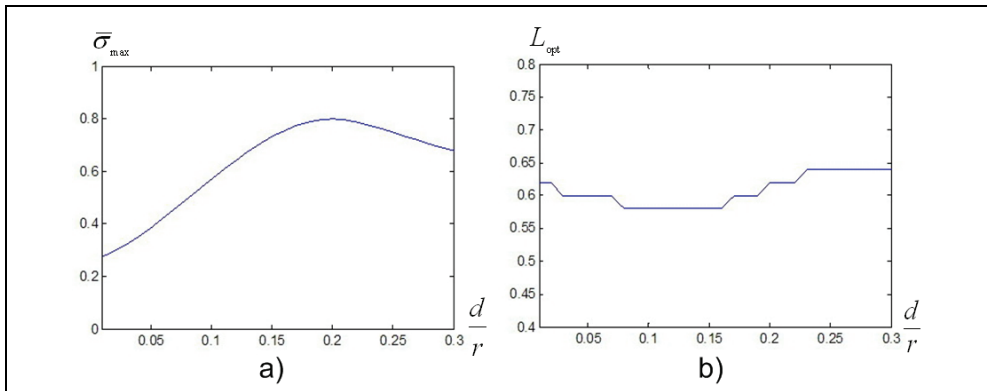
Fig. 6. The plots of a) $\overline{\sigma}_{\max}$ and b) L_{opt}, for $0 < d \leq 0.3$

6. Conclusion

In this paper, we presented the local and global isotropy analysis of a fully actuated caster wheeled omnidirectional mobile robot (COMR) with the steering link offset different from the wheel radius. First, based on the kinematic model, the necessary and sufficient isotropy conditions of a COMR were derived. Second, four different sets of all possible isotropic configurations were identified, along with the expressions for the isotropic steering link

offset and the isotropic characteristic length. Third, using the local isotropy index, the number of the isotropic configurations and the role of the isotropic characteristic length were examined. Fourth, using the global isotropy index, the optimal characteristic length and the optimal steering link offset were determined for maximal global isotropy.

7. Acknowledgement

8. References

Holmberg, R. (2000). *Design and Development for Powered-Caster Holonomic Mobile Robot*. Ph. D. Thesis, Dept. of Mechanical Eng., Stanford University

Muir, P. F. & Neuman, C. P. (1987). Kinematic modeling of wheeled mobile robots. *J. of Robotic Systems*, Vol. 4, No. 2, pp. 281-340

Campion, G.; Bastin, G. & Novel, B. D`Andrea. (1996). Structural properties and classification of kinematic and dynamic models of wheeled mobile robots. *IEEE Trans. Robotics and Automation*, Vol. 12, No. 1, pp. 47-62

Kim, S. & Kim, H. (2004). Isotropy analysis of caster wheeled omnidirectional mobile robot. *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 3093-3098

Park, T.; Lee, J.; Yi, B.; Kim, W.; You, B. & Oh, S. (2002). Optimal design and actuator sizing of redundantly actuated omni-directional mobile robots. *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 732-737

Kim, S. & Moon, B. (2005). Local and global isotropy of caster wheeled omnidirectional mobile robot. *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 3457-3462

Oetomo, D.; Li, Y. P.; Ang Jr., M. H. & Lim, C. W. (2005). Omnidirectional mobile robots with powered caster wheels: design guidelines from kinematic isotropy analysis. *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pp. 3034-3039

Kim, S & Jung, I. (2007). Systematic isotropy analysis of caster wheeled mobile robots with steering link offset different from wheel radius. *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 2971-2976

McGhee, R. B. & Frank, A. A. (1968). On the stability of quadruped creeping gaits. *Mathematical Biosciences*, Vol. 3, No. 3, pp. 331-351

Papadopoulos, E. G. & Rey, D. A. (1988). A new measure of tipover stability margin for mobile manipulators. *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 3111-3116

Strang, G. (1988). *Linear Algebra and Its Applications*. Saunders College Publishing

Saha, S. K.; Angeles, J. & Darcovich, J. (1995). The design of kinematically isotropic rolling robots with omnidirectional wheels. *Mechanism and Machine Theory*, Vol. 30, No. 8, pp. 1127-1137

UML-Based Service Robot Software Development: A Case Study[*]

Minseong Kim[1], Suntae Kim[1], Sooyong Park[1], Mun-Taek Choi[2],
Munsang Kim[2] and Hassan Gomaa[3]

Sogang University[1],
Center for Intelligent Robotics Frontier 21 Program
at Korea Institute of Science and Technology[2],
George Mason University[3]
Republic of Korea[1,2],
USA[3]

1. Introduction

Robots have been used in several new applications. In recent years, both academic and commercial research has been focusing on the development of a new generation of robots in the emerging field of service robots. Service robots are individually designed to perform tasks in a specific environment for working with or assisting humans and must be able to perform services semi- or fully automatically (Kawamura & Iskarous, 1994; Rofer et al., 2000). Examples of service robots are those used for inspection, maintenance, housekeeping, office automation and aiding senior citizens or physically challenged individuals (Schraft, 1994; Rofer et al., 2000). A number of commercialized service robots have recently been introduced such as vacuum cleaning robots, home security robots, robots for lawn mowing, entertainment robots, and guide robots (Rofer et al., 2000; Kim et al., 2003; You et al., 2003; Pineau et al., 2003; Kim et al., 2005).

In this context, Public Service Robot (PSR) systems have been developed for indoor service tasks at Korea Institute of Science and Technology (KIST) (Kim et al., 2003; Kim et al., 2004). The PSR is an intelligent service robot, which has various capabilities such as navigation, manipulation, etc. Up to now, three versions of the PSR systems, that is, PSR-1, PSR-2, and a guide robot Jinny have been built.

The worldwide aging population and health care costs of aged people are rapidly growing and are set to become a major problem in the coming decades. This phenomenon could lead to a huge market for service robots assisting with the care and support of the disabled and elderly in the future (Kawamura & Iskarous, 1994; Meng & Lee, 2004; Pineau et al., 2003). As a result, a new project is under development at Center for Intelligent Robotics (CIR) at KIST, i.e. the intelligent service robot for the elderly, called T-Rot.

In our service robot applications, it is essential to not only consider and develop a well-defined robot software architecture, but also to develop and integrate robot application components in a systematic and comprehensive manner. There are several reasons for this:

- First, service robots interact closely with humans in a wide range of situations for providing services through robot application components such as vision recognition, speech recognition, navigation, etc. Thus, a well-defined robot control architecture is required for coherently and systematically combining these services into an integrated system.
- Second, in robot systems, there are many-to-many relations among software components as well as hardware components. For instance, a local map module requires range data from a laser scanner, ultrasonic sensors, and infrared sensors, as well as prior geometrical descriptions of the environment. On the other hand, the laser scanner should provide its data to a path planner, localizer, and a local map building module. These relationships, as well as interactions among software or hardware modules, must be carefully analyzed and systematically managed from an early stage of development in order to understand the big picture.
- Third, the functional performance of each software and hardware module becomes highly dependent on the architecture, as the number of robot platforms increases (Kim et al., 2004), and new services are added, or existing services are removed or updated to address changes in user needs.
- Fourth, previously developed software modules like maps, localization, and path planners can be directly reused for new tasks or services by service robot developers. Thus, a robot architecture, as well as systematic processes or methods, are required to support the implementation of the system, to ensure modularity and reusability.

As a consequence, in the previous work (Kim et al., 2003; Kim et al., 2004), the Tripodal schematic control architecture was proposed to tackle the problems. Many related research activities have been done. However, it is still a challenging problem to develop the robot architecture by carefully taking into account user needs and requirements, implement robot application components based on the architecture, and integrate these components in a systematic and comprehensive way. The reason is that the developers of service robots generally tend to be immersed in technology specific components, e.g. vision recognizer, localizer and path planner, at an early stage of product development without carefully considering architecture to integrate those components for various services (Kim et al., 2005). Moreover, engineers and developers are often grouped into separate teams in accordance with the specific technologies (e.g., speech processing, vision processing), which makes integration of these components more difficult (Dominguez-Brito et al., 2004; Kim et al., 2005). In such a project like T-Rot, particularly, several engineers and developers (i.e., approximately, more than 150 engineers) from different organizations and teams participate in the implementation of the service robot. Each separate team tends to address the specific technologies such as object recognition, manipulation, and navigation and so on. Engineers who come from different teams are concerned with different characteristics of the system. Thus, a common medium is required to create mutual understanding, form consensus, and communicate with each other for successfully constructing the service robot. Without such a medium or language, it is difficult to sufficiently understand the service robot system and interact between teams to integrate components for services.

Within the domain of software engineering, many approaches have been suggested for a systematic and complete system analysis and design, and for the capture of specifications. The object-oriented paradigm (Booch, 1994; Jacobson, 1992) is a widely-accepted approach to not only cover the external and declarative view of a system, but also at the same time bridge seamlessly with the internal implementation view of a system (Jong, 2002). Object-oriented concepts are crucial in software analysis and design because they focus on fundamental issues of adaptation and evolution (Gomaa, 2000). Therefore, compared with the traditional structured software development methods, object-oriented methods are a more modular approach for analysis, design, and implementation of complex software systems, which leads to more self-contained and hence modifiable and maintainable systems. More recently, the Unified Modeling Language (UML) (UML, 2003; Fowler & Scott, 2000) has captured industry-wide attention for its role as a general-purpose language for modeling software systems, especially for describing object-oriented models. The UML notation is useful to specify the requirements, document the structure, decompose into objects, and define relationships between objects in a software system. Certain notations in the UML have particular importance for modeling embedded systems (Martin et al., 2001; Martin, 2002), like robot systems. By adopting the UML notation, development teams thus can communicate among themselves and with others using a defined standard (Gomaa, 2000; Martin et al., 2001; Martin, 2002). More importantly, it is essential for the UML notation to be used with a systematic object-oriented analysis and design method in order to be effectively applied (Gomaa, 2000).

As a result, our aim is to develop the intelligent service robot based on the systematic software engineering method, especially for real-time, embedded and distributed systems with UML. To do so, we applied the COMET method, which is a UML based method for the development of concurrent applications, specifically distributed and real-time applications (Gomaa, 2000). By using the COMET method, it is possible to reconcile specific engineering techniques with the industry-standard UML and furthermore to fit such techniques into a fully defined development process towards developing the service robot systems.

In this paper, we describe our experience of applying the COMET /UML method into developing the intelligent service robot for the elderly, called T-Rot under development at CIR. In particular, we focused on designing an autonomous navigation system for the service robot, which is one of the most challenging issues for the development of service robots.

Section 2 describes the hardware configuration and services of the T-Rot, and discusses the related work. Section 3 illustrates how to apply the COMET method into designing and developing the autonomous navigation system for the service robot, and discusses the results of experiments. The lessons learned from the project are summarized in section 4, and section 5 concludes the paper with some words on further work.

2. Background on T-Rot

2.1 PSR and T-Rot

At KIST, intelligent service robots have been developed in large-scale indoor environments since 1998. So far, , PSR-1, PSR-2, which performs delivery, patrol, and floor cleaning jobs, and a guide robot Jinny, which provides services like exhibition guide and guidance of the road at a museum, have been built (Kim et al., 2003; Kim et al., 2004) (see Fig. 1). The service robot T-Rot is the next model of the PSR system under development for assisting aged

persons. Development of T-Rot, in which our role is developing and integrating robot software, started in 2003 by mainly CIR with more than 10 groups consisting of more than 150 researchers and engineers from academia and industry. This project is based on the needs and requirements of elderly people through the studies and analysis of the commercial health-care market for providing useful services to them. Thus, the aim of this project is to develop the intelligent service robot for the elderly by cooperating and integrating the results of different research groups. This project that is divided into three stages will continue until 2013 and we are now in the first stage for developing the service robot incrementally to provide various services.

PSR-1   **PSR-2**   **Jinny**
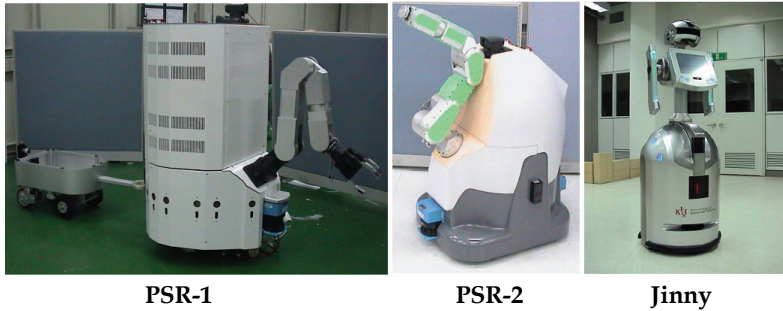
Fig. 1. KIST service robots

2.2 Hardware of T-Rot

The initial version of T-Rot, as shown in Fig. 2, has three single board computer (SBC), that is, mobile Pentium 4 (2.2GHz) and 1GB SDRAM on each SBC. In terms of software environment, Linux Red hat 9.0 and RTAI (Real-Time Application Interface) (RTAI, 2004) are used as operating system. Fig. 3 shows hardware configuration as a whole. As mentioned earlier, development of T-Rot is conducted incrementally for various services and thus the platform will be extended with manipulators and robot hands later. In our project, we developed the robot software based on the initial version of the platform. The details of the hardware platform are described in Table 1.

Fig. 2. T-Rot robot hardware platform
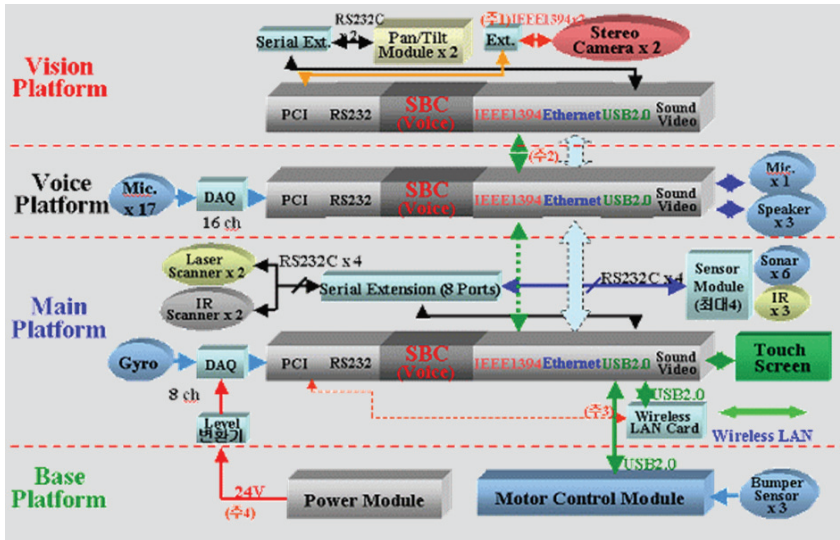
Fig. 3. T-Rot robot hardware platform configuration

SBC	Intel Mobile Pentium 4 (2.2 GHz)
	1GB SDRAM
	30GB Hard Disk
Voice	16 microphones for speaker localization
	1 microphone for speech recognition
	1 speaker for speech generation
Vision	2 stereo vision cameras for recognizing users and objects (1288 H x 1032 V maximum resolution and 7Hz frame rates)
	Pan/Tilt for controlling the vision part
Sensor	2 laser scanners (front and back)
	2 IR scanners (front and back)
	12 Ultrasonic sensors
	1 Gyroscope sensor for measuring balance
Actuator	2 actuators for two drive wheels (right and left)
	2 free wheels (the support wheels)
	2 Servo Motors (100 [w])
	2 encoders (2048 ppr)
	2 bumpers
Interface	1 TFT LCD & Touch (10.4″ 1024x768, 26000 colors)
	KVM (Keyboard/Mouse)
	Wireless LAN for communications

Table 1. T-Rot hardware platform devices

2.3 Robot services

Some of the primary services under-developed that the initial version for T-Rot provides for the elderly are described as below.

- Voice-based Information Services: The robot T-Rot can recognize voice commands from a user (i.e., an aged person) via microphones equipped with the robot and can synthesize voices for services. While a user is watching TV, the user can ask some questions about the specific TV program or request a task to open an Internet homepage by speaking the TV program name.

- Sound Localization and Voice Recognition: A user can call a robot's predefined name, to let the robot recognize the call while the robot knows the direction to move to the user. This service analyzes audio data from 3 microphones on the shoulder for sound localization and 16 mic array on the head for speech recognition to recognize the command from the user.

- Autonomous navigation: A user can command the robot to move to a specific position in the map to perform some task. For instance, the robot can navigate to its destination in the home environment via its sensors, which include laser scanners and ultrasonic sensors. The robot plans a path to the specified position, executes this plan, and modifies it as necessary for avoiding unexpected obstacles. While the robot is moving, it constantly checks sensor data from its sensors every 200 ms.

- An errand service: The robot can carry objects that a user (i.e., an aged person) usually uses, like a plate, books, a cane a cup of tea, beverages, etc according to the user's instructions. For instance, the user can order the robot to bring a cup of tea or beverage by speaking the name of the drink.

Of these T-Rot services, our emphasis was on the autonomous navigation service, which is one of the most challenging issues and is essential in developing service robots, particularly mobile service robots to assist elderly people. It includes hardware integration for various sensors and actuators, and the development of crucial navigation algorithms like maps, path planners, and localizers as well as software integration of software modules like a path planner, a localizer, and a map building module.

2.4 Control architecture of PSR

Up to now, there have been many related research activities to develop efficient and well-defined control architectures and system integration strategies for constructing service robots. A recent trend is that many control architectures are converging to a similar structure based on a hybrid approach that integrates reactive control and deliberation (Kim et al., 2004). At KIST, for developing service robots, that is PSR-1, PSR-2, and Jinny in the previous work (Kim et al., 2003; Kim et al., 2004), the Tripodal schematic control architecture was proposed as the solution to the problem.

One important point of Tripodal schematic design is to integrate robot systems by using a layered functionality diagram. The layered functionality diagram is a conceptual diagram of three layers for arrangement of various hardware and software modules and functions. It also shows the connectivity and the information flow between components. Those layers are composed of deliberate, sequencing, and reactive layers based on the hybrid approach. The purposes of the deliberate layer are to interface with a user and to execute a planning process. The sequencing layer is classified into two groups, that is, the controlling part that executes the process by managing the components in the reactive layer and the information

part that extracts highly advanced information from sensor data. The reactive layer controls the real-time command and hardware-related modules for sensors and actuators. The detailed description of whole control architecture of the PSR is introduced in (Kim et al., 2003).

However, as described earlier, in order to effectively apply this approach and the UML notation to developing service robots, it is essential to use a systematic software engineering process or methods like object-oriented analysis and design methods, especially for real-time and embedded systems. We believe that only a systematic and comprehensive software development process and method will be able to resolve the issues discussed before and will be vital for success in developing service robots.

2.5 The COMET method

COMET (Gomaa, 2000) is a method for designing real-time and distributed applications, which integrates object-oriented and concurrent processing concepts and uses the UML notation (UML, 2003; Fowler & Scott, 2000). The COMET object- oriented software life cycle model is a highly iterative software development process based around the use case concept. Therefore, in this project, the COMET method with UML was used to develop a system for autonomous navigation by the intelligent service robot, T-Rot. The method separates requirements activities, analysis activities and design activities, and these activities are briefly described as below. The details are described in section 3 with the case study.

- Requirements modeling - A use case model is developed in which the functional requirements of the system are defined in terms of actors and use cases.
- Analysis modeling - Static and dynamic models of the system are developed. The static model defines the structural relationships among problem domain classes. A dynamic model is then developed in which the use cases from the requirements model are refined to show the objects that participate in each use case and how they interact with each other.
- Design modeling – The software architecture of the system is designed, in which the analysis model is mapped to an operational environment. For distributed applications, a component based development approach is taken, in which each subsystem is designed as a distributed self-contained component.

3. Applying the COMET/UML method to T-Rot

In this section, we explain how to develop robot software for the autonomous navigation system with the COMET/UML method. In our project, the UML notation conforms to UML 1.3 and the Rational Rose tool is used.

3.1 Requirements modeling

Capturing the functional requirements of the system is the first phase in software development, which defines what the system should do or provide for the user. In our approach, developers can catch the functional requirements or services by using the use case model in terms of use cases and actors (see Fig. 4). To identify and define the requirements of the system more clearly, the system has to be considered like a black box. In the service robot, the actor can be usually a human user as well as external I/O devices and external timer.
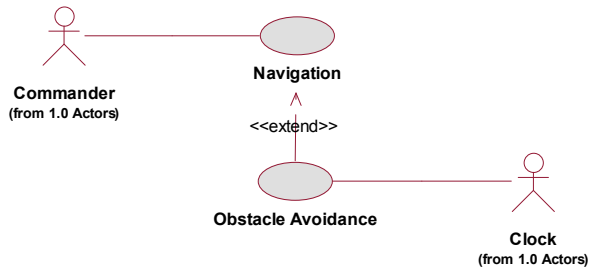
Fig. 4. Use case diagram for Navigation

Table 2 shows a specification for *Navigation* use case. In our navigation system, we identified a *Commander* and a *Clock* as an actor. While the robot is moving, if the robot recognizes obstacles, it should avoid them for continuing to go to the destination. Even when humans or objects suddenly appear, the robot must be able to stop to avoid crashing into those. However, in order to do this, the robot has to check that there are obstacles by using sensor data more often (e.g., every 50 ms) than the normal navigation system does (e.g., every 200 ms). As a result, the *Obstacle Avoidance* use case is extended from the *Navigation* use case. During the *Navigation* use case is executing, if the obstacles are recognized, then the *Obstacle Avoidance* use case is triggered to perform the emergency stop of the robot. If the obstacles disappear, the robot moves again to the destination.

Summary	The Commander enters a destination and the robot system moves to the destination.
Actor	Commander
Precondition	The robot system has the grid map and the current position is known
Description	1. The use case begins when the commander enters a destination. 2. The system calculates an optimal path to the destination. 3. The system commands the wheel actuator to start moving to the destination. 4. The wheel actuator notifies the system that it has started moving 5. The system periodically reads sensor data and calculates the current position. 6. The system determines that it arrives at the destination and commands the wheel actuator to stop. 7. The wheel actuator notifies the system that it has stopped moving and the use case is finished.
Alternative	6.1. If the system doesn't arrive at the destination, it keeps moving.
Postcondition	The robot system is at the destination and waiting for the next destination.

Table 2. Navigation use case

3.2 Analysis modeling
3.2.1 Static modeling

The objective of static modeling is to understand the interface between the system and the external environment and to describe the static structure of the system under development by developing a system context class diagram. It is specifically important for real-time and embedded systems like robot systems (Gomaa, 2000). The system context class diagram can be determined by static modeling of the external classes that connect to the system.
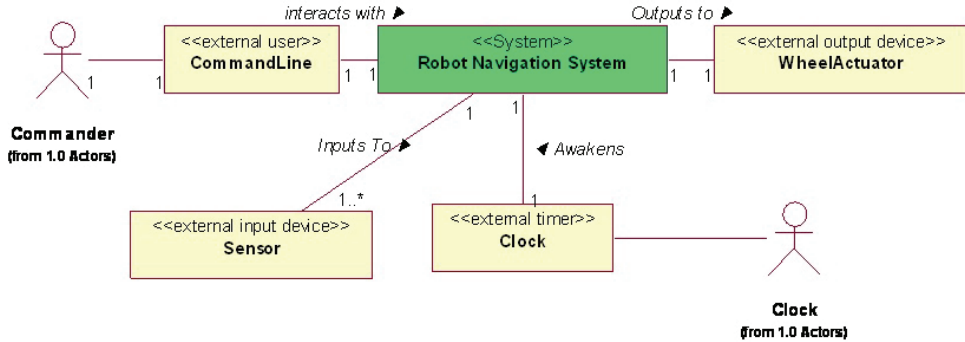
Fig. 5. Robot Navigation System context class diagram

The system context class diagram of the *Robot Navigation System* is shown in Fig. 5, which illustrates the external classes to which the system has to interface. In our navigation system, a commander enters a destination via a command line, to which the robot should move. The system uses sensor data via various sensors such as laser scanners, IR scanners, ultrasonic sensors, etc and it controls the wheels of the robot via the wheel actuator. Therefore, the external classes correspond to the users (i.e., a *Commander* who interacts with the system via a *Command Line*), and I/O devices (i.e., a *Sensor* and *Wheel Actuator*). A *Clock* actor needs an external timer class called *Clock* to provide timer events to the system. This external timer class is needed to periodically check sensor data via those sensors for avoiding obstacles (i.e., doing the emergency stop) while the robot is moving.

Next, to structure the *Robot Navigation System* into objects, object structuring needs to be considered in preparation for dynamic modeling. The objective of the object structuring is to decompose the problem into objects within the system. We identified the internal objects according to the object structuring criteria in COMET (see Fig. 6). In our system, interface objects, i.e. a *Command Line Interface*, *Sensor Interface* and *Wheel Actuator Interface* are identified by identifying the external classes that interface to the system, i.e. the *Command Line*, *Sensor*, and *Wheel Actuator*, respectively. There are four entity objects identified, that is, a *Current Position*, *Destination*, *Navigation Path* and *Navigation Map*, which are usually long-living object that stores information. In addition to those objects, there is a need for control objects, which provide the overall coordination for objects in a use case and may be coordinator, state-dependent control, or timer objects. The *Navigation System* has a state-dependent control object called *Navigation Control* that controls the wheel actuator and sensors. The states of the *Navigation Control* object are shown on a *Navigation Control* statechart (this will be discussed in the dynamic modeling). There are two timer objects, i.e. a *Navigation Timer* and an *Obstacle Avoidance Timer*. The *Obstacle Avoidance Timer* is activated by a timer event from an external timer to periodically check that there is any obstacle

around the robot. On the other hand, the *Navigation Timer* is started by the *Navigation Control* object and generates a timer event for navigation. Also, a *Localizer* algorithm object and *Path Planner* algorithm object are identified, which encapsulate an algorithm used in the problem domain, namely the autonomous navigation.
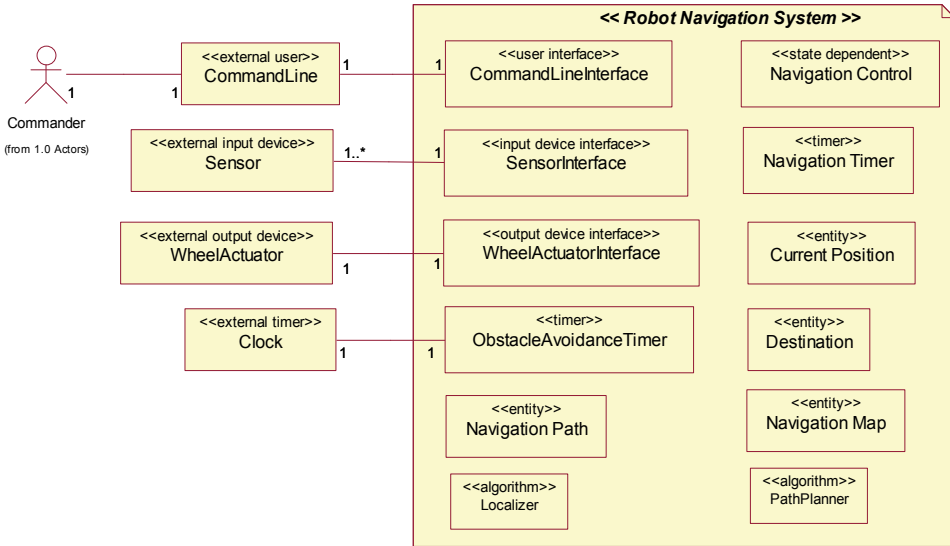
Fig. 6. Object structuring class diagram for Navigation System

3.2.2 Dynamic modeling

Dynamic modeling emphasizes the dynamic behavior of the system and plays an important role for distributed, concurrent and real-time system analysis. The dynamic model defines the object interactions that correspond to each use case and thus is based on the use cases and the objects identified during object structuring. In our case, collaboration diagrams are developed to show the sequence of object interactions for each use case. Additionally, if the collaboration involves the state-dependent object, which executes a statechart, the event sequence is shown on a statechart.

In the navigation system, the *Localizer* has the algorithm which can calculate the current position based on sensor data via sensors. So, the role of the *Localizer* is to update the current position of the service robot. In the *Path Planner* object, there is a method for calculating a path to arrive at the destination based on both sensor information and the current position that is calculated at the *Localizer*. The *Navigation Timer* is an internal timer that is controlled by the *Navigation Control*. After the destination is entered from the external user, the *Navigation Control* starts the *Navigation Timer*, then the timer generates a timer event periodically (i.e., every 200ms) until the *Navigation Control* stops the timer.

The *Navigation* use case starts with the commander entering the destination into the navigation system. The message sequence number starts at 1, which is the first external event initiated by the actor. Subsequence numbering in sequence is from 1.1 to 1.18 as shown in Fig. 7. The next message sequence activated by the *Navigation Timer* is numbered 2, followed by the events 2.1, 2.2, and so forth. The following message sequences are illustrated in the collaboration diagram (see Fig. 7).
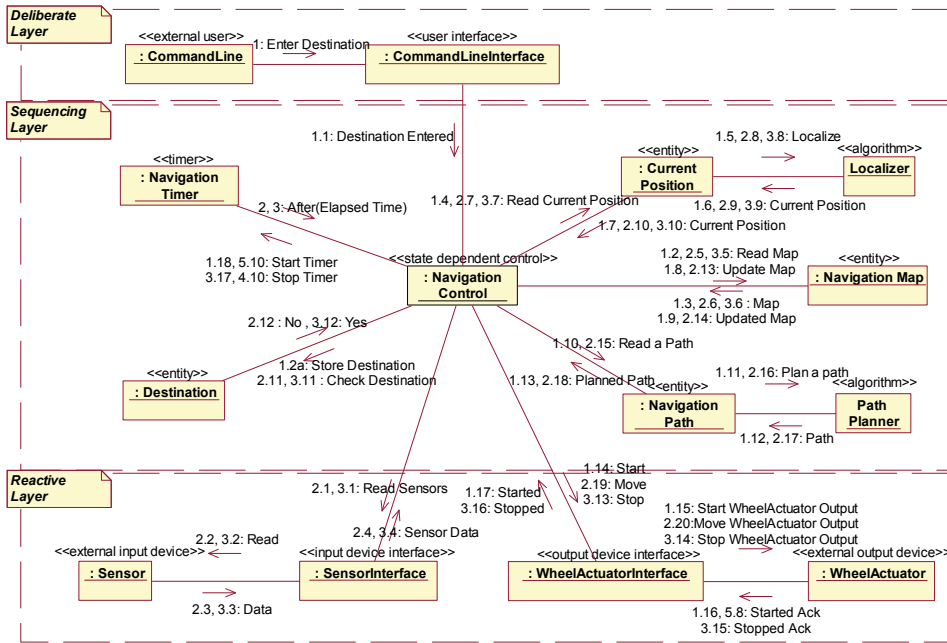
Fig. 7. Collaboration diagram for Navigation use case

The collaboration diagram for the *Obstacle Avoidance* use case is shown in Fig. 8. When activated by the *Obstacle Avoidance Timer* every 50 ms, the *Sensor Interface* object reads sensor data via various sensors (Events 4.1, 5.1, 6.1). If an obstacle is recognized, the *Obstacle Avoidance Timer* sends the emergency stop message to the *Wheel Actuator Interface* (Event 4.5). Afterwards, the timer also sends a suspend event to the *Navigation Control*. If the obstacle disappears, the timer sends a restart event to the *Navigation Control* for the robot to move again.
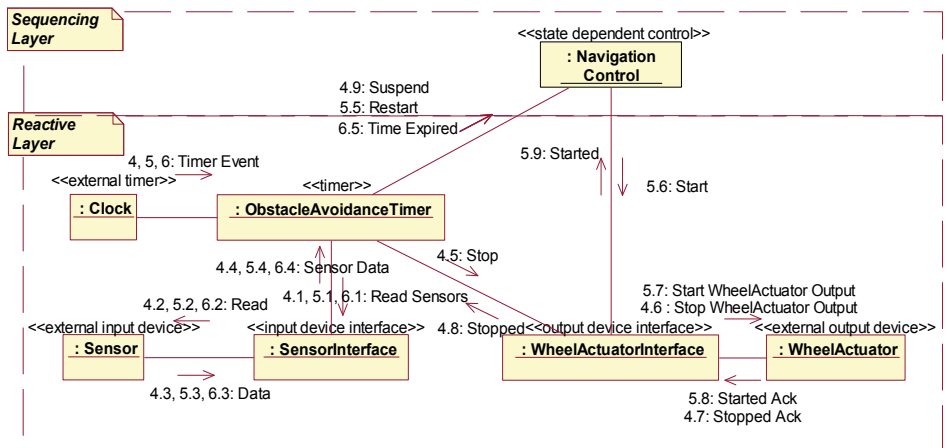
Fig. 8. Collaboration diagram for Obstacle Avoidance use case

With COMET, the software architecture can be based on a software architectural style (pattern) such as client/server or layers of abstraction. In our project, the layered strategy of the Tripodal schematic design described in section 2 is applied for design and modeling of the robot system, which provides a conceptual diagram of three layers (i.e., deliberate, sequencing, and reactive layers) for arrangement of various hardware and software modules and functions. Therefore, in the collaboration diagrams (see Fig. 7 and 8), the *Command Line Interface* is located in the deliberate layer and the *Sensor Interface*, *Wheel Actuator Interface*, and *Obstacle Avoidance Timer* are in the reactive layer. The others are positioned in the sequencing layer.
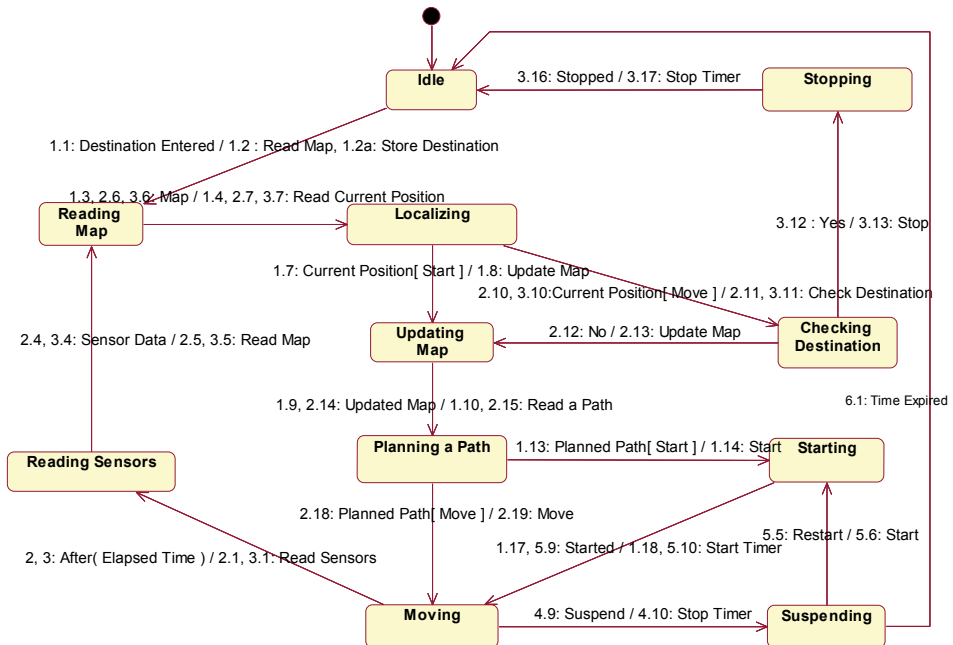
Fig. 9. Statechart for Navigation Control

In our navigation system, after drawing the collaboration diagrams for the *Navigation* and *Obstacle Avoidance* use cases which include the *Navigation Control* state-dependent object, we develop a *Navigation Control* statechart, which is executed by the *Navigation Control* object. The statechart needs to be considered in connection with the collaboration diagram. Specifically, it is required to take into account the messages that are received and sent by the control object, which executes the statechart (Gomaa, 2000). An input event (e.g., 1.1: *destination entered*) into the *Navigation Control* object on the collaboration diagram should be consistent with the same event shown on the statechart. The output event, which causes an action, enable or disable activity, like 1.2: *Read Map* (which cases an action) on the statechart must be consistent with the output event depicted on the collaboration diagram.

Because the statechart modeling involves two state-dependent use cases in the navigation system, it is also required to consolidate the two partial statecharts to create a complete statechart. The complete statechart for both the *Navigation* and *Obstacle Avoidance* use cases is shown in Fig. 9.

3.3 Design modeling
3.3.1 Software architecture
In this phase, all collaboration diagrams developed for use cases in the analysis model are merged into the consolidated collaboration diagram. The consolidated collaboration diagram is thus intended to be a complete description of all objects and their interactions.

The consolidation of the two collaboration diagrams respectively supporting the two use cases is shown in Fig. 10. Some objects and message interactions appear on more than one collaboration diagram. For instance, the *Navigation Control*, *Navigation Timer*, *Sensor Interface* and *Wheel Actuator Interface* objects participate in both the *Navigation* and *Obstacle Avoidance* use cases. For those objects, their message interactions are only shown once in the consolidated collaboration diagram.

3.3.2 Architectural design of distributed real-time systems
The robot system is a distributed embedded system and executes on distributed nodes by the communication methods like TCP/IP, CAN (Controller Area Network), and Wire/Wireless LAN. With COMET, a distributed real-time system is structured into distributed subsystems. Tasks in different subsystems may communicate with each other via several types of message communication, such as asynchronous, synchronous with reply, synchronous without reply, and client/server communications, etc. Hence, we should define distributed nodes and their messages to each node.
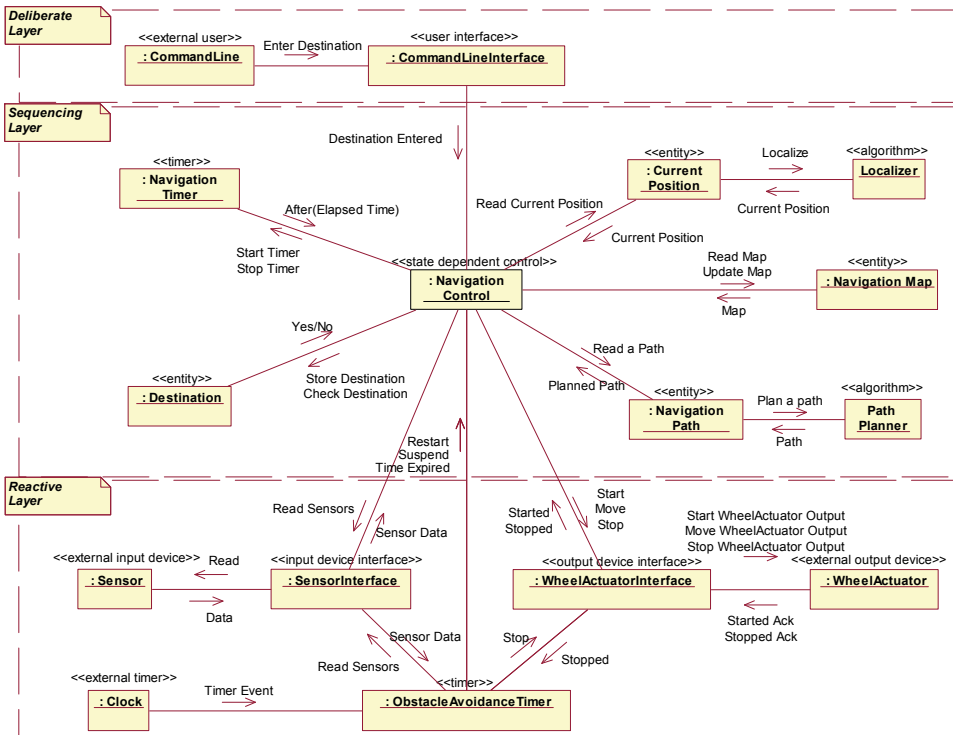
Fig. 10. Consolidated collaboration diagram for Navigation System

The overall distributed software architecture for the robot navigation system is depicted in Fig. 11. In the robot system, objects that are part of the navigation are located in the robot navigation system. The robot navigation system communicates with the external I/O devices via synchronous message without reply communication and with the external timer via asynchronous message communication.
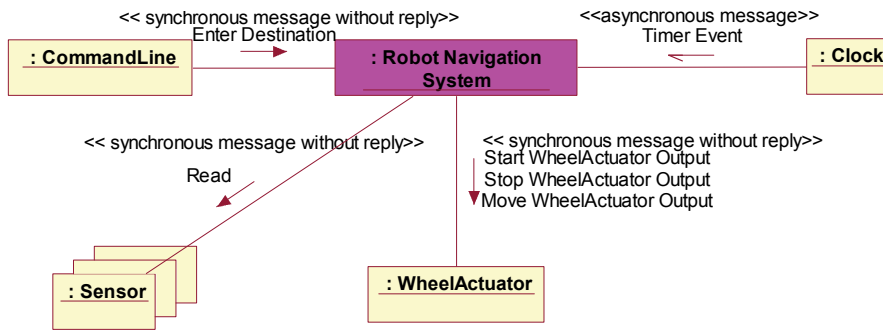
Fig. 11. Distributed software architecture for Navigation System

3.3.3 Task structuring

During the task structuring phase, a task architecture can be developed in which the system is structured into concurrent tasks, and the task interfaces and interconnections are defined. A task is an active object and has its own thread of control. In this sense, the term "object" will be used to refer to a passive object in this paper. In COMET, task structuring criteria are provided to help in mapping an object-oriented analysis model of the system to a concurrent tasking architecture. At the end of this phase, a task behavior specification (TBS) is developed.

The task architecture for the *Navigation System* is shown in Fig. 12. In order to determine the tasks in the system, it is necessary to understand how the objects in the application interact with each other based on the collaboration diagrams. In the collaboration diagram of Fig. 7, the *Localizer* object reads sensor data and the map from the *Current Position* object, calculates a new current position, and sends the current position to the *Current Position* object for updating it. Thus, the *Localizer* object is structured as an asynchronous algorithm task called *Localizer*. There are two asynchronous algorithms, i.e. *Localizer* and *Path Planner*, which are internal asynchronous tasks. There are four passive entity objects, i.e. *Destination*, *Current Position*, *Navigation Map*, and *Navigation Path*, which do not need a separate thread of control and are further all categorized as data abstraction objects. The *Sensor* and *Wheel Actuator* are a passive input device and a passive output device, respectively because they do not generate an interrupt on completion of the input or output operation.

The *Navigation Control* is a state-dependent control object that executes the *Navigation Control* statechart and is structured as a control task because it needs to have a separate thread of control. The *Navigation Control* object can be combined with the *Command Line Interface*, *Navigation Timer*, *Sensor Interface*, and *Wheel Actuator Interface* objects into one task, Navigation Controller, based on the control clustering task structuring criterion because it is

not possible for them to execute concurrently (see the middle of Fig. 12). The *Obstacle Avoidance Timer* object is structured as a periodic task, activated periodically to read sensor data. It can be grouped with the *Sensor Interface* and *Wheel Actuator Interface* into one sequentially clustered task, *Obstacle Avoidance Controller* based on sequential clustering since those are carried out in a sequential order. The design of those composite tasks, the *Navigation Controller* and *Obstacle Avoidance Controller* are considered in the next section (i.e., detailed software design).
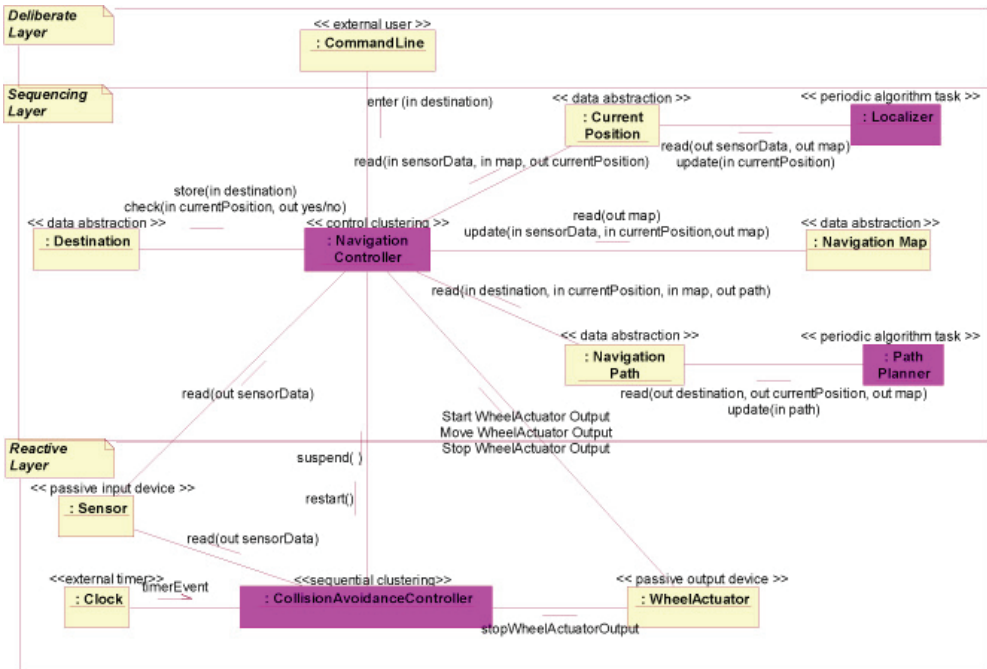
Fig. 12. Task architecture for Navigation System

After developing the task architecture, a task behavior is described for specifying the characteristics of each task based on COMET. During the task structuring, the TBS focuses on the task inputs and outputs. One part of the TBS, i.e. the task's event sequencing logic is defined in the detailed software design phase.

3.3.4 Detailed software design
The internals of composite tasks which have passive objects nested inside them are designed, detailed task synchronization issues are addressed, and each task's internal event sequencing logic is defined in this phase. Before this is done, the information hiding classes (from which the passive objects are instantiated) are designed. In particular, the operations of each class and the design of the class interfaces are determined and specified in a class interface specification (because of space limitation, the detailed TBS and the class interface specification have not been included).

Let us consider the internal design of the *Navigation Controller*, which is a composite task designed as a control clustering task, to show the nested information hiding objects (see Fig. 13). The information hiding objects are the *Navigation Control* state-dependent control object, the *Sensor Interface* and *Wheel Actuator Interface* objects, the *Navigation Timer* object and the user interface object, the *Command Line Interface*. In addition, the *Navigation Controller* contains one coordinator object called *Navigation Coordinator*, which receives incoming messages and coordinates the execution of the other objects. That is, the *Navigation Coordinator* extracts the event from the request and calls *Navigation Control.processEvent (in event, out action)* (see Fig. 13). The *Navigation Control* returns the action to be performed like store, check, start, etc according to the state transition table. Afterwards, the *Navigation Coordinator* initiates the action.
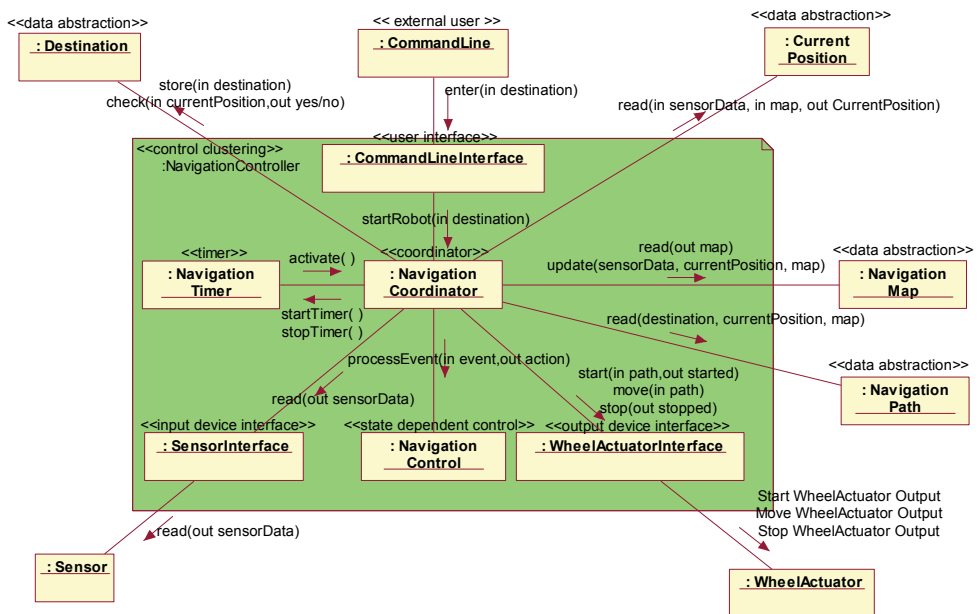
Fig. 13. Detailed software design for Navigation Controller

In our system, communication between tasks such as the *Navigation Controller*, *Localizer*, and *Path Planner* is through data abstraction classes like the *Current Position* and *Navigation Path*. As a result, connector objects (Gomaa, 2000) are not used for the message communication interface between tasks.

Lastly, the task's event sequencing logic is specified, which describes how the task responds to each of its message or event inputs. However, instead of using informally Pseudo code in COMET, in this project, task event diagrams are developed for tasks by using the UML sequence diagrams for understanding and readability, which turned out to be very useful when to implement the tasks. Fig. 14 illustrates the task event diagram for the *Navigation Controller*.
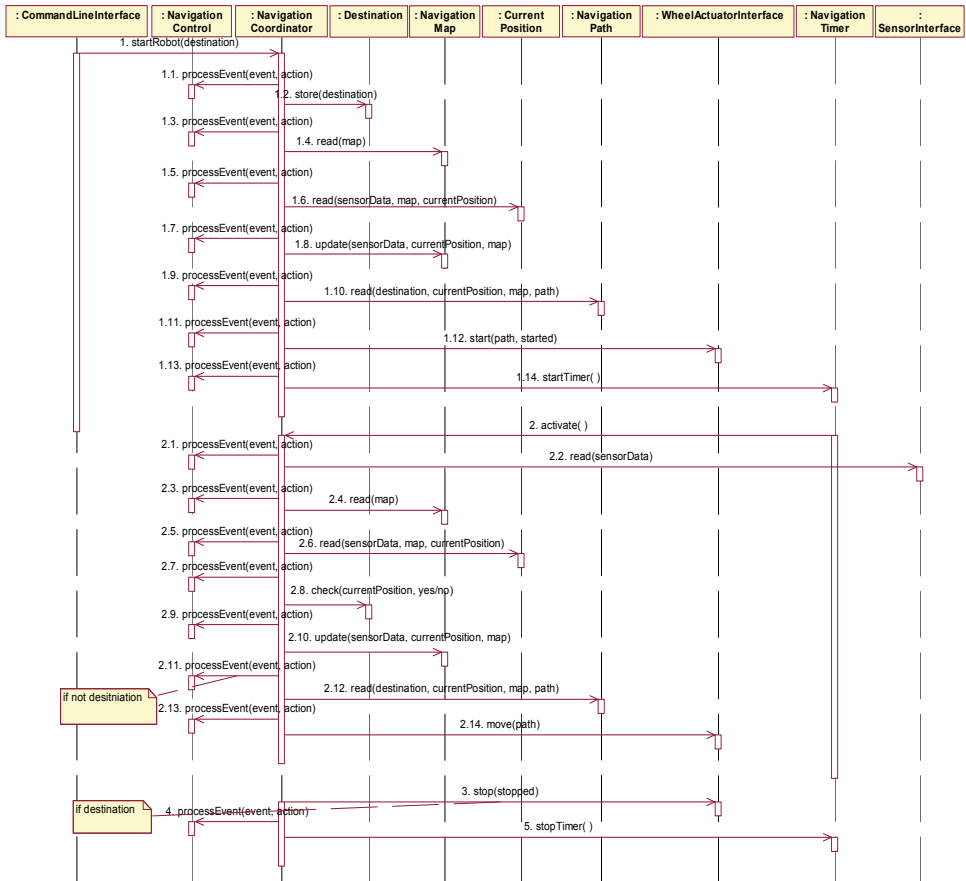
Fig. 14. The task event diagram for Navigation Controller

4. Lessons learned

This section summarizes the lessons learned from the project where we successfully applied the object-oriented method with UML to developing the service robot.

4.1 UML for service robot domain
Through the case study, we found that the UML standard was very useful as a notation for specifying the requirements, documenting the structure, decomposing into objects, and defining relationships between objects especially in a service robot system. Certain diagrams and notations were particularly importance for analyzing, designing, and modeling service robot systems as follows.

- Use case diagrams: With the use case model, services or functions (i.e., functional requirements), which a service robot performs or provides, can be defined in terms of actors who are users of the robot system and use cases. Each use case defines the behavior of some aspect of the robot system without revealing its internal structure.

- Class diagrams: The class diagram notation is used to depict the static model, which focuses on the static structure of the robot system. The class diagram shows the classes of objects in the system, their internal structure including attributes, their operations, and their relationships to other classes (such as associations and generalization/inheritance).
- Collaboration diagrams: This diagram shows how objects that participate in each use case interact with each other by sending and receiving messages in the dynamic model. It defines a specific way to use objects in the robot system by showing the possible interactions between them, especially to satisfy the needs described in the use case, namely provide the services. Compared to a sequence diagram, the diagram in particular is useful for synthesizing the collaboration diagrams to create the software architecture of the system as discussed in section 3.3.
- Sequence diagrams: This diagram show objects interaction arranged in time sequence and in particular could be used to describe the task event sequencing logic, which describes how the task responds to each of its message or event inputs. In COMET, the event sequencing logic is usually described informally in Pseudo code. We found that the sequence diagram can help the engineers describe the task event sequencing logic and implement the tasks by showing the order in which messages are passed between tasks and objects.
- State chart diagrams: The service robot system is highly state-dependent like real-time embedded systems. This diagram describes how state-dependent aspects of the system are defined by a finite state machine and can help design and developing highly state-dependent systems. It is also possible for this diagram to model object behavior over several use cases with the collaboration diagrams.

In addition, by using the UML notation as a defined standard, different research groups and development teams can communicate among themselves and with others to develop and integrate specific components for providing various services.

4.2 Importance of systematic process/method for service robot domain

In order to effectively apply the UML notation and the robot control architecture like the Tripodal schematic control architecture to developing service robots, it is essential to use them with a systematic software engineering process or method, like an object-oriented analysis and design method, especially for real-time and embedded systems. It is not possible to resolve the issues in integrating and developing the service robots discussed before without systematic and comprehensive software development methods, particularly for service robots.

In our case study, we applied COMET/UML method to developing the service robot. The COMET object-oriented software life cycle model is a highly iterative software development process based around the use case concept. In the requirements model, the service or functions (i.e., the function requirements) of the robot system are defined in terms of actors and use cases. In the analysis model, the use case is refined to describe the objects that participate in the use case, and their interactions. In the design model, the robot software architecture is developed, emphasizing issues of distribution, concurrency, and information hiding. This project showed that this was a viable approach because applying the COMET method with UML led to developing an effective service robot architecture by carefully

taking into account user needs and requirements, implementing technical components based on the architecture, and integrating these components in a systematic and comprehensive fashion.

4.3 Customizing the COMET method for service robot domain

Service robots like , PSR-1, PSR-2 and Jinny have been built at KIST based on the Tripodal schematic control architecture. The Tripodal schematic design addressed on developing efficient and well-defined control architecture and a system integration strategy for constructing service robots. T-Rot is the next model of the PSR system under development for assisting aged persons. One of our aims is to develop the intelligent service robot for the elderly by cooperating and integrating the results of different research groups in accordance with the Tripodal schematic control architecture that has already been implemented on the PSR and successfully tested. Thus, the layered strategy of the Tripodal schematic design has been applied for design and modeling of the T-Rot. In the collaboration diagrams of the analysis modeling, and the consolidated collaboration diagram and the task architecture of the design modeling, the *Command Line Interface* is located in the deliberate layer for interfacing with a user, while the *Sensor Interface*, *Wheel Actuator Interface*, and *Obstacle Avoidance Timer* are in the reactive layer for controlling and managing the components in the reactive layer. The *Navigation Control*, *Navigation Timer*, *Destination*, *Current Position*, *Navigation Path*, *Navigation Map*, *Localizer*, and *Path Planer* are positioned in the sequencing layer for controlling the robot motion by executing relatively simple computations in real-time. As a result, the Tripodal schematic control architecture was helpful in arranging various hardware and software modules and functions.

Additionally, as stated in section 4.1, in COMET, the event sequencing logic is usually described informally in Pseudo code. We found that the sequence diagram can help the engineers describe the task event sequencing logic and implement the tasks by showing the order in which messages are passed between tasks and objects. Hence, instead of using informal Pseudo code, task event diagrams were developed for tasks by using the UML sequence diagrams to improve understanding and readability. It turned out that these task event diagrams are very useful when implementing these tasks.

4.4 Human communication

Human communication to understand and develop what is desired of the service robot is likely to be more difficult than expected. In our case study, most engineers who are involved in the project come from the mechanical or robotics engineering field. The different research groups and teams tend to focus on their own technology and components and thus it is not easy to realize how much knowledge they have and how much information will need to be made explicit and communicated to integrate those components for the service robot. Several things can be done to improve the situation. One is for engineers from different teams, especially software engineers and mechanical engineers to work together for analyzing, designing, and developing the robot system during the project. It is very important that all engineers and developers from different groups and teams interact directly. Also, in order to develop a common ground for understanding the domain, technology, process and method, a common medium or language such as UML is critical.

In addition to the standard notation like UML, guidelines about what notation to use, when to use it, and how to use the notation comprehensively and systematically are required. This

is why the method like COMET is needed. Domain knowledge and experiences in each area will make it much easier to communicate what is desired, e.g. service robot domain, the autonomous robot navigation, vision processing, and so on for software engineers, and object-oriented concepts, software development process, and UML, etc for mechanical engineers. If there is relatively little domain knowledge and experience, to have one day or half-day technical workshop is needed. This has proved useful in a variety of settings in the development of the robot system, such as developing and increasing background knowledge of the domain and technology.

4.5 Necessity of multi-aspect integration method for service robot domain

A service robot should be able to perform several tasks autonomously to provide various services for human beings in a dynamic and partially unknown environment by applying both technology and knowledge. In order to be able to achieve complex tasks, perform new tasks, and integrate data learned from experience for the robot services, it is required to consider not only the robot's behavior, but also other robot's characteristics such as learning, planning, decision-making, and knowledge representation. It is necessary to allow existing robot behaviors to be used in new ways, to plan for accomplishing more complex tasks, to reuse the knowledge of one task in other tasks, and to complete tasks more efficiently by learning various action sequences.

In the case study, we focused on designing and modeling the robot's behavioral aspect, which is related to the sequencing and reactive layers in the Tripodal layered design, by applying the COMET/UML method. However, it is clear that planning and learning abilities have to also be considered when designing and developing a service robot, which correspond to the deliberate layer that is responsible for interfacing with a user and executing the planning process. As a consequence, a task manager, which is located in the deliberate layer, has been in charge of these robotic abilities in the project. Because the planning process is knowledge based and not reactive, a different analysis and design approach is needed for the task manager. Hence, we are convinced that methods to model the robot's learning, planning and decision making aspects as well as to incorporate, use and maintain task knowledge are necessary. Furthermore, it is essential to integrate these methods with the COMET method into a multi-aspect integration method for developing service robot software.

5. Conclusions and future work

Service robots have been suggested for a growing number of applications. A service robot is a complex system as it includes various technical components (i.e., hardware and software) to be integrated correctly and many different research groups to develop the components. As a result, it is not only essential to develop complex algorithms or technical components, but also to integrate them adequately and correctly to provide the various robot services.

In the paper, we have presented our case study where we developed the autonomous navigation system for the intelligent service robot for the elderly, T-Rot. The object-oriented method for real-time embedded systems, COMET has been applied to the service robot T-Rot with the industry standard UML. It makes it possible to reconcile specific engineering techniques like robot technologies with the UML notation and furthermore to fit such techniques into a fully defined development process towards developing the service robot

system. In this way, we contribute to developing service robot software with UML in a systematic manner.

The service robot T-Rot is still under development (at this point, we are at the first stage of total three stages). Thus, the current status of our work is to extend applications that include vision processing, speech processing and manipulation for providing various robot services. Also, we work on designing the knowledge-based task manager for improving the robot's ability.

6. Acknowledgements

7. References

Booch, G. (1994). Object-Oriented Analysis and Design with Applications. 2nd edn, Addison Wesley

Dominguez-Brito, C.; Hernandez-Sosa, D.; Isern-Gonzalez, J. & Cabrera-Games, J. (2004). Integrating robotics software, Proceedings of IEEE International Conference on Robotics and Automation, pp. 3423-3428, New Orleans, LA, May 2004, IEEE Computer Society Press

Fowler, M. & Scott, K. (2000). UML Distilled 2nd Edition. Addison Wesley

Gomaa, H. (2000). Designing Concurrent, Distributed, and Real-Time Application with UML, Addison-Wesley Object Technology Series

Jacobson I. (1992). Object-Oriented Software Engineering, Addison Wesley

Jong, G. (2002). A UML-Based Design Methodology for Real-Time and Embedded Systems, Proceedings of Design Automation and Test in Europe (DATE2002), pp. 776-778, Paris, France, March 2002, IEEE Computer Society Press

Kawamura, K. & Iskarous, M. (1994). Trends in service robots for the disabled and the elderly, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'94), pp. 1647-1654, Kyongju, Korea, October 1994, IEEE Computer Society Press

Kim, G.; Chung, W.; Kim, M. & Lee, C. (2003). Tripodal Schematic Design of the Control Architecture for the Service Robot PSR, Proceedings of the IEEE Conference on Robotics and Automation, pp. 2792-2797, Taiwan, 2003, IEEE Computer Society Press

Kim, G.; Chung, W.; Kim, M. & Lee, C. (2004). Implementation of Multi-Functional Service Robots Using Tripodal Schematic Control Architecture, Proceedings of IEEE Conference on Robotics and Automation, pp. 4005-4010, New Orleans, LA, May 2004, IEEE Computer Society Press

Kim, M.; Lee, J.; Kang, K.; Hong, Y. & Bang, S. (2005). Re-engineering Software Architecture of Home Service Robots: A Case Study, Proceedings of 27th International Conference on Software Engineering (ICSE2005), pp. 505-513, St. Louis, USA, May 2005, IEEE Computer Society Press

Martin, G. (2002). UML for Embedded Systems Specification and Design: Motivation and Overview, Proceedings of Design Automation and Test in Europe (DATE2002), pp. 773-775, Paris, France, March, 2002, IEEE Computer Society Press

Martin, G.; Lavagno, L. & Louis-Guerin, J. (2001). Embedded UML: a merger of real-time UML and codesign, Proceedings of the Nnth nternational Smposium on Hardware/Sofware Codesign (CODES 2001), pp. 23-28, ISBN 1-58113-364-2, Copenhagen, Denmark, April 2001, ACM 2001

Meng, Q. & Lee, M.H. (2004). Learning and Control in Assistive Robotics for the Elderly, Proceedings of the IEEE Conference on Robotics, Automation and Mechartonics, pp. 71-76, Singapore, December 2004, IEEE Computer Society Press

Pineau, J.; Montemerlo, M.; Pollack, M.; Roy, N. & Thrun, S. (2003). Towards robotic assistants in nursing homes: Challenges and results. Robotics and Autonomous Systems, Vol. 42, No. 3-4, 2003, 271-281, ISSN 0921-8890

Rofer, T.; Lankenau, A. & Moratz, R. (2000). Service Robotics-Applications and Safety Issues in an Emerging Market, Proceedings of Workshop W20 on European Conference on Artificial Intelligence (ECAI) 2000, Berlin, August 2000

Real-Time Application Interface. (2004). Available at: http:// www.rtai.org

Schraft, R.D. (1994). Mechatronics and robotics for service applications. IEEE Robotics and Automation Magazine, Vol. 1, No. 4, December 1994, 31-35, ISSN 1070-9932

OMG Unified Modeling Language, Version 1.5. (2003). Available at:http://www.uml.org

You, B.; Hwangbo, M.; Lee, S.; Oh, S.; Kwon, Y. & Lim, S. (2003). Development of a Home Service Robot 'ISSAC', Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003), pp. 2630-2635, Nevada, Las Vegas, 2003, IEEE Computer Society Press

Universal Design with Robots Toward the Wide Use of Robots in Daily Life Environment

Nobuto Matsuhira[1], Junko Hirokawa[1],
Hideki Ogawa[1] and Tatsuya Wada[2]
[1]Corporate Research & Development Center, Toshiba Corporation
[2]Tama Art University
Japan

1. Introduction

Robot technology has been cultivated by developing robots that work in special environments, such as industrial robots, robots for performing maintenance in nuclear power plants, and robots for use in space. On the other hand, the application fields of robots are expanding to include aspects of daily life, for example medical robots in hospitals, welfare robots in homes for the elderly, robots to perform cleaning in buildings and robots for use at home. Furthermore, the essentials of robot technology (RT) have been applied widely to endow mechatronics products with intelligence. Advanced examples include implementation of a robotic function in an automobile so that it automatically maintains a distance from the car in front, and an automatic parking function. In particular, various kinds of robots for home use, such as security robots, caretaking robots, information service robots, and communication robots, are now being developed (Japan Robot Association, 2003). A robot that provides support to people at home is generally called a "Daily Life Support Robot" or "Human Symbiosis Robot" in Japan (Japan Robot Association, 2005, 2006), (Matsuhira et al., 2005), (Hosoda et al., 2006). However, it is still difficult to realize a home robot capable of moving through doors and coping with differences in level, although the growing popularity of barrier-free designs are helpful in this regard. Here, we consider the concept of Universal Design (UD). UD, which is a design concept that aims to satisfy the needs of everybody in daily life, is also important for home robots or robots in daily life environment. Robots can move easily where wheelchairs can move easily. Robots can easily handle what a person who has trouble handling things can handle easily. Similarly, robots can easily find a sign that is easy for a person with impaired sight to find. A robot will be thought as one of users for UD. Thus the UD is important. Improvement of the environment by applying the UD concept is expected to lead to expansion of the sphere of robot activity and to spur practical use of robots. We propose Universal Design with Robots (UDRob™), a universal design concept encompassing both people and robots (Matsuhira et al., 2004), (Wada, 2004). A conceptual design of a robot-system based on UDRob™ has been developed and is presented here. So far, robot design has been mainly considered in terms of the figure or shape of robots themselves. We adopt the UD both for robots and the

environment as shown in Fig. 1. We believe that this new concept is important for facilitating the practical use of robots. A major problem concerns the performance of robots working in homes. Industrial robots perform predetermined tasks in predetermined environments. In society, many people need help in view of aging population especially in Japan. There are expectations that robots will help such people. The problem is that the environments and tasks for home robots are diverse. UDRob™ is a novel solution that originated from the collaboration between robotic engineers and designers. Toshiba Corporation and Tama Art University jointly conducted this research from 2004 to 2006.
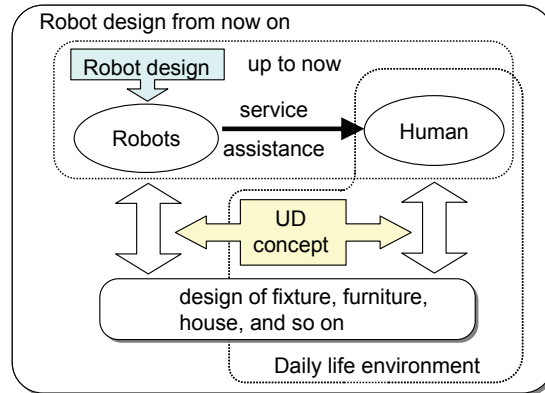
Fig. 1. New concept of robot design using UD

2. Universal design with robots

According to the Japan Robot Association (JARA), the Japanese market for service robots will be worth JPY 3,000 billion in 2010, increasing to JPY 8,000 billion in 2025 (Matsuhira & Ogawa, 2004). Main application field is the daily life support. Although a great deal of R&D on robotics is being done in Japan, there are few robots in practical use. There is a big gap between expectations and the actual performance of robots at present. One of the main reasons is that environments for the robots are diverse and the robot performance cannot catch up with the application requirements. In practice, it is very difficult to develop robots that have sufficient flexibility for the environments encountered in daily life such as *where is the object ?*, *what is it ?*, and *how to handle it ?* UD is one solution to this problem.

Hereafter, well-known definitions of UD are adapted for robots.

a)   Equitable Use: A robot can use the objects.
b)   Flexibility in Use: A robot handles the objects freely.
c)   Simple and Intuitive: A robot handles the objects easily.
d)   Perceptible Information: A robot easily recognises the objects.
e)   Tolerance for Error: A robot handles the objects safely.
f)   Low physical effort: A robot handles the objects by simple motion, without fine positioning accuracy or dextrous motion being required.
g)   Size and space for approach and use: A robot approaches and handles the objects easily.

Essentially, these definitions mean that for robots and humans it is easy to recognise objects, access objects, and handle objects. If we solve the interface design, including mechanical and

audio-visual consideration, from the viewpoint of the UD concept, it will be possible to use robots more widely in daily life.

There have been few researches on UD that include robots. Based on a related concept, WABOT-HOUSE was built to coexist with robots (Sugano et al., 2006), and a practical experiment of robots was conducted on a public road in Fukuoka prefecture, Japan, with the aim of having robots play an active part in environments encountered in the course of daily life (Japan Robot Association, 2004). On the other hand, attempts have been made to combine networks and robots so that robots can acquire information, which includes self-localization data, map data, and what to do, from environmental sensors and knowledge such as "networked robot" (Hagita et al., 2005), (Chong et al., 2004). Recently information-structured environments have been actively developed as a common platform technology for next-generation robots (Tanie et al., 2007), (Kanda et al., 2007), (Hasegawa et al., 2006), (Ohba et al., 2007), (Sugawara et al., 2007). So far, information technology (IT) environment has been improved drastically in Japan as a common infrastructure. From now on, it is inevitable to arrange the physical interface as environmental contact point for robots to work in various applications as shown in Fig. 2. In a factory or a power plant, environments can easily be arranged for robots. However, a handle designed specially for a maintenance robot may be difficult for people to use. For a robot to work at home, an integrated design covering both the environment and the robot is required. In this paper, the design of a home robot was studied, considering the environment in which people live, and features of houses such as doors, steps, and stairs, based on UDRob™.
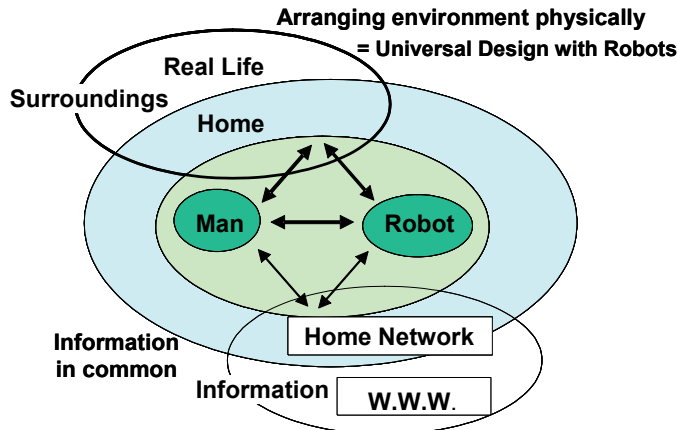
Fig. 2. Robot and infrastructure of the living and the networking environment

3. Adopting universal design with robots

We expect robots to move around freely and work with arms and hands, handling things in the home or elsewhere. There are many difficult issues to be solved in daily life environment such as motion control, localization, planning, recognition and perception. Here, UDRob™ can facilitate them. We focused on "to move", "to handle", "to look", and "to listen" and examined the arrangement of robot design from the perspective of design of the home.

3.1 To move

Regarding movement, robots are initially expected to move in a wide space such as public space or buildings, and then gradually be introduced into a more confined space such as a home. We have to secure aisles for robots, as well as stairs, steps, and doors. Obviously, it is advantageous to expand the robot's sphere of activity. For instance, an aisle where a wheelchair can pass easily is also suitable for a robot. Stairs and steps will be replaced by slopes. As for doors, power-assisted doors, automatic doors, and small doors for pets are already in use. We call this approach "Environmental interface design". It is not easy to facilitate movement of a robot and provide sufficient work space for it. But once the necessity of the robot is recognised, a suitable environment will be arranged. On the contrary, people will clean up their room for robots to work in the near future.

3.2 To handle

Handling comes after moving. Mechanical interface of handling targets should be as common as possible. Concretely, shapes of handling targets should facilitate easy handling by robots. So far, shapes of handling targets have been unified to facilitate handling by robots. For example, the handling part and coupling part of maintenance robots for power plants or space development are unified into several types as standard interfaces. We call this approach "Interface design of working target". These designs are helpful for simplifying tasks to reduce the cost of both hardware and software, and improve the reliability.

Consider a maintenance robot and maintenance work used in a nuclear power plant or in space development. A captive bolt is used avoiding falling out of hole. Picking up a fallen bolt and inserting it into the hole again is a very difficult task for robots and the fallen bolt may disturb the operation of the apparatus. To ease the positioning when the robot assembles components or fits together by insertion, compliance mechanisms to absorb misalignment are necessary. For example, a spring is used to release an overload, a tapered-shape is good to guide for fine positioning, a circular shape is also useful because parts can be inserted into a hole from any direction. There are many designs for maintenance robots or astronauts who cannot move dextrously because they are wearing the space suits with big gloves. Handles in there should be big enough to grasp when wearing gloves.

3.3 To look

There are many situations in which it is necessary for a robot to recognize the user's face and to measure what is where by using images. Accuracy of image processing is influenced greatly by a change of lighting or depends on the background. It is necessary to configure lighting condition, or to fix a place where more accurate results can be obtained. For recognizing places or positions or target objects, the heavy load for image processing can be reduced by using geometrical markers that show specific information of localization. Although image processing technology is advancing, robots don't have to rely on everything about environmental recognition to highly developed image processing in case there is another way to solve it. On the other hand, it is possible to acquire localization data from a specific protocol or networking device through a network by, for example, using radio-frequency identification tags (RFID tags) (Kim et al., 2005, 2006). We call this approach

"Environmental informative interface design". Furthermore, it is possible to have the shape of a target object serve as a landmark itself, such as a door, a doorknob, and a window frame. If it is possible to recognize from the shape of a handle that a door opens by pulling or pushing, or a window opens by sliding, this would simplify recognition and manipulation tasks not only for robots but also for humans. In the SLAM technology of mobile robots, a natural scene captured by a vision sensor is used to determine the location of the robot in the map (Thrun, 2001).

There are many geometric designs for interiors, floors, walls, etc. It's easy for a robot to use them to identify its position in the environment. IT is also important for the robot, and the robot is systemized with the network environment as an actual movable agent. But information from the actual shape of the object is more important as a UD, because signal information cannot be perceived by humans.

3.4 To listen

A robot is already able to recognise the human voice. But voice recognition is usually carried out through a microphone near to the user's mouth. In reality, however, the human is often distant from the robot. It is very difficult for the robot to understand what the user says because there is a lot of background noise in the daily life environment such as TV. Microphone array technology and noise cancelling technology have been developed to improve the performance of voice recognition (Amada et al., 2004), (Asano, 2004), (Brandstein, 1999), (Brandstein & Ward, 2001), (Nakadai et al., 2003). But it is expected to be a long time before robots can match the ability of humans to understand humans' voices in daily life environments. However if we talk to the robot using simple words, with a clear and loud voice as we do when we talk to the elderly, the robots are able to understand them easily. This is the concept of UDRob™.

4. Design of robot and environment

Tama Art University studied an environmental design suitable for a working home robot and human daily life. Storing things includes taking out and putting back, and delivering things so that it is realized by a robot in combination of environmental recognition with vision sensing, handling with arms and hands, and moving. The relationship among shelf as storage, tray as common interface, humans and robots are to be considered in the daily life.

a) Total image of the living room at the home

Fig. 3 shows a living room image with robots designed by UDRob™. The user assorts daily necessities to store in a tray divided by colour so that the robot recognizes the colour and marker in order to grasp a specific tray and deliver it to the user. Taking the example of daily medication, the user puts the medicine on the tray and sets the time schedule for taking it so that the robot delivers it to him/her as a regular basis, without running the danger of forgetting taking the medication, or overmedication. As mentioned above, the example of environmental design and practical use of the robot is shown as a concept of UDRob™.

Fig. 3. Image of a living room designed with UDRob™ concept

Fig. 4. Shared Shelf

(a) Variation of tray

(b) Tray as an interface

(c) Setting tray into the shelf
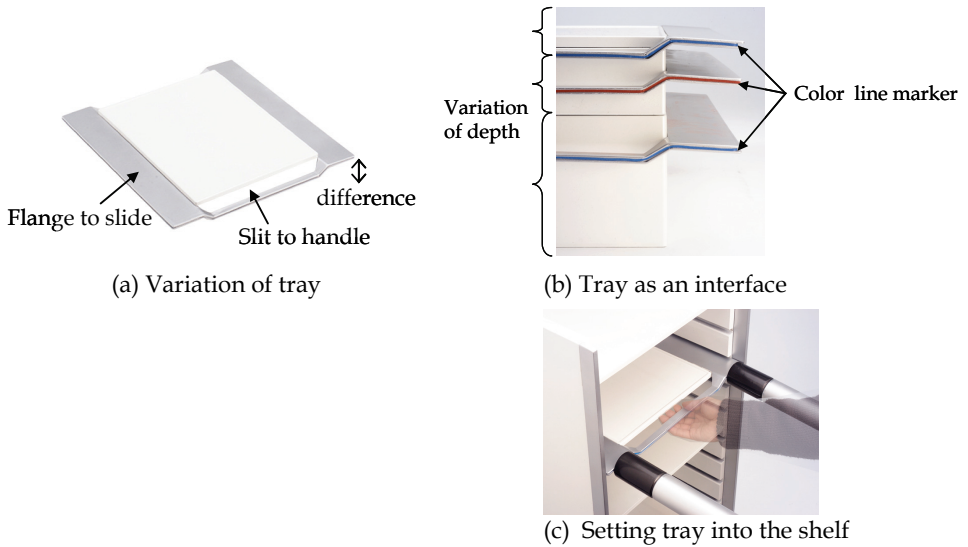
Fig. 5. Variation of tray

b) The shelf and the tray

The robot handles the tray unit that contains categorised objects. There are various kinds of daily necessities in our life. For the robot, handling each of them is not practical. It is equipped with the tray as a common interface and the user can put things on the tray. Fig.4 shows the studied design of the storage shelf that is shared by the robot and the user. This shelf consists of a frame and tray-shaped drawers as shown in Fig. 5. The user has only to set things on the tray and the robot delivers it to the shelf. There are various trays, some are flat, others have a profile, and the interface of the tray is designed to be suitable for both humans and robots. It is an interface design of working target. The shelf has no door so that robot can easily access it. The edge of the tray is painted or lighted in different colour to be distinguished easily by both the robot and the user. The task of carrying things at home is supported by sharing the shelf, easy to access and use by both.

c) The robot

Fig. 6. shows the mock-up of the designed robot that carries things from room to room at home. It moves with driving wheels, and two arms that can be folded in its body. The end of the arm is equipped with a gripper hand and a hand-eye camera that recognizes colour marker of the edge of the tray. The task is accomplished by moving with the tray set on the top of robot's body. Fig. 7. shows the sequence of the robot approaching the shelf and putting the tray on it. Fig. 8. shows the transportation scene with floor markers. The markers on the floor or wall are used as landmarks to show the robot a certain point or a direction of movement. Colour variations, geometric design patterns, and LED markers would also constitute a landmark. Combining landmarks and floor/wall designs can give advantages to the user. They can show the user the sequence of the robot's activity. Markers are helpful for both the robot and the user. It is an environmental interface design
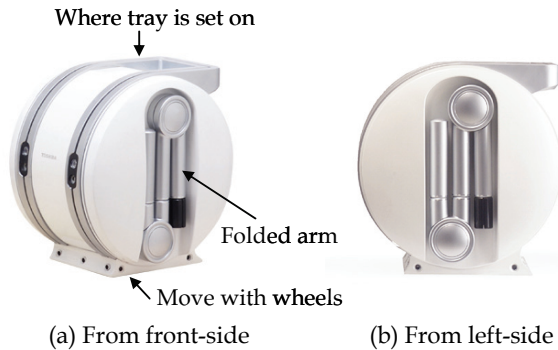
Fig. 6. Mock-up of conceptually designed porter robot
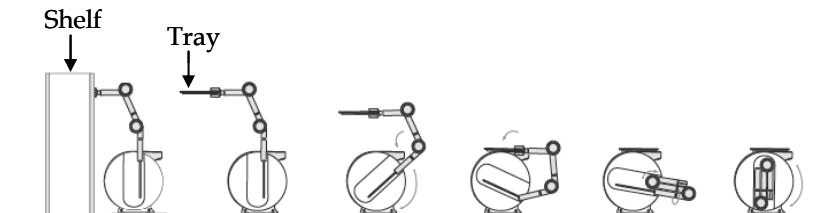
Fig. 7. Sequence of setting a tray on robot with its extended arm

Fig. 8. Robot in movement, guided by line-marker

5. Introduction of UDRob™ to a real robot

Home robots are to be as small and light as possible considering the production costs and safety. If a robot is large, it becomes a big obstacle in a room at home. Fig. 9 shows the robotic interface home appliance ApriAlpha™ developed by Toshiba (Yoshimi et al., 2004), (Matsuhira et al., 2005). ApriAlpha™ controls networked home electric appliances such as TV, room lights, and air-conditioners by direction of its user's voice, and gets the information like weather forecast from the internet to tell it to the user. Table 1 shows the main specification of ApriAlpha™. It was originally developed as a robotic interface of information, and its size seems to be reasonable for this purpose. Although the robot is too small to carry our daily necessities, we can redesign it as a porter robot with the concept of UDRob™ as mentioned in the previous section. Among the shelf, the robot, and its user, the trays are the common interfaces designed for not only human but also robot and shelf, as shown in Fig. 10. Interface design is always to be considered in the environment, e.g., UDRob™.

Fig. 9. ApriAlpha™

Height	420 mm
Diameter	380 mm
Weight	9.5 kg
Velocity	0.5 m/sec
Power source	Li+ Battery
Driving wheels	2

Table 1. Specification of ApriAlpha™
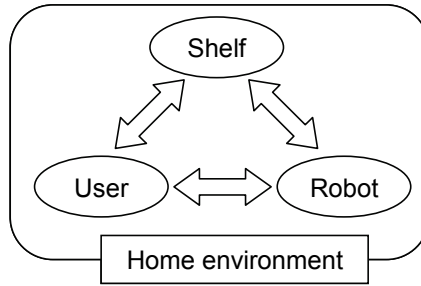
Fig. 10. Relation of each element in home environment

a) Interaction between Tray and Robot

Where should be the interface placed between the tray and the robot? If you want the robot to carry things, they will be set on the carrier, and it can be pushed, tugged, or mounted by the robot. Among them, mounting things on the robot is selected so that the robot can move smoothly without controlling the carrier's motion. Fig. 11 shows the configuration of extendable arms. The arms are stored on both sides of the body. For making a delivery, extendable arms are designed to support the tray.

b) Interaction between Tray and Robot or User or Shelf

A self-positioning mechanism of the tray to the extended arm is developed to make it easier to place the tray on the robot. Fig. 11 (b) shows the sketch of it. Even if the tray is misplaced on the extended arms by humans or by the transfer mechanism of the shelf, the tray is automatically shifted to a stable position guided by a cam mechanism that consists of rollers and curved shape rails. The mechanism absorbs both the misalignment and the impact force during placement. The A4-size-tray is enough for carrying daily necessities such as wallet, pass, cellular phone and envelopes.
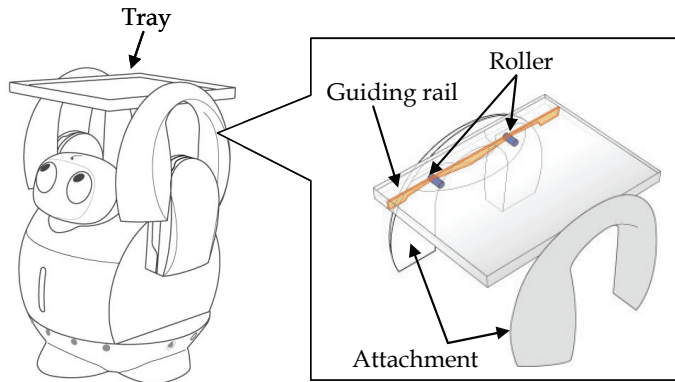
(a) Extendable arm to support tray   (b) Self-positioning mechanism

Fig. 11. Robot supporting the tray with extendable arm

c) Interaction between User and Robot

The extendable arms are designed not only for the support of the tray but also as the interface to the user. The robot is too short to serve the user. The arms can extend up to about 0.6 m in height to make it suitable for a user sitting on a chair to access the tray. Fig. 12

shows the relationship between the robot and the user. Fig. 13 shows the image of serving a light meal.
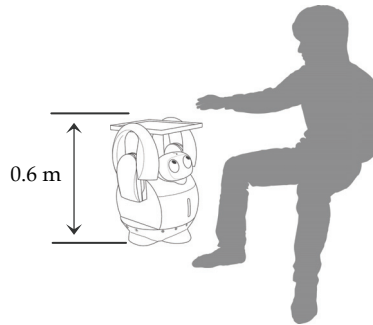
Fig. 12. Robot-User interaction

Fig. 13. Image of ApriAlpha™ serving a light meal

6. Conclusions

Robots are expected to provide help to those who need it in the aging society like Japan. Recently various kinds of home-use robots have been developed to support people in their daily life. However, robot's tasks are hindered in many different ways. Universal Design with Robots (UDRob™) has been proposed to cope with these problems. Interface designs for mobility, handling, and image processing are especially considered as a basic concept of UDRob™. Conceptual design of the robot system based on UDRob™ has been described here and it helps the realization of coexistence between robots and humans. Arrangement of environmental design has the potential of tremendously improving a robot's activities. However, that arrangement should be also useful to its user. The point of UDRob™ is not on requiring a special design for robots but on arranging a common design for humans and robots. As a result, it will improve the quality of our life. Collaboration between designers and robotics engineers are expected to be intensifying in the near future.

7. References

Amada, T. and Kanazawa, H. (2004). Microphone Array Technique for Speech Recognition, *Toshiba Review*, Vol.59, No.9, pp.42-44, 2004 (in Japanese) .

Asano, F. (2004). Separation of Sound, *J. SICE*, Vol.43, No.4, pp.325-330, 2004 (in Japanese).

Brandstein, M. (1999). Time-delay estimation of reverberated speech exploiting harmonic structure, *J. Acoust. Soc. Am.*, 105, 5, pp.2914-2929, 1999.

Brandstein, M. and D. Ward, D. (2001). *Microphone Arrays*, Springer Verlag, Berlin, 2001.

Chong, N. Y.; Hongu, H.; Ohba, K.; Hirai, S.; Tanie, K. (2004). A Distributed Knowledge Network for Real World Robot Applications, *Proc. of the 2004 IEEE/RSJ Int. Conf. on IROS (IROS2004)*, Sendai, Japan, pp.187-192, 2004.

Japan Robot Association (2003). Cleaning, Caretaking and Security Robot, *Robot*, No.152 (in Japanese).

Japan Robot Association (2004). Measures for the Promotion of the Robot Industry by the Sates or the Local Government, *Robot*, No.159 (in Japanese).

Japan Robot Association (2005). Daily Life Support Robots, *Robot*, No.167 (in Japanese).

Japan Robot Association (2006). Safety of Service Robots, *Robot*, No.168 (in Japanese).

Kim, B. K.; Miyazaki, M.; Ohba, K.; Hirai, S. and Tanie, K. (2005). Web Services Based Robot Control Platform for Ubiquitous Functions, *Proc. of the 2005 IEEE International Conference on Robotics and Automation (ICRA2005)*, Barcelona, Spain, pp.703-708, 2005.

Kim, B. K.; Tomokuni, N.; Ohara, K.; Tanikawa, T.; Ohba, K.; Hirai, S.(2006). Ubiquitous Localization and Mapping for Robots with Ambient Intelligence, *Proc. of the 2006 IEEE/RSJ Int. Conf. on IROS (IROS2006)*, Beijing, China, pp.4809-4814, 2006.

Hagita, N.; Kuwabara, K.; Tokuda, H. (2005). Introduction to Network Robots, Proc. Of Workshop on Network Robot Systems, *IEEE International Conference on Robotics and Automation (ICRA2005)*, Barcelona, Spain, pp.1-4, 2005.

Hasegawa, T. & Muarkami, K. (2006). Robot Town Project: Supporting Robots in an Environment with Its Structured Information, *Proc. of The 3rd International Conference on Ubiquitous Robots and Ambient Intelligence*, pp. 119-123, Seoul, Korea, 2006.

Hosoda, Y.; Egawa, S.; Tamamoto, J.; Yamamoto, K.; Nakamura, R. and Togami, M. (2006). Basic Design of Human-Symbiotic Robot EMIEW, *Proc. of the 2006 IEEE/RSJ Int. Conf. on IROS (IROS2006)*, Beijing, China, pp.5079-5084, 2006.

Tanie, K. & Matsuhira, N. (2007). Common Platform Technology for Next Generation Robots, *IEEE International Conference on Robots and Automation (ICRA2007)*, Workshop of Network robot systems, Roma, Italy, 2007.

Kanda, T.; Shiomi, M.; Perrin, L.; Nomura, T.; Ishiguro, H. and Hagita, N. (2007). Analysis of People Trajectories with Ubiquitous Sensors in a Science Museum, *Proc. of IEEE International Conference on Robotics and Automation (ICRA2007)*, pp.4846-4853.

Matsuhira, N. & Ogawa, H. (2004). A Trend of Developing Home Robot Leading High Technology, *Toshiba Review*, Vol.59, No.9, pp.2-8, 2005 (in Japanese).

Matsuhira, N.; Ozaki, F.; Ogawa, H.; Yoshimi, T. and Hashimoto, H. (2005). Expanding Practicability of Home Robots in cooperation with Networked Appliances, *Workshop on Advanced Robotics and its Social Impacts (ARSO)*, Nagoya, Japan, 2005.

Matsuhira, N.; Ogawa, H.; Yoshimi, T. and Mizoguchi, H. (2005). Development of Life Support Robots that Coexist in Harmony with People, *Proc. of the 36th International Symposium on Robot (ISR2005)*, TU 4H-5, Tokyo, 2005.

Nakadai, K.; Hidai, K.; Mizoguchi, H; Okuno, H. G. and Kitano, H. (2003). Real-Time Human Tracking by Audio-Visual Integration for Humanoids, *J. RSJ*, Vol.21, No.5, pp.57-65, 2003 (in Japanese) .

Ohba, K.; Onda, H.; Tanikawa, T.; Kim, B. K.; Tomizawa, T.; Ohara, K.; Liang, X.; Kim, Y. S.; Do, H. M. and Sugawara, T. (2007). Universal-design for environment and manipulation framework, *The 8th SICE System Integration Division Annual Conference (SI2007)*, 3A2_5, Hiroshima (in Japanese).

Yoshimi, T.; Matsuhira, N.; Suzuki, K.; Yamamoto, D.; Ozaki, F.; Hirokawa, J. and Ogawa, H. (2004). Development of a Concept Model of a Robotic Information Home Appliance, ApriAlpha, *International Conference on Intelligent Robots and Systems (IROS)*, pp.205-211, Sendai, Japan, 2004.

Wada, T. (2004). A Design of Home Robot, *Toshiba Review*, Vol.59, No.9, pp.15-19, 2004 (in Japanese).

Thrun, S. (2001). A Probabilistic On-Line Mapping Algorithm for Teams of Mobile Robots, *The Internatinal Journal of Robotics Research*, Vol.20, No.5, pp.335-363, 2001.

Sugano, S.; Shirai, Y. and Chae, S. (2006). Environment Design for Human-Robot Symbiosis - Introduction of WABOT-HOUSE Project -, *International Symposium on Automation and Robotics in Construction 2006 (ISARC2006)*, pp152-157, Tokyo, Japan, Oct.3-5

Sugawara, T.; Kenichi Ohara, K.; Lee, J. H.; Tomizawa, T.; Kim, Y. S.; Liang, X. F.; DO, H. M.; Kim, B. K.; Onda, H.; Tanikawa, T.; Sumi, Y.; Sonoyama, T. and Ohba, K. (2007). The Study on the Support for Robot due to Physically Environmental Structure, *The 8th SICE System Integration Division Annual Conference (SI2007)*, pp.125-126, Hiroshima (in Japanese).

Development of Common Platform Technology for Next-Generation Robots

Tomomasa Sato*, Nobuto Matsuhira* and Eimei Oyama*
The University of Tokyo, Toshiba Corp. & AIST
* Council for Science and Technology Policy
Coordination Program of Science and Technology Projects (Next Generation Robots)
Japan

1. Introduction

Various research and development projects for robots have been carried out by a large number of research groups. However, many research groups spend a lot of time and money to develop the basic robotic hardware and software for each application field, which might be used as common infrastructure. In addition, apart from the market for industrial robots, markets for service robots are few and tiny.

The application-independent common technology for robots of any kind, which can be used by most robot developers and engineers, is defined as "common platform technology." In order to increase the efficiency of future robot research and development, the establishment of the common platform technology is a matter of urgency. The common platform is relevant to recent topics in robotics, including robotics structured environment, networked robot, robot middleware, reorganized dynamic simulator, and so on.

In order to develop next-generation robots capable of working in complex and dynamic environments, and not just in factories, the "information-structured environment" is a key technology. For example, the position information of the robots, objects and the humans in the working environment are required for any robotic application and might make the working environment closer to the structured environment, such as those in factories, using information technology (IT). In the information-structured environment, the robots can work as if they were in factories well organized in terms of the working environment for robots. If this position information can be acquired on the basis of a common infrastructure, then research or development of the next-generation robots can be executed more efficiently. Although Smart Room (Pentland, 1995), Aware Home (Kidd et al., 1999), Intelligent Space (Lee & Hashimoto, 2002; Lee et al., 2004), and Robotic Room (Sato et al., 2004) related to the information-structured environment have been developed, they are not considered as the robotic infrastructure.

On the other hand, software technology applied as the common platform technology, such as a robot simulator, is also an important issue. Although there is a lot of robot software developed in the world and several robot simulators (Gerkey et al., 2003; Michel, 2004; Nesnas et al., 2006; Gates, 2007; Lee et al., 2007), there are few simulators that guarantee the consistency of software modules. The simulator would be able to simulate the performance

of a robot synthetically using developed software modules, to guarantee their connectivity, reusability and would be extendable.

In Japan, information-structured environments as an environmental platform and a robot world simulator as a software platform are ongoing projects within the framework of the common robot platform technology for robotics applications of any kind. These projects are the first challenge to arrange the robot research and development infrastructure in the world. Here, we introduce these projects (Izawa, 2006; Tanie et al., 2006).

2. The next-generation robot coordination program

First of all, the background of these projects is explained. In Japan, the Council for Science and Technology Policy defines themes of national and social importance worthy of promotion, and coordinates related ministries in reconsideration of the conventional separate measures of individual ministries. The Coordination Program of Science and Technology Projects was established in 2005 to promote these themes, in order to strengthen coordination while eliminating duplication of related measures (Izawa, 2006; Tanie et al., 2006). The following eight themes were addressed initially: post-genome, emerging and re-emerging infectious diseases, ubiquitous networks, next-generation robots, biomass utilization technology, hydrogen & fuel cell, nano-bio-technology, and local science & technology clusters. In 2007, six new themes were added to these eight themes. Focusing on "common platform technology for next-generation robots," the core missions of the Next-Generation Robot Coordination Program are to promote the robot research and development of each ministry and to provide society with basic infrastructure technology for robots, thus enabling the development of services to be provided by various robots. Additionally, the stated objective to "lead the world in developing core robot technology useful in everyday life, both in the home and in urban environments" is a strategically important concern of the "3rd Term Science and Technology Basic Plan of Japan" for 2006-2010. Developing robots useful to society is also a concern of "INNOVATION 25," a governmental long-term strategy indicator.

3. Common platform technology

There have been various research and development projects of robots in individual ministries. After examination of these projects, it was found that required research and development were conducted for every application, and that certain ministries have clearly indicated technology that might be used as a common infrastructure. It follows that the technology capable of common use by robot developers and engineers when conducting research and development of robots was defined as application-independent "common platform technology." It was raised as an issue to be urgently addressed through the Next-Generation Robot Coordination Program in order to increase the efficiency of future robot development. Specifically, there are two aspects; namely, "information-structured environment" and "basic software for robot development," which are each discussed below (Izawa, 2006; Tanie et al., 2006).

3.1 Information-structured environment as an environmental platform
Although a robot's working environment will change depending on its application, technology using sensors to measure a robot's own position in relation to its environment is

required for any application. Various developments are underway in regard to such technology. If position information can be acquired on the basis of a common structure, then research and development of robot systems for specific applications can be executed more efficiently.

Furthermore, in the near future, robots themselves are not only expected to be equipped with intelligence and software, but also to utilize knowledge of their environment through integration with other technologies such as IT, ubiquitous computing, network communication technology, and use of GPS and RFID tags. These technologies have been developed as "networked robotics." The importance of "environmental information structuring technology" (that is, the embedding of programs, information, and knowledge for robots in the environments in which they operate as common infrastructure technology for developing various robots) is expected to increase. Therefore, in utilizing IT and similar technologies from this point onwards, it becomes necessary to research and develop standard models for information-structured environments. Two questions arise to be discussed: firstly, how should communication among embedded apparatus and robots, both indoors and outdoors, be standardized? Secondly, what kind of equipment (such as RFID tags) and information (such as environmental maps) should be embedded in the environment? The concept of the information-structured environment is shown in Fig. 1.
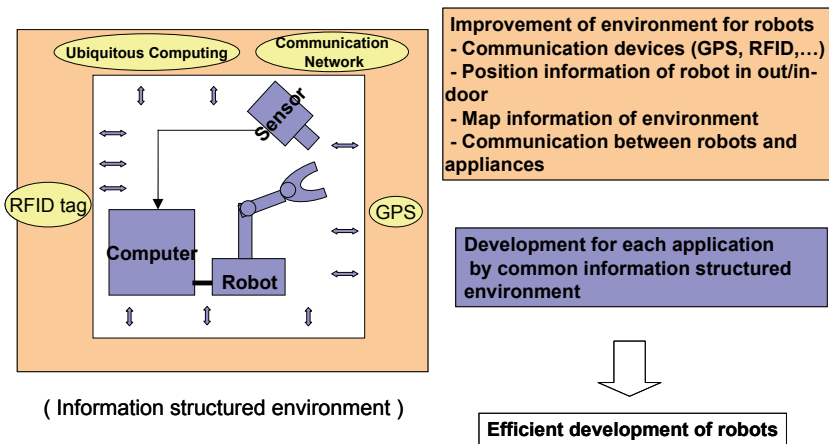
Fig. 1. Concept of Information-Structured Environment

3.2 Basic software for robot development as a software platform

At the same time, various types of software are needed in order to develop robots. Since an effective mechanism enabling robot researchers and engineers to share such software has not been established, they have each developed their own software independently in different research projects and research organizations. However, similar software is often developed over multiple projects and a considerable amount of shared software exists. Moreover, there has been no means of evaluating the performance of a developed robot in comparison to that of other robots in the same program environment. It is therefore essential to build social infrastructure that enables the provision of software for shared use. To that end, research and development of a robot world simulator to serve as a common management system for robot software are needed. Based on distributed object technology,

such a simulator would be able to simulate the performance of a robot synthetically in terms of its hardware, sensors and sensing functions, control structure, and other functions, including work environment, environmental objects, etc. Furthermore, it would be necessary for the simulator to guarantee the connectivity and reusability of various kinds of developed robot software, as well as its ability to accumulate functions and to be expandable. The concept of a robot world simulator is shown in Fig. 2.
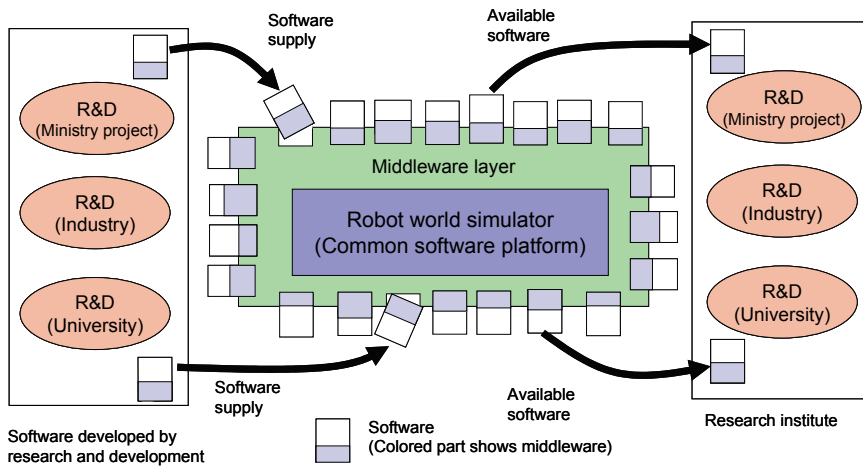
Fig. 2. Concept of Robot World Simulator (Software Platform)

4. Toward realization of a common platform

Aimed at the "effective and efficient promotion of coordination programs for science and technology projects" as supported by the special coordination fund for Promoting Science and Technology from the Japanese Ministry of Education, Culture, Sports, Science and Technology (MEXT), and toward further development of a common platform technology, the four three-year projects described below are being implemented starting from 2005 and 2006. (The organization responsible for each project is indicated in parentheses.)

4.1 Structuring Robot Town 2005-2007 (Kyushu University)
The objective of the Robot Town Project is to develop a common platform enabling robots to work in the ordinary environments encountered in everyday life. In the platform, sensors and RFID tags are embedded and connected with a network to support robotic activities (Hasegawa et al., 2006, 2007; Kurazume et al., 2007). Fig. 3 depicts a conceptual diagram of the robot town. Fig.4 illustrates the daily life support by robots in/out of a house in the robot town.

To achieve autonomous robotic activities in such an environment, distributed sensors such as cameras and laser range finders (LRFs), and RFID tags connected with a network are distributed in the environment. Real-time data from the sensors and the robots are integrated by the Town Management System (TMS) together with GIS and other databases as shown in Fig.5. Robots in the platform can obtain miscellaneous information required for their activities from the TMS. The information includes the positions and the motions of

humans, cars, robots and obstacles in the platform. This platform is built in Fukuoka Island City and is opened in 2008.

Robot Town: collaborated work with robots and humans

GPS

Wireless LAN

Town management center

Distributed vision system

Intelligent car

RFID tag

Intelligent marker

Supporting experiment in hospital

Information structured store, building, hospital

Supporting experiment in/out of house
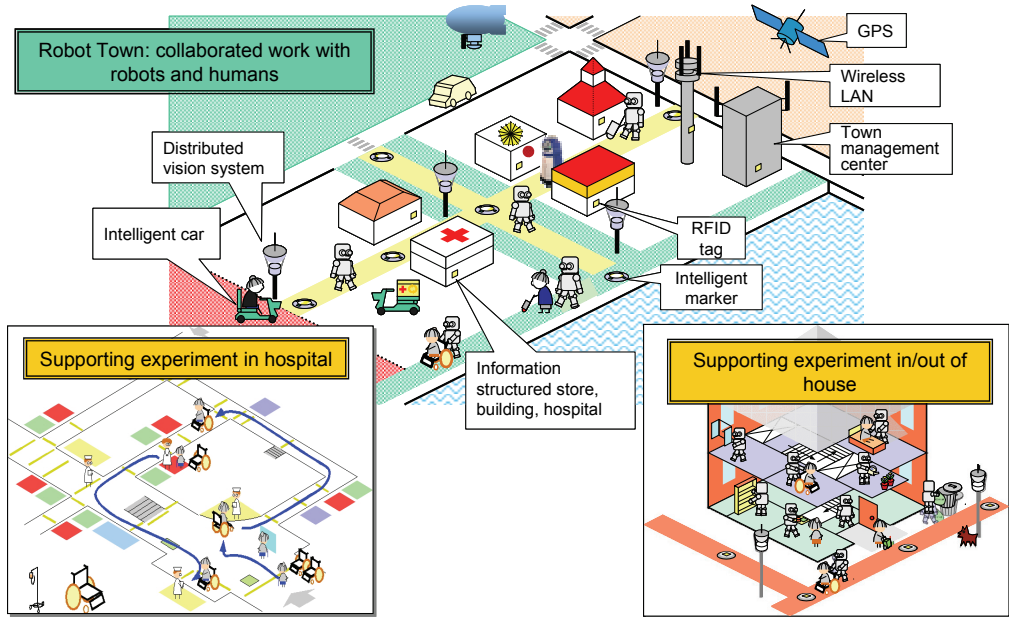
Fig. 3. Concept of Robot Town

Operation of a welfare Bed for aged person

Open/close of door/window

R/D House

Kitchen task Dinning Setup

Indoor-Outdoor task

Embedded RFID tag

Distributed Vision

Entrance/ Departure
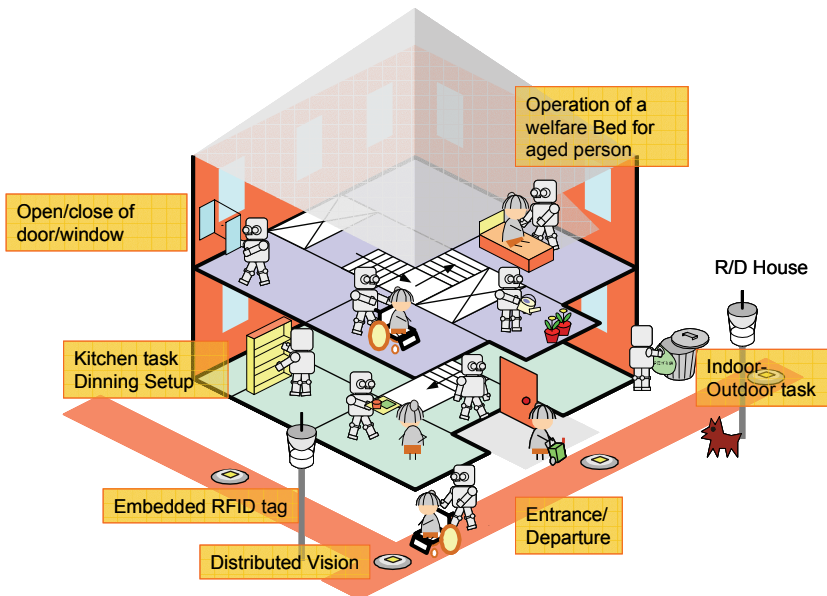
Fig. 4. Daily Life Support by Robots in/out of a house in Robot Town
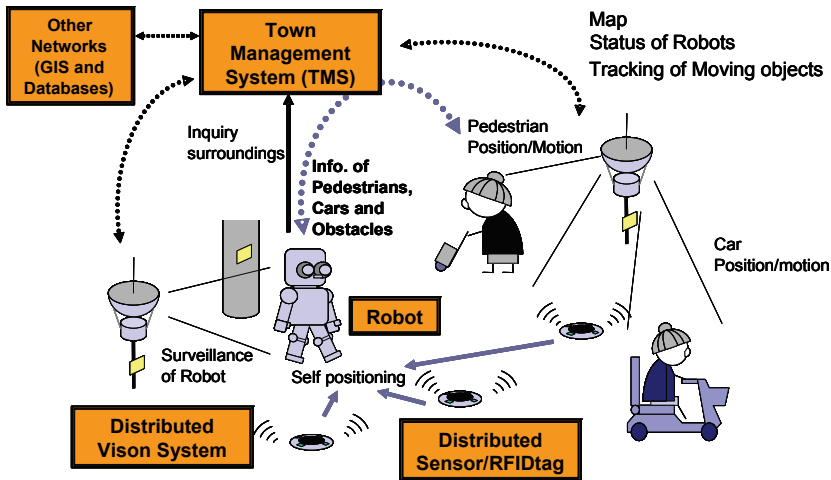
Fig. 5. Information Structuring of Robot Town

Fig. 6. Experimental House and Its Surroundings in Robot Town

Fig. 6 shows the experimental house and its surroundings. A number of open experiments have been conducted since January 2007. In the experiment in January 2007, when the parked car was detected by the sensor mounted on the house, the event of the arrival of the car was transmitted to the TMS and the TMS directed the command to the wheelchair robot to go to the parking area to bring the baggage automatically using RFID tags shown as cards mounted on the road as shown in Fig. 7. In the experiment in January 2008, when the wheelchair robot was called at the experimental bus stop, the robot went automatically to the person demanding the robot, using camera image from distributed vision system and RFID tags on the road as shown in Fig.8. Inside the house, two types of robots carried out the object-handling from the handling robot to the porter robot using the TMS as shown in

Fig. 9 (a). The TMS knows the position of the robots from distributed sensors and each object with tags. Fig. 9 (b) shows the real-time positions of the human and robots in the house.

(a) Start from the Base   (b) Carry the Baggage

(c) Move along the Road   (d) Return to the Entrance of the House

Fig. 7. Scenes of the Experiment of Wheelchair Robot in January 2007
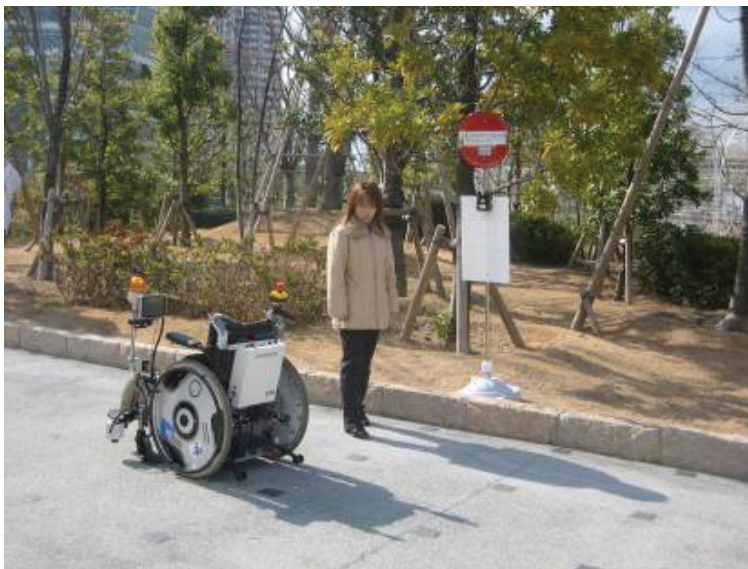
Fig. 8. Scenes of the Experiment for Demanding the Wheelchair Robot in January 2008

(a) Dual Arm Robot and Porter Robot   (b) Display of Real-Time Positions of Humans
                                            and Robots
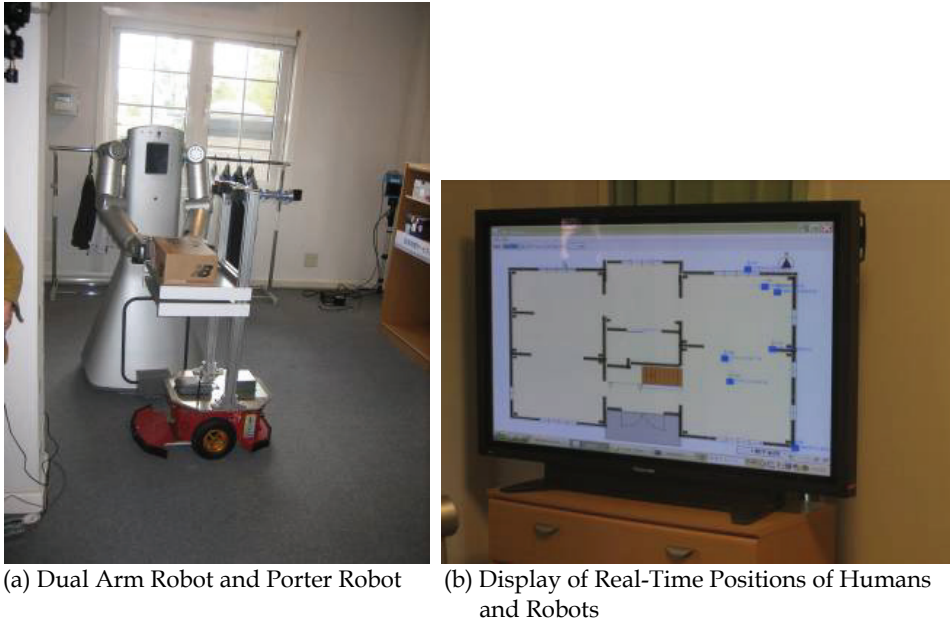
Fig. 9. Scenes of the Experiment in the Experimental House

4.2 Structuring of human behavior measurement 2006-2008 (Advanced Telecommunications Research Institute International (ATR))

A new framework for structuring environmental information based on humans' positions will be proposed. Structuralizing environmental information based on precise positions of humans who move in and out of buildings is one of the most important issues for realizing robots capable of providing various services (Kanda et al., 2007; Oike et al., 2007; Glas et al., 2007). For acquiring such information, the framework consists of three fundamental technologies: "Real-time robust measuring and recording of humans' positions," "Structuring environmental information based on the relationship between obtaining spatial information and the history of humans' positions," and "Constructing a common platform to provide the structured environmental information." The robotic service applications utilize the structured environmental information as shown in Fig. 10. It has the four-layered model, consisting of sensor, segment, primitive and service-and-application layers, to give the meanings in terms of space and behaviour for the robot services such as guidance, navigation and introduction. Fig. 11 shows the human position measurement system of the platform.

This platform is to be built in the Kansai area. There are two platforms planned; one is located at the lobby of the NICT (National Institute of Information and Communications Technology) Keihanna Building and another is UCW (Universal City Walk, Osaka). The former is named the Keihanna platform. The later is named the UCW platform. Fig. 12(a) shows the outside of the Keihanna platform. Fig. 12(b) shows the equipment of the platform.

Since the UCW is located at the front of Universal Studio Japan (USJ), many people walk through there. Fig. 13(a) and Fig. 13(b) show the outside of the UCW platform and the equipment of the platform, respectively. The sensing system composed of several cameras and laser rangefinders (LRFs) was set around the open space, and more than ten people were detected simultaneously by the sensing system and their behaviour was labelled as shown in the display. In the experiment in January 2008, the shop guidance by the robot was carried out in the UCW as shown in Fig. 14.
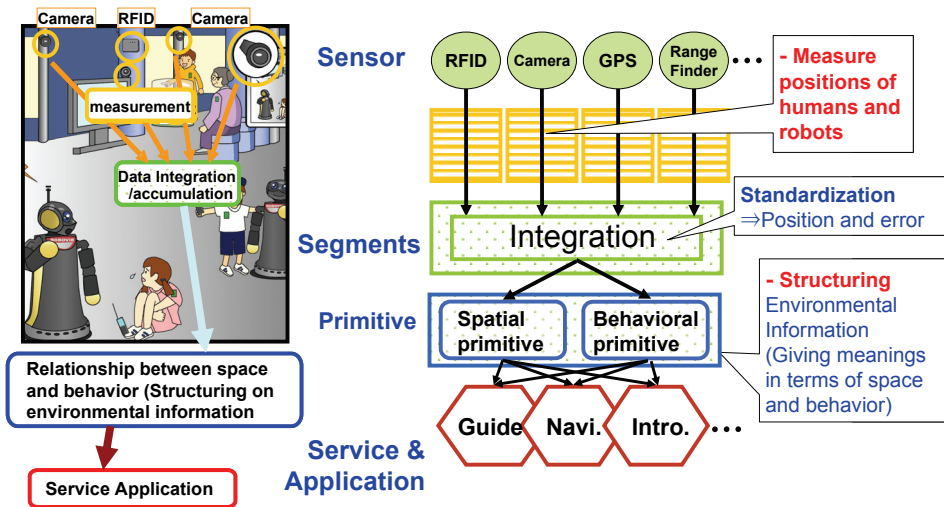
Fig. 10. Four-Layer Model for Structuring Environmental Information
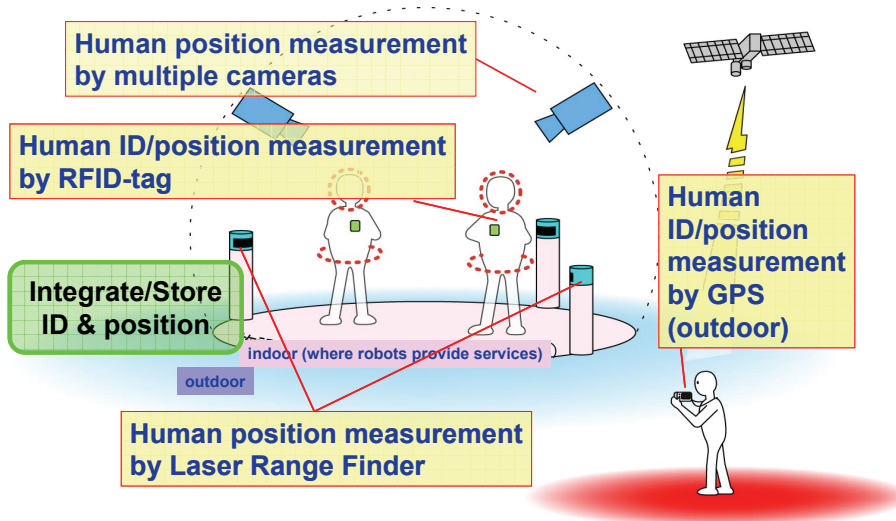
Fig. 11. Human Positioning System

(a) NICT Keihanna Building

(b) Lobby of NICT Keihanna

Fig. 12. Keihanna Platform

(a) Outside UCW

(b) Equipment of UCW Platform

Fig. 13. UCW Platform

(a) Communication Robot talking to a Person

(b) Display of Real-time Positions of Humans and Robots

Fig. 14. Scenes of the Experiment in UCW Platform

4.3 Structuring of robot task 2006-2008 (AIST)

The goal of this project is to establish a common platform for robot infrastructure, i.e. universal design for next-generation robots, enabling various tasks in different environments by different robots (Ohara et al., 2008; Kamol et al., 2007; Sugawara et al., 2007). Fig. 15 illustrates the conceptual diagram of the universal design. To construct the robot platform, an environmental and robot manipulation framework is to be developed. The framework will be considered in terms of its physical level and intelligence level. This platform is to be built in the Tsukuba and Kanagawa areas. The prototype of the platform is built in AIST, Tsukuba.

The prototype platform is equipped with RFID tags, cameras and LRFs. Furthermore, Pseudlite, i.e. indoor GPS, Starlite, i.e. infrared LED transmitter, and other sensors are integrated into the platform (Ohara et al., 2008; Kamol et al., 2007; Sugawara et al., 2007). Fig. 16. illustrates the distributed sensors for robot localization in the platform.
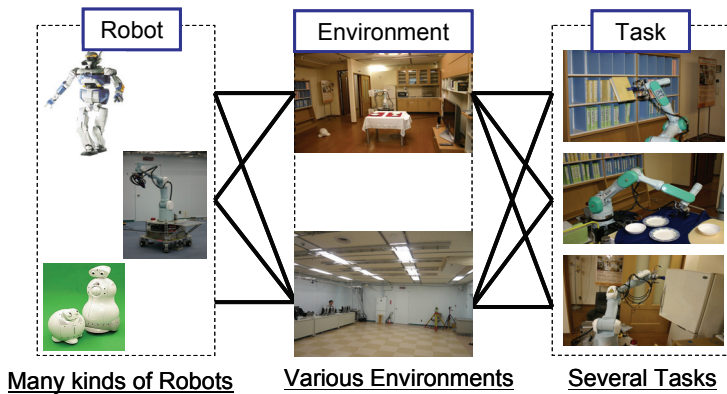
Fig. 15. Concept of Universal Design for Next-Generation Robots
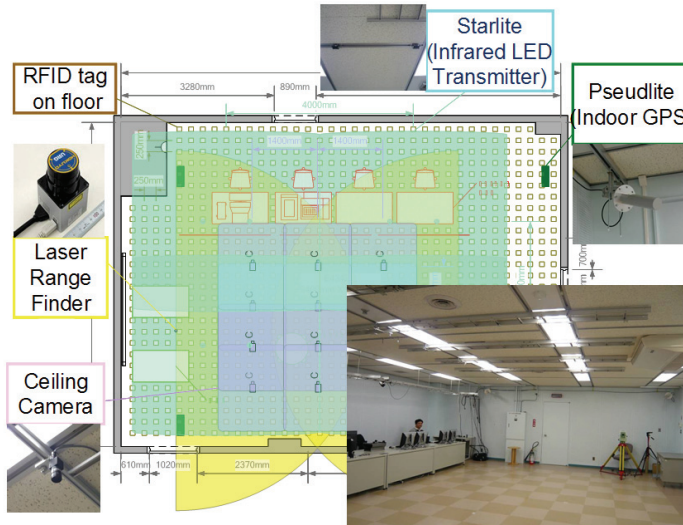
Fig. 16. Distributed Sensors in the Prototype of the Platform in Tsukuba

The interface of each sensor is standardized by RT (Robot Technology) middleware that has been proposed as a standard middleware for robotic technologies. The RT middleware is described in Section 4.4. The fine structure of a sensor is held as a profile. Each robot in the platform can obtain its position information in the same manner. It should be noted that the project is related to the standardization of the robotic localization service (Object Management Group, 2007).

For robotic tasks, distributed RFID tags, which have links to the knowledge database for robotic tasks, and visual markers indicating the knowledge are developed. Fig. 17 shows the concept of the knowledge storage distributed in the information-structured environment. To perform manipulation tasks, fine positioning within 5 mm and knowledge of the object to be handled are necessary. So, sensing strategy is changed depend on the distance and the knowledge is obtained through RFID tags and visual markers. A matrix code, also known as a 2D barcode, is utilized as the marker. The position and orientation of the matrix code is utilized for the visual servo of the robotic arm as shown in Fig. 18(a). The marker is named Coded Landmark for Ubiquitous Environment (CLUE). Since the matrix code of the CLUE is invisible under ordinary lighting, the CLUE has no effect on the design of the object, to which the CLUE is attached. The code emerges in ultraviolet lighting as shown in Fig.18(a). As a physical interface, the universal handle shown in Fig. 18(b) is developed so that a robot is able to handle miscellaneous doors easily. Furthermore, structuring of typical robotic tasks is conducted based on the pick and place task since most robotic tasks are divided into the pick and place tasks.

The experiment was carried out in October 2007. The demonstration task was carried out in which the robot opens the door of the refrigerator, picks up the package and places it on the electric range, and finally places the package on the table.

Fig. 19 shows some scenes of the robotic experiments.
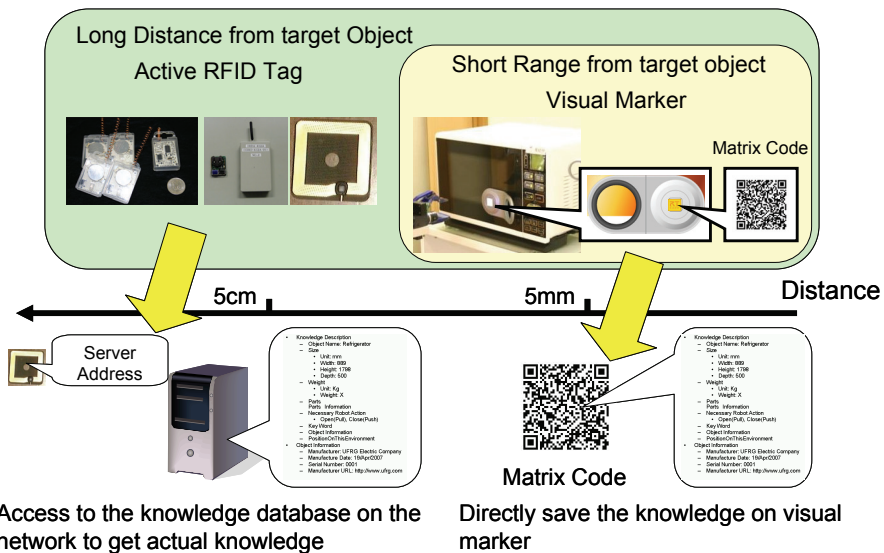
Fig. 17. Distributed Knowledge Storage in Information-Structured Environment

(a) Matrix Code Emersion
     by Ultraviolet Lighting (Left Image)

(b) Door Opening by Universal handle

Fig. 18. Universal Handle with CLUE (Coded Landmark for Ubiquitous Environment)

(a) Dish Handling

(b) Book Handling

(c) Container Handling
    with Universal Handle

Fig. 19. Scenes of Experiments

4.4 Robot World Simulator 2005-2007 (National Institute of Advanced Industrial Science and Technology (AIST))

The objective of this project is to develop a robot simulator composed of distributed object modules (Nakaoka et al., 2007). The distributed object modules are implemented by RT Middleware (RTM). RTM is intended to establish a common platform based on the distributed object technology that would support the construction of various networked robotic systems by the integration of various network-enabled robotic elements called RT Components (RTCs) (OpenRTM-aist, 2007). RTM was adopted as a draft version of International Standard by the OMG in Oct. 2006. Fig. 20 shows the conceptual diagram of the simulator, the real robot and the RTCs. The RTC is a sharable robotic software module. The conceptual diagram of RTM and RTC is shown in Fig. 21.

The name of the simulator is OpenHRP3 (Open architecture Human-centred Robotics Platform 3). OpenHRP3, based on the humanoid robot simulator OpenHRP2 developed by AIST (Kanehiro, 2004), was partially open for limited users from 2007. The simulator will be open for unlimited users from April 2008. The robot world simulator OpenHRP3 will also be open to robot developers as a result of a "distributed-component robot simulator." Fig. 22 shows the user interface of OpenHRP3.

To enhance the dynamics simulation, a forward dynamics algorithm is to be developed and implemented by efficient O(n) and O(log n) algorithms, utilizing parallel computing (Yamane et al., 2006a, 2006b, 2007a, 2007b). Fig. 23 shows the simulation result and the experimental result of the humanoid robot made by OpenHRP3. The tendency of the motion

is almost the same. The simulation results were confirmed by a number of experiments using real robots. This is one of the advanced properties of OpenHRP3 as a robot dynamics simulator. Fig.24 shows a closed loop linkage mechanism, multi-transporter robots, a humanoid robot, a wheelchair and a robot arm as samples of simulations.
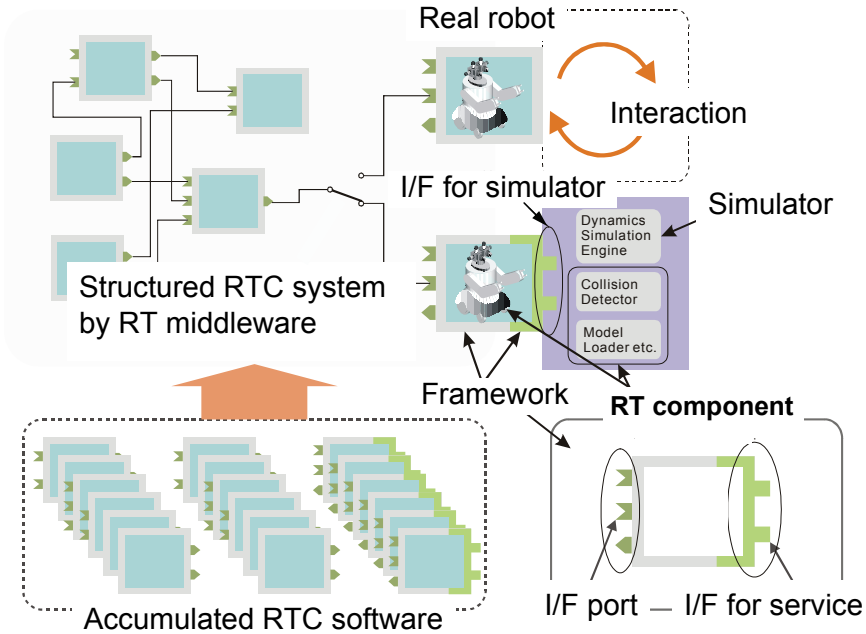
Fig. 20. Simulator, Real Robot and RT Components
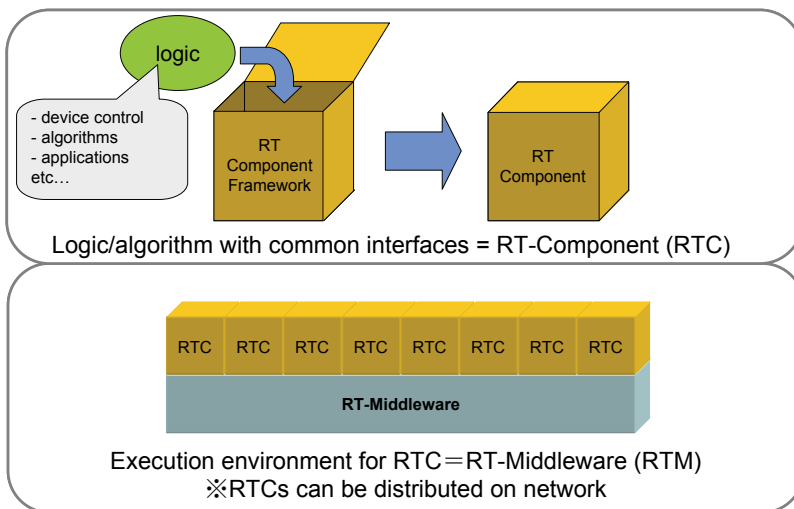
Fig. 21. Conceptual Diagram of RTM and RTC
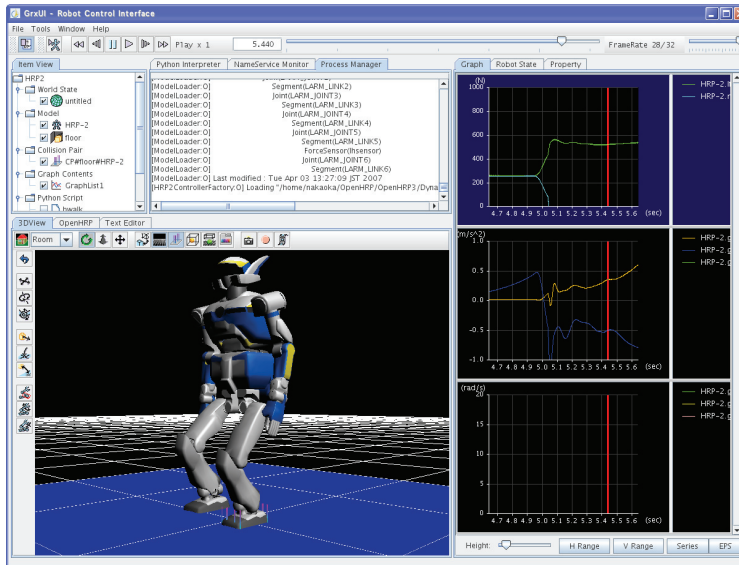
Fig. 22. GUI of OpenHRP3
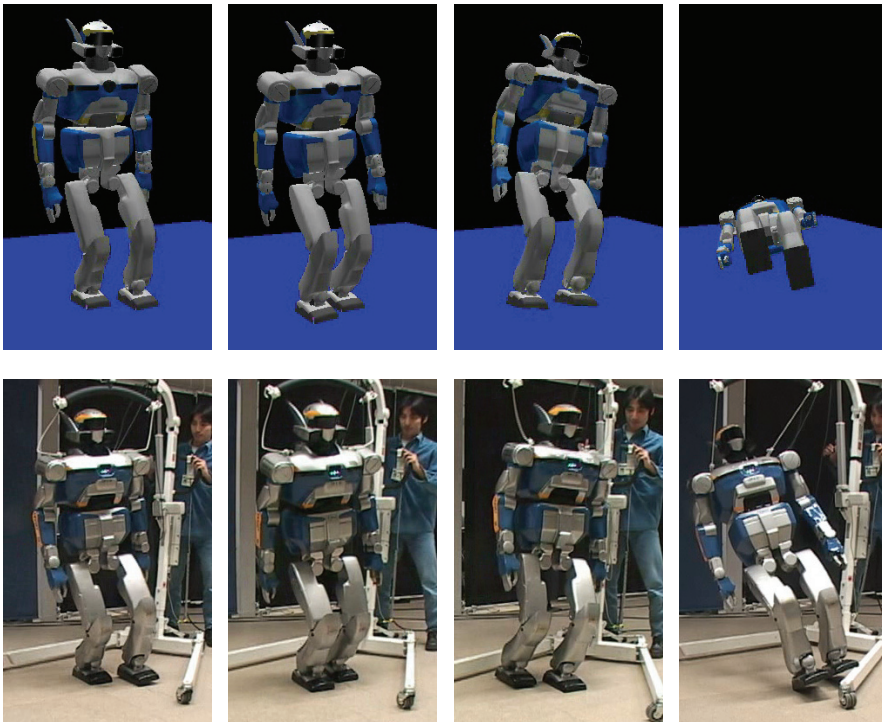
Fig. 23. Simulation Result and Experimental Result
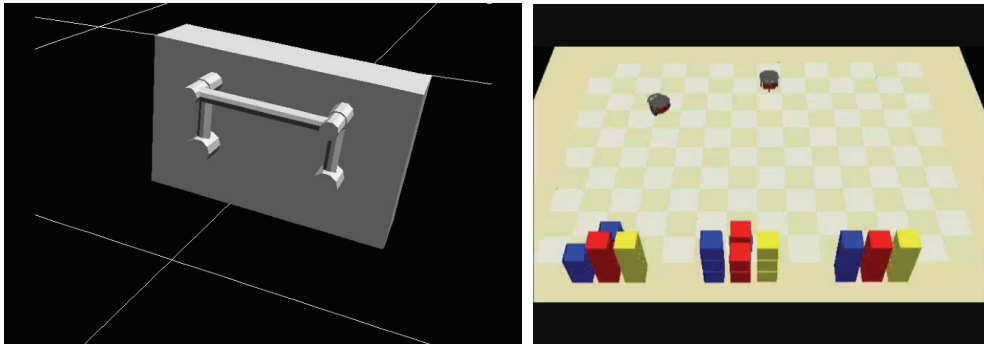
(a) Closed Loop Linkage Mechanism   (b) Multi-Transporter Robots
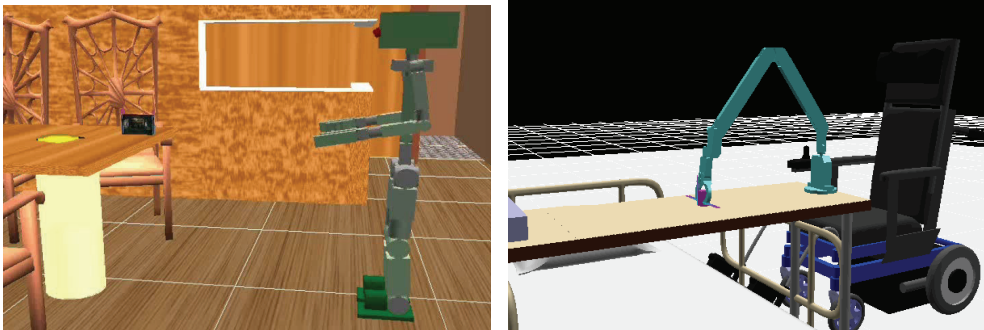
(c) Humanoid Robot   (d) Wheelchair and Robot Arm

Fig. 24. Various Simulation Tasks

4.5 General remarks on the platforms

As indicated above, the aim of these projects is to enable development and construction of a diverse range of environmental information structured platforms, ranging from the structure of a town to the structure of a work-space on a desk. Furthermore, a working environmental platform is to be built and installed, after research is completed, for common use by numerous robot researchers and engineers in the Fukuoka, Kansai, and Kanagawa areas. Additionally, the robot simulator is intended for public release in order to promote the sharing of software. These trials provide robot developers with a tool set which not only provides software usable solely for robot development, but also includes the environment in which a robot works. The overview of the common platforms project is shown in Fig. 25. Table 1 shows the specifications of the three environmental platforms.
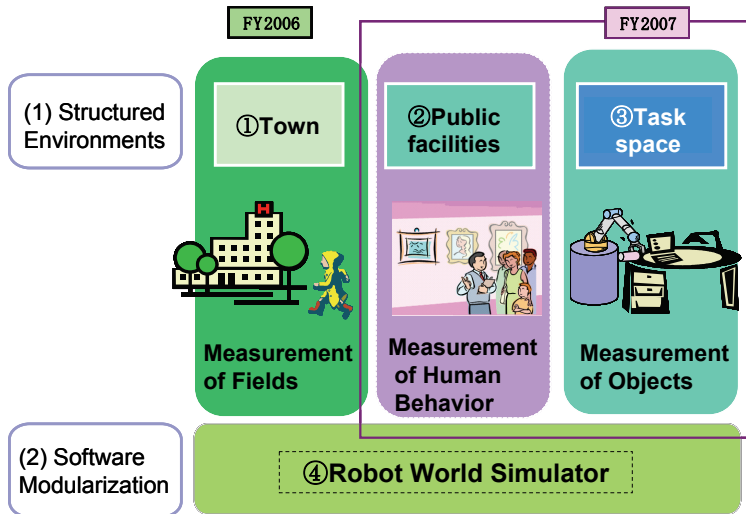
Fig. 25. Overview of Common Platform

PJ	Robot town by Kyushu Univ.	Human behavior measurement by ATR	Robot task by AIST
Measurement accuracy	Same as other Platforms	50 mm	50 mm 5 mm (for manipulation)
Embedded devices	RFID Cameras (distributed camera system) LRF GPS	RFID Cameras LRF GPS	RFID Cameras LRF iGPS (psudo-light)
Middleware	RT-middleware	Cross ML	RT-middleware
Provided function	-Position; robot, moving object -ID information -API	-Position; robot, humans -ID information -API	-Position; robot, objects -API including RT-components
Demonstration	Convey baggage & clothes, Guidance	Human support, e.g., Guidance	Clear the table & book return to the shelf
Open site	Experimental house and roads in Fukuoka Island City	NICT lobby and UCW in Kansai	Experimental house in Kanagawa Robot Park

Table 1. Specifications of Environmental Platforms (not final version)

5. Conclusions

In this paper, the outline of the "common platform technology for next-generation robots" promoted in the coordination program of science and technology projects for next-generation robots were introduced. Two projects, the robot town project and the world

robot simulator project, were completed by the end of March 2008 and some projects will be utilized in the new robotic project on robot intelligence of Ministry of Economy, Trade and Industry (METI). Standardization of information-structured environments and utilization of these platforms are to be discussed. The collaboration among ministries toward international standardization such as RT middleware and robot localization has started. Thus, collaborative robotic activities among ministries are becoming widespread. It is envisaged that robotics research will be accelerated by means of this core platform provided throughout society, which will work to disseminate services utilizing robots and will therefore spur development of the robot industry.

6. Acknowledgments

7. References

Kidd, C.D.; Orr, R.; Abowd, G. D.; Atkeson, C. G.; Essa, I. A.; Blair MacIntyre; Mynatt, E.; Starner, T. E. & Newstetter, W. (1999). The Aware Home: A Living Laboratory for Ubiquitous Computing Research, *Proceedings of the 2nd International Workshop on Cooperative Building 1999 (CoBuild'99)*.

Gates, B. (2007). A Robot in Every Home, *Scientific American*, January 2007.

Glas, D. F.; Miyashita, T.; Ishiguro, H. & Hagita, N. (2007). Laser Tracking of Human Body Motion Using Adaptive Shape Modelling, *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS2007)*, pp.602-608.

Gerkey, B.; Vaughan, R. T. & Howard, A. (2003). The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor System, *Proceedings of the 11th International Conference on Advanced Robotics (ICAR 2003)*, pp. 317-323.

Hasegawa, T. & Muarkami, K. (2006). Robot Town Project: Supporting Robots in an Environment with Its Structured Information, *Proceedings of The 3rd International Conference on Ubiquitous Robots and Ambient Intelligence (URAI2006)*, pp. 119-123.

Hasegawa, T.; Murakami, K.; Kurazume, R.; Senta, Y.; Kimuro, Y. & Ienaga, T. (2007). Robot Town Project: Sensory Data Management and Interaction with Robot of Intelligent Environment for Daily Life, *Proceedings of The 4th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI2007).*

Izawa, I. (2006). The National Promoting Policy for Robot Technology, *2006 Annual Conference of the Robotic Society of Japan (RSJ2006). (in Japanese)*

Kamol, P.; Nikolaidis, S.; Ueda, R. & Arai, T. (2007). RFID Based Object Localization System using Ceiling Cameras with Particle Filter, *Proceedings of The 2nd International Symposium on Smart Home (SH'07).*

Kanda, T.; Shiomi, M.; Perrin, L.; Nomura, T.; Ishiguro, H. & Hagita, N. (2007). Analysis of People Trajectories with Ubiquitous Sensors in a Science Museum, *Proceedings of IEEE International Conference on Robotics and Automation (ICRA2007)*, pp.4846-4853.

Kanehiro, F.; Hirukawa, H. & Kajita, S. (2004). OpenHRP: Open Architecture Humanoid Robotics Platform, *International Journal of Robotics Research*, Vol.23, No.2, pp.155-165.

Kawaji, K.; Wang, Q.; Sasaki, T.; Hashimoto, H. (2007). Multiple Objects Localization in Intelligent Space - Utilizing User Hands Position Information from Position Server - , *Proceedings of The 4th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI2007)*, pp.389-394.

Kurazume, R.; Tobata, Y.; Iwashita, Y. & Hasegawa, T. (2007). 3D Laser Measurement System for Large Scale Architectures Using Multiple Mobile Robots, *Proceedings of The 6th International Conference on 3-D Digital Imaging and Modelling.*

Lee, J. & Hashimoto, H. (2002). Intelligent Space -concept and centents-, *Advanced Robotics*, Vol.16, No.3, pp.265-280.

Lee, J.; Morioka, K. Ando ,N. & Hashimoto, H. (2004). Cooperation of Distributed Intelligent Sensors in Intelligent Environment, *IEEE/ASME Transactions on Mechatronics*, Vol.9, No.3, pp.535-543.

Lee, K.; Kim K.; Shim, H. & Lee, J. (2007). Unified S/W Platform for Ubiquitous Robot, AnyRobot Studio, *Proceedings of The 4th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI2007).*

Michel, O. (2004). Cyberbotics Ltd - WebotsTM: Professional Mobile Robot Simulation, *International Journal of Advanced Robotic Systems*, Vol.1, No.1, pp. 39-42.

Nakaoka, S.; Hattori, S.; Kanehiro, F.; Kajita, S. & Hirukawa, H. (2007). Constrained-based Dynamics Simulator for Humanoid Robots with Shock Absorbing Mechanisms, *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems 2007 (IROS 2007)*, pp.3641-3647.

Nesnas, I. A.; Simmons, R.; Gaines, D.; Kunz, C.; Diaz-Calderon, A.; Estlin, T.; Madison, R.; Guineau, J.; McHenry, M.; Shu, I.. & Apfelbaum, D. (2006). CLARAty: Challenges and Steps Toward Reusable Robotic Software, *International Journal of Advanced Robotic Systems*, Vol. 3, No.1, pp. 23-30.

Object Management Group (2007). Request for Proposal: Robotic Localization Service, OMG document robotics/07-06-25.

Ohara, K.; Sugawara, T.; Lee, J. H.; Tomizawa, T.; Do, H. M.; Liang, X.; Kim, Y. S.; Kim, B. K.; Sumi, Y.; Tanikawa, T.; Onda, H.; Ohba, K. (2008). Visual Mark for Robot Manipulation and Its RT Middleware Component, *Advanced Robotics*, Vol.22, No.3.

Oike, H.; Wada, T.; Iizuka, T.; Wu, H.; Miyashita, T. & Hagita, N. (2007). Detection and Tracking using Multi Color Target Models, *Proceedings of SPIE Optics East 2007.*

OpenRTM-aist (2007). http://www.is.aist.go.jp/rt/OpenRTM-aist/html-en/

Pentland, A. (1995). Machine Understanding of Human Action, *Proceedings of 7th International Forum Frontier of Telecommunication Tech*.

Sato, T.; Harada, T. and Mori, T., Environment-type Robot System "Robotic Room" Featured by Behavior Media, Behavior Contents, and Behavior Adaptation, *IEEE/ASME Transactions on Mechatronics*, Vol. 9, No. 3, pp.529-534, 2004.

Sugawara, T.; Tomokuni, N.; Lee, J.; Tomizawa, T.; Ohara, K.; Kim, B. K. & Ohba, K. (2007). Development of Ubiquitous Mobile Manipulator System with RT-Middleware, *Proceedings of the International Conference on Control, Automation and Systems 2007 (ICCAS2007)*, pp. 2473-2476.

Tanie, K. & Matsuhira, N. (2006). Common Platform Technology for Next Generation Robots, 2N12, DS15, *Proceedings of Annual Conference of Robotics Society of Japan (RSJ2006). (in Japanese)*

Yamane, K. & Nakamura, Y. (2006). Stable Penalty-Based Model of Frictional Contacts, *Proceedings of International Conference on Robotics and Automation (ICRA2006)*, pp. 1904-1909.

Yamane, K. & Nakamura, Y. (2006). Parallel O(logN) Algorithm for Dynamics Simulation of Humanoid Robots, *Proceedings of IEEE-RAS International Conference on Humanoid Robotics*, pp. 554-559.

Yamane, K. & Nakamura, Y. (2007). Automatic Scheduling for Parallel Forward Dynamics Computation of Open Kinematic Chains, *Robotics: Science and Systems*.

Yamane, K. & Nakamura, Y. (2007). Robot Kinematics and Dynamics for Modelling the Human Body, *Proceedings of International Symposium on Robotics Research*, pp. 77-88.

Intelligent Space for Human Centered Robotics

Kazuyuki Morioka, Joo-Ho Lee and Hideki Hashimoto
Meiji University, Ritsumeikan University, University of Tokyo
Japan

1. Introduction

Robot systems for supporting human life are desired recently. For that purpose, the robots need to recognize a human living environment that changes dynamically. However, it is difficult to achieve it only with the sensors carried in a stand-alone robot. Then, the research field on the intelligent environments based on the sensor network has been expanding [Sato], [MacDorman]. The intelligent environments generally utilize many intelligent devices, such as computers and sensors in order to improve the performance of the robots. We proposed the intelligent space (iSpace) [Lee, (2002)] as the human-robot coexistent system. The Intelligent Space is designed to achieve a human-centred service by accelerating the physical and psychological interaction between human beings and robots. The iSpace exploits intelligent devices with processing and networking parts in addition to sensing images from colour CCD cameras. The iSpace with vision based intelligent devices achieves a contact-free wide-area location system without an additional burden to users. Vision based iSpace also has the possibility that the further information can be obtained depending
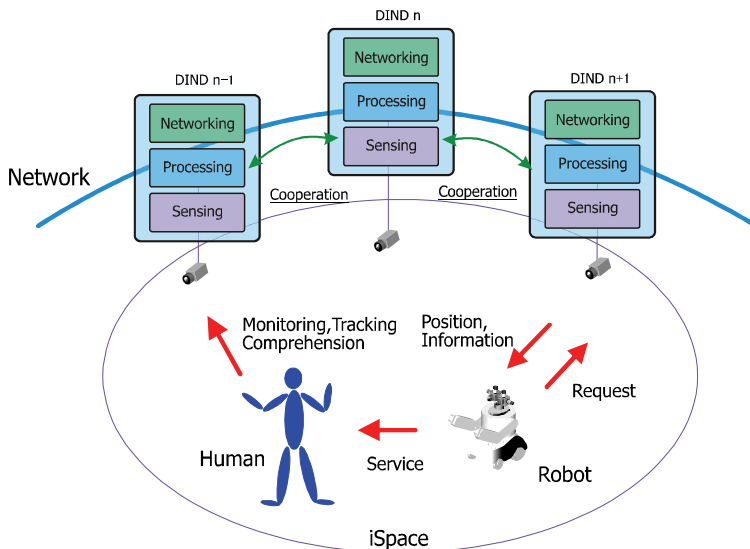
Fig. 1. Intelligent Space

on image processing. Distributed vision devices offer promising prospects as the infrastructure in a human-centred robotic system for these reasons. The most important task of the sensing infrastructure in the iSpace is object tracking and localization of moving objects. Object tracking system in the iSpace is regarded as the multi-camera multi-object tracking system. There are two major problems in the multiple-camera multiple-object tracking system. The first problem is the traditional correspondence problem from frame to frame over time in the image sequence. The second is the object correspondence problem among different cameras in order to achieve seamless tracking. This paper introduces the vision based Intelligent Space and describes colour based target tracking with multiple vision sensors. The iSpace aim to achieve human-centred robotic systems with a tracking system based on vision sensors. To get close each other stably and physically is required for cooperative works between human beings and robots. One of solutions for achievement of such a human-centred robotic system is a human-following robot. A mobile robot for following human beings is also described as an application of tracking system in the Intelligent Space.

2. Intelligent space

2.1 Concept

The Intelligent Space is a space where many intelligent devices are distributed throughout the whole of the space, as shown in Fig. 1. These intelligent devices have sensing, processing and networking functions, and are named distributed intelligent networked devices (DINDs). One DIND consists of a CCD camera for acquiring space information, and a processing computer which has the functions of data processing and network interfacing. These devices observe the positions and behaviours of both human beings and robots coexisting in the iSpace. Based on the accumulated information, the environment as a system is able to understand the intention of human beings. For supporting humans, the environment/system utilizes machines including computers and robots.
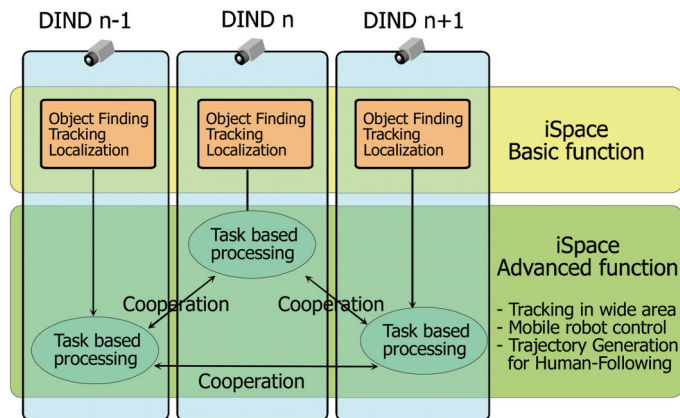
Fig. 2. DIND Configuration

The role of each DIND is separated into two parts as shown in Fig. 2. Each DIND should find human beings and robots as new target objects. Then, it tracks the target objects and gets location of them as basic functions independently of condition of other cameras,

because of keeping the robustness and the flexibility of the system. On the other hand, the information acquired by each DIND is shared among the DINDs through the network communication system. At least, target tracking in wide area monitored by multiple DINDs must be achieved as the advanced function of DIND. The protocol and data structure between the DINDs should be decided for achievement of advanced functions. It also becomes easy to install new technology as new modules to the iSpace with the unified protocol and data structure between the DINDs and modules in advance. In the iSpace, distributed devices based system enables the achievement of a flexible system based on this feature. The human-following robot introduced in this chapter is a new module for the iSpace, as it is one of the applications located in a level higher than the module of target tracking and location system based on distributed devices.

2.2 Intelligent space with vision based DIND

There are two major problems in the multiple-camera multiple-object tracking system like the iSpace as described before. The first problem is the traditional correspondence problem from frame to frame over time in the image sequence. The second is the object correspondence problem among different cameras for seamless tracking. For solving the first problem, color model based object tracking algorithms, such as MeanShift algorithm, are effective with fast and robust object tracking. This paper also describes one of the tracking methods applied for the iSpace in the following section.

On the contrary, there are several approaches to solve the latter problem in the recent literatures. These approaches include feature matching, 3D information and etc. These are mainly problems of image processing. However, in order to solve multi-camera tracking problem, the other problem establishment as sensor system architecture is needed. Many DINDs are placed redundantly and randomly as a distributed vision sensor network in the iSpace. An appropriate camera for seamless tracking should be selected from many candidates according to a tracking capability of each camera. The following section introduces a handing over protocol for sharing the information among DINDs and tracking the targets in wide area.

Target position in the space is mainly communicated between the basic function and the advanced functions as shown in Fig.2. It is also possible to install new tracking algorithms in the part of basic functions by exploiting the target position as the lowest level interface between two functions.
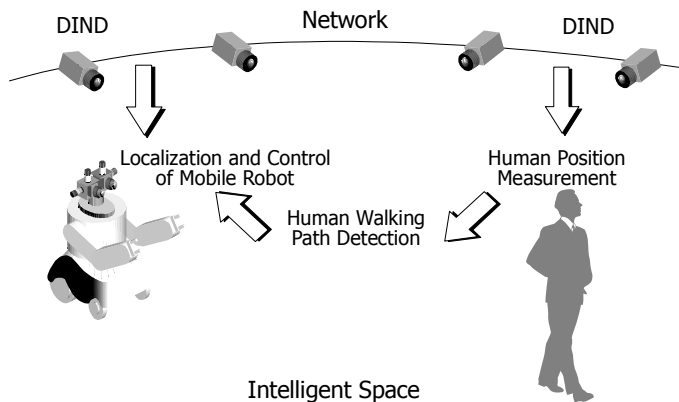
Fig. 3. Human-Following Robot in iSpace

2.3 Human centered robot in intelligent space

A mobile robot is one of the physical agents for supporting human beings in the iSpace. In order for human beings and robots to coexist and to perform cooperative works, they have to interact closely. A solution for achievement of human-centred robotic system is a human-following robot keeping a certain relative positional relationship between a human being and a robot. The human-following robot is able to provide human-centred services with following behaviour. In the case that a mobile robot accompanies a human being, the robot is able to easily acquire detailed information associated with a target person. It has been also reported that a close distance between a human being and a robot leads to mutual interactions [Sidenbladh]. Human-following robot needs several technologies, such as recognition of human beings, position estimation of human beings, and a control strategy for following human beings. In order to recognize human beings, most human-following robots developed before mounts many sensors, such as cameras, ultrasonic sensors and etc. These sensors detect the relative position from a mobile robot to a target human. The mobile robot in a literature [Sidenbladh] recognizes a human's skin colour using a camera, and traces a target human with pan-tilt control of a camera. Some of proposed human-following robots burden a target person with special equipments. It is not reasonable for a mobile robot to continue following a human being with avoiding other obstacles, without missing the target walking at a natural speed, since stand-alone robots have limitations in terms of recognition performance. Tracking system of the Intelligent Space is suitable as solutions of these problems. A mobile robot cooperates with multiple DINDs distributed in the environment. The DINDs recognize the target human and the mobile robot, and give control commands to the robot as shown in Fig.3. The mobile robot can receive necessary supports for human-following control from environmental sensors in iSpace.

3. Tracking vision in intelligent space

3.1 Color based object tracking

One of the object trackers in the DIND exploits colour histograms which represent the target feature. The object representation based on the colour histogram is relatively stable against deformation and occlusion. Recently, the tracking system based on mean shift algorithm is reported that it is suitable for the color region tracking [Comaniciu]. In this algorithm, weighted mean shift is used for multiple color region tracking. Weight is computed based on the color information of the object regions and background image as shown in Fig.4. In the research field of motion-based object tracking, one of the challenging tasks is how to deal well with the rapid movement of the object. An integrated method mean shift and Kalman filter has been proposed in the previous studies. It has proven to be efficient and relatively robust to the rapid movement of the object. In addition, this method has been compensated the weakness of mean shift tracker with Kalman filter. However, in case that the movement of a target is changed suddenly by collision with obstacles such a human, floor, and so on, the tracker loses the target object and there are few chances to recover. By adding changes to this algorithm, we were able to deal with the above problem. The dissimilarity of histograms between the model and the candidate region is expressed by a metric based on the Bhattacharyya distance. This measure has shown the benefit to achieve stable and efficient tracking in our research. In order to localize the target object, the mean shift procedure and the Kalman filter is used. In this research, we added feedback loop after

the mean shift procedure. Tracking experiments in which our method coped with the sudden change of the object movement are presented.
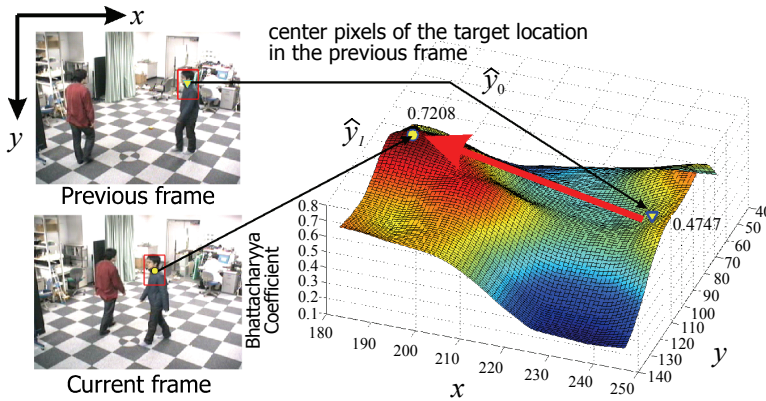
Fig. 4. MeanShift Tracker

A new approach toward target tracking [Morioka, (2007)] is shown in Fig.5. In conventional algorithm using mean shift and Kalman filter, the target location is decided through the "path1" on the Fig.5. To handle the sudden change of the movement, we added feedback loop "path2" after the mean shift process. In this algorithm, regular target localization process goes through "path1". In case that unexpected change of the movement happened and the tracker lost the object, this estimation depends on the Bhattacharyya distance, the second localization procedure starts from the position of the target in the previous frame and with the size of the region is temporarily expanded through "path2". The magnification
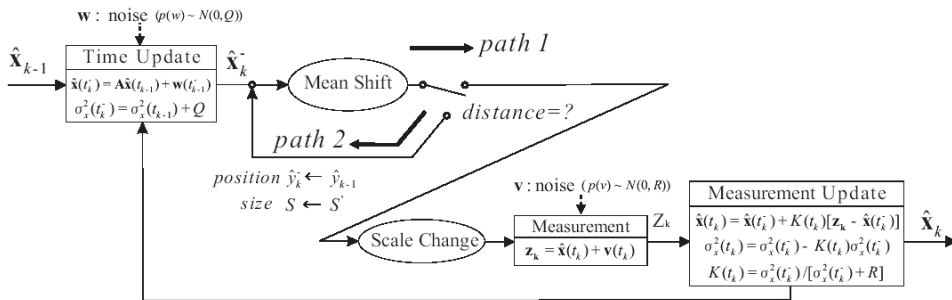
Fig. 5. Hybrid Tracking Algorithm
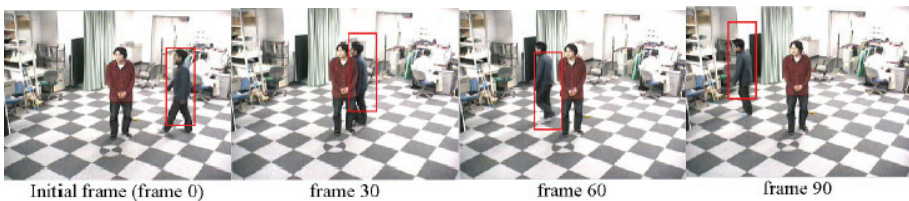
Fig. 6. Tracking Results

ratio is application dependent. We use an area less than twice the target area and adopt scalar value of the velocity vector of the target object. The means shift tracker has the weakness. That is unable to cope with rapid movement of object because the object is not included in the predicted area with Kalman filter. On the other hand, the mean shift tracker shows stability and efficiency in case that the target region overlaps with the target object. The function of "path2" is to operate the mean shift. The greatest advantage of our hybrid algorithm is that continuation of the object tracking is possible efficiently, without applying load to a system. Fig.6 shows the experimental result of human tracking with the proposed tracker. In this experiment, HSV colour space is used for configuration of colour histogram. Each axis of the HSV colour space is divided to 32 bins. Image coordinates of tracked objects are converted to world coordination based on camera parameters by using a set of two DINDs. World coordinates can be obtained from one DIND in the case that the height of target object is known.

3.2 Handing over for seamless tracking of mobile robot

A DIND is an independent device, and its functions, including tracking and localization of target objects, are performed completely within it. Especially, tracking and localization of a mobile robot is the most important task for achievement of human-centred robotic system. In addition to localization, a DIND sends control signals based on localization results to a target mobile robot. Thus, if a mobile robot is moving in the area a DIND is monitoring, the robot is guided without any difficulty. However, to guide robots in a wider area, which a single DIND cannot cover, we define the DIND that has the control authority of a robot as a dominant DIND for the robot. When a robot moves from an area to a different area, the robot's dominant DIND should be automatically changed. This is called "handing over of control authority" [Lee, (2004)]. A dominant DIND has the control authority of the robots and at one time only a single dominant DIND exists for each robot. Therefore, the control authority should be smoothly handed over to the next DIND at the proper time and location. To solve this problem, a reliability rank is devised. High reliability stands for that a DIND can guide a robot robustly and precisely. Generally, the area near a DIND and the centre area of an image captured by a DIND, have the highest reliability rank; while the boundary of the image and the area far from the DIND have the lowest reliability since a camera is adopted as a sensor for a DIND. Fig.7 shows that the monitored area is divided into several areas and the actual reliability ranks are determined.

Fig.8 shows a process for handing over of the control authority of a target robot. Circles indicate the monitoring areas of cameras. The monitoring areas of dominant DINDs are coloured. First, the dominant DIND requests information from the other DINDs about the reliability rank of the target robot. The other DINDs reply with their reliability rank concerning the robot and the current dominant DIND. The dominant DIND compares these values with its own rank. If the other DIND has a higher rank than its own rank, then the current dominant DIND transfers the authority of control to the other DIND, which has the higher rank. Then the new dominant DIND tracks and controls the robot.

Fig.9 shows the results of the handing over among three DINDs. (x,y) coordinates of a tracked mobile robot in world coordinate system are plotted in this figure. In this experiment, a simple colour tracking is exploited for tracking in a single DIND [Appenzeller]. At the beginning, the mobile robot was located in the monitoring area of

DIND 3. Three kinds of signs of Fig. 9 show the ID of DIND which the mobile robot was tracked by. According to the position of the robot, the dominant DIND changed in the order of DIND 3, 1, 2. It turned out that the mobile robot continued to be tracked, even when it's dominant DIND changed. The estimated positions of the mobile robot were discontinuous at the point where the dominant DIND changed. This was caused by position measurement error resulting from camera calibration and image processing. However, the error hardly influenced the control of the mobile robot because of the proposed control method as follows.
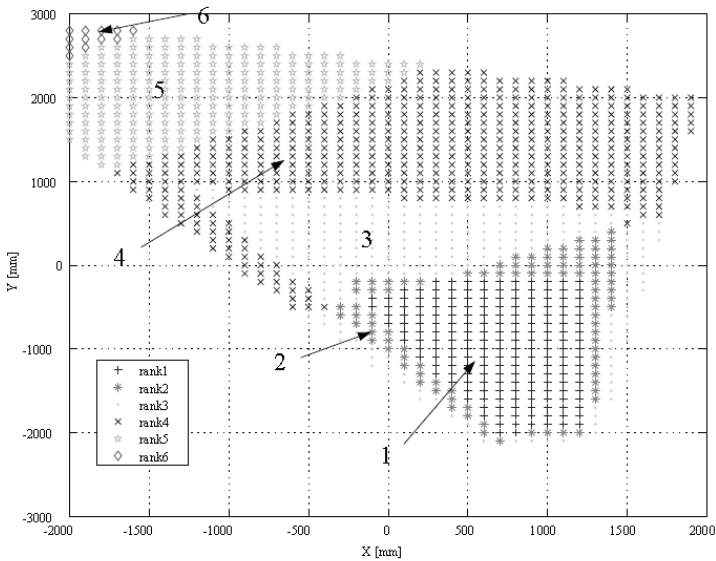
Fig. 7. Reliability Map of Monitoring Area
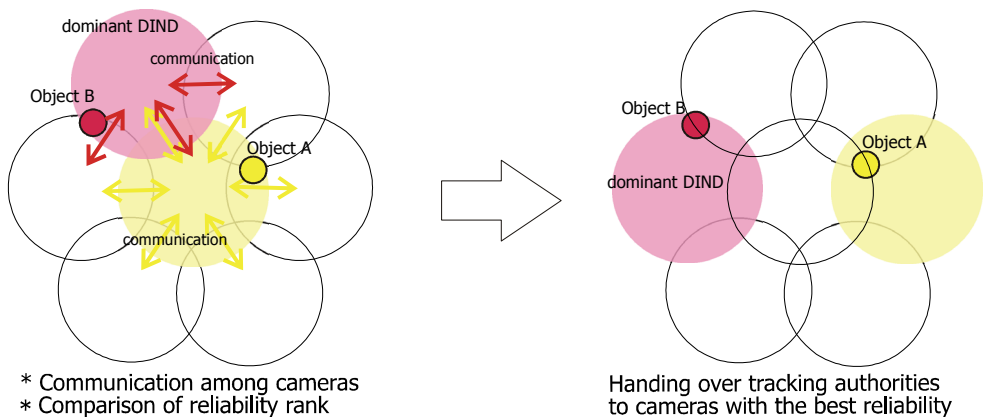
Fig. 8. Handing Over
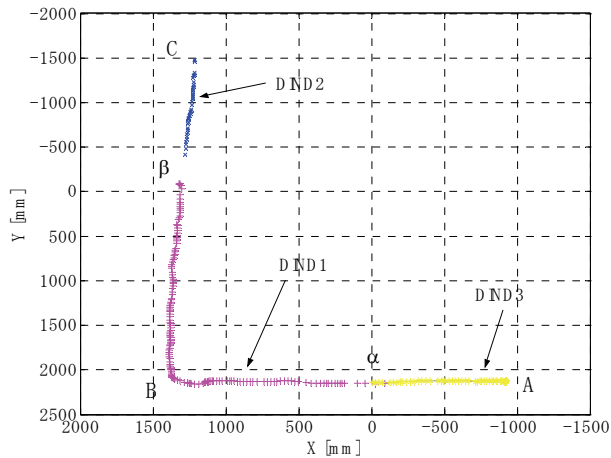
Fig. 9. Tracking Result with Handing Over Protocol

4. Application – human-following mobile robot

4.1 Human-following mobile robot in iSpace

In this section, a human-following mobile robot is introduced as an application of vision based iSpace. Several effective features are expected from the human-following robot as a physical agent in iSpace. The first feature is that human following can be realized without depending on performances of mobile robots. Intelligence of not a robot itself but the iSpace is mainly used for navigation of a mobile robot in iSpace [Lee, (2002)]. For this reason, it is possible to perform human-following operations in iSpace with any type of mobile robot, even though robots lack sensors and intelligence. The second feature is that human following is possible even in a complicated environment. The iSpace usually acquires information about target objects such as human beings, robots and obstacles with cooperation of multiple DINDs. Since mobile robots are managed by DINDs from the environment side, the iSpace is able to generate each robot's path such that they do not collide with each other. Therefore, the robots are able to move smoothly, even if they are in a complicated environment. The third feature is that the iSpace can include various functions; behaviour-based control [Szemes] and spatial memories [Niitsuma] and etc. When these functions and the human-following robot system are combined, more meaningful actions of mobile robots will be achieved. When a robot with such advanced functions is in an ordinary space, the robot is able to afford similar functions as the robots in the iSpace. However, its serviceable area is limited to a range near the robot. It is difficult to provide suitable services for people in far from the robot. In such a case, a human being is required to approach the mobile robot in order to inform his/her requirements and to receive services from the robot. A human-following mobile robot in the iSpace is one of the best solutions as a human-centred robotic system for these reasons.

4.2 Control and system configuration

A differential wheel velocity-type mobile robot is used for the human-following robot because this type of robot has a simple and compact structure. It is suitable for a physical

agent that has to interact with human beings in complicated environments. Our mobile robot is based on the Pioneer2-DX. This mobile robot is connected to the DIND network via wireless LAN. Since the height of the mobile robot is already known, the position of a mobile robot is reconstructed from one camera image.. Positions and headings of a mobile robot are measured with tracking of three colour markers mounted on the top of a robot.

A tracking control is performed to a mobile robot in order to follow a human being. Many studies have been performed in the field of the tracking control. In the control of a nonholonomic mobile robot, Brockett's theorem proved that a smooth state feedback law for an asymptotically stable to one point of the state space does not exist [Nakamura]. In recent years, various closed-loop control systems which can overcome the feedback stabilization impossibility of Brockett's theorem have been proposed. In the iSpace, since a human walking trajectory is newly generated in every step, it can be considered as a function of time. Therefore, the application of tracking control is effective. However, the target trajectory for control of a mobile robot is limited to continuous and smooth ones in most cases. A human-following robot should be able to track actual human walking trajectories, including abrupt changes in velocity and direction. A human-following robot with a conventional tracking control may lose stable tracking of a target person. A tracking control method in the iSpace for following human beings should be newly proposed.

The estimated human position data is not proper control input required for a mobile robot to follow a human being since estimated position data contains errors in the form of calibration error and image processing error. When the heads of humans are measured by vision sensors in a short sampling period, their velocity and direction also change drastically. In addition to unstableness of position data measured by the iSpace, the direction and velocity of human walking sometimes changes suddenly and unpredictably. In this case, the mobile robot may fail to follow the target human and lose stable movement by a sudden change in a velocity input. It is considered that a mobile robot cannot follow the target human with conventional control law derived only from the distance between the target human and the mobile robot. There are fundamental differences between a mobile robot and a human being in the level of motion. A human is able to move freely by foot. However, a differential wheel velocity type mobile robot with nonholonomic constraints cannot move as freely as does a human. To trace a human who walks freely, a control strategy that overcomes the limitations of nonholonomic constraints is needed. This chapter introduces a control law with a virtual spring model for absorbing the kinematic difference between the human and the mobile robot [Morioka, (2004)]. The proposed control law is derived from an assumption that a human being and a mobile robot are connected by a virtual spring as shown in Fig.10. An input velocity to a mobile robot is generated on the basis of an elastic force of a virtual spring in this model. In the proposed control system, the virtual spring works as a low pass filter and absorbs adverse fluctuations. The proposed virtual spring model is able to absorb the gap between a motion of human being and that of the mobile robot.

DIND software is actually implemented with configuration as shown in Fig. 11. A program for DIND basic functions is always running in each DIND. This basic program includes image capture, image processing for target tracking, and reconstruction of 3D coordinate. In the case of 3D reconstruction of a target with unknown height, two cameras are combined as a sensor part of DIND. A DIND basically makes connections to other DINDs with the client/server method of UDP protocol. A program for DIND basic functions in each DIND is always running for obtaining target measurements. Server programs for advanced

functions are also running in order to receive requests for passing the control authority from other DINDs or to provide measurement positions of target human beings. Client programs are executed according to the situation needed for advanced functions. For example, client for advanced task #0 will start in the case that DIND requests handing over to other DINDs. The client for advanced function #1 for making a connection to the mobile robot become active if DIND has a control authority for a human-following robot. It provides the velocity inputs for a human-following mobile robot. The client for advanced function #2 is also executed for obtaining target human positions in human-following task.
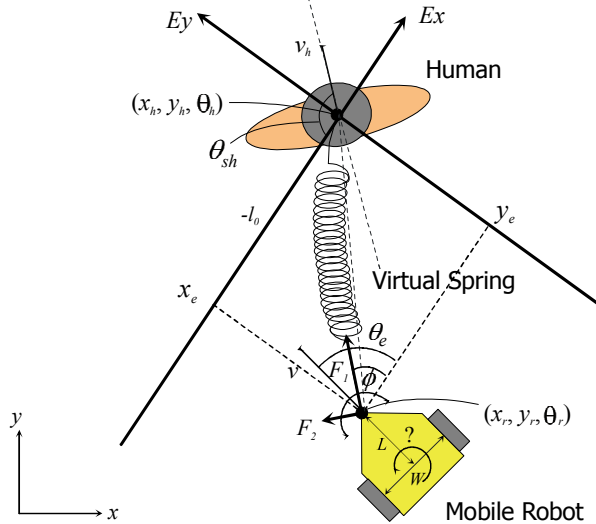
Fig. 10. Virtual Spring Model
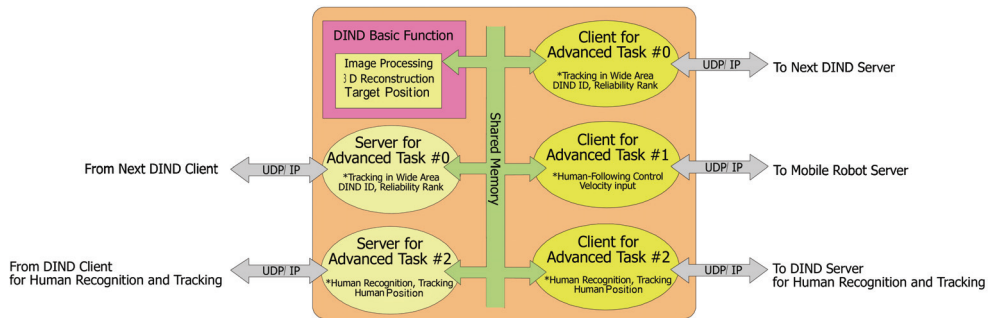
Fig. 11. Hybrid Tracking Algorithm

4.3 Experiment

The results of the experiment are shown in Fig.12. In this experiment, four DINDs for robot control and one DIND with two cameras for human tracking are placed in the iSpace. A server program for human recognition and tracking are active in the DIND for human tracking. The programs for advanced tasks except to a server for human recognition and

tracking become active in the DINDs for robot control according to the situations. In this experiment, a simple colour tracking [Appenzeller] is applied for basic functions. DINDs for robot control have a colour model of a mobile robot, and a DIND for human tracking just has a skin colour model of human beings in advance. A human walked along a hexagonal path. The mobile robot followed him along the course plotted by four kinds of signs in Fig.13. Four signs in Fig.13 show that the dominant DIND for the mobile robot changed according to the handing over protocol. In spite of the unstable human position estimation, the mobile robot was controlled smoothly. The low-frequency fluctuations in the human trajectory were cut off by the virtual spring and the robot was controlled smoothly. It is found from the results that the proposed algorithm allows the robot to follow the target without excessive motions. In Fig.12, the dashed line represents the position measured by the encoders in the robot. The dashed line deviates significantly from the trajectory of the robot measured by the DINDs. This is mainly caused by slips of the wheels and the wrong internal parameters of the robot. It is also an advantage for the human-following robot realized in the iSpace to be robust against various errors that disturb the robot's navigation. Although this human path is simple from a global point of view, complex motions, such as variations in the velocity and direction of humans are included in the human's actual movement. The validity of the human-centred robotic system based on the iSpace was verified from this experimental result.
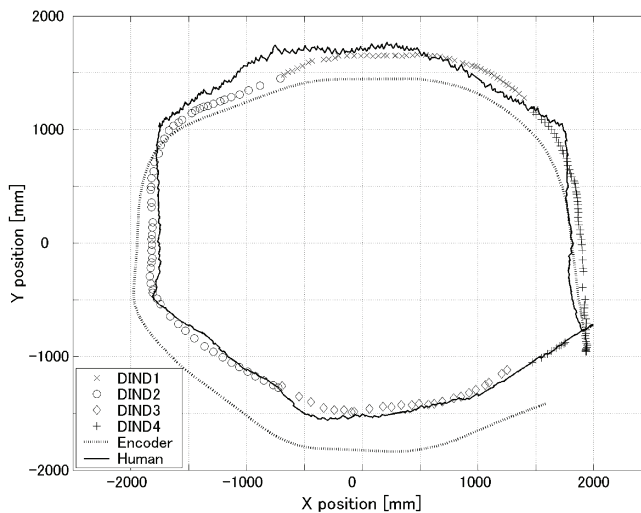
Fig. 12. Experimental Results of Human-Following

5. Conclusion

In this book chapter, the Intelligent Space for achievement of human-centred robotic system was presented. The positions of target objects in the iSpace should be measured with multiple DINDs installed in a wide area. This chapter introduced a hybrid tracking algorithm including MeanShift and Kalman filter for object tracking in one DIND. To track target objects in a wide area and control mobile robots based on environmental measurement, cooperation of the DINDs, effective communication and role assignment are

required. The handing-over protocol for a mobile robot control was explained. Finally, a human-following mobile robot was introduced as an application of the iSpace for a human-centred robotic system. A human-following robot is based on the measurement infrastructure of the iSpace with multiple DINDs. A proposed control with a virtual spring model was implemented as a new module in each DIND. The result shows that human-following is easily achieved in the iSpace. Future studies will involve applying this system to complex environments where many people, mobile robots and obstacles coexist. It is necessary to investigate the influence of a human-mobile robot which maintains a fixed distance between a robot and a target human with introducing knowledge of cognitive science and social science.

6. References

Appenzeller, G.; Lee, J.-H.; & Hashimoto, H. (1997). Building Topological Maps by Looking at People: An Example of Cooperation between Intelligent Space and Robots, *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol.3, pp.1326-1333, ISBN 0-7803-4119-8, Grenoble, France

Comaniciu, D.; Ramesh V. & Meer, P. (2003). Kernel-Based Object Tracking, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol.25, No.5, pp.564-577, ISSN 0162-8828

Lee, J.-H & Hashimoto, H. (2002). Intelligent Space - concept and contents. *Advanced Robotics*, Vol.16, No.3, pp.265-280, ISSN 0169-1864

Lee, J.-H.; Morioka K.; Ando N. & Hashimoto H. (2004). Cooperation of Distributed Intelligent Sensors in Intelligent Environment. *IEEE/ASME Trans. on Mechatronics*, Vol.9, No.3, 2004, pp.535-543, ISSN 1083-4435

MacDorman, K. F.; Nobuta, H.; Ikeda, T.; Koizumi, S. & Ishiguro, H. (2004). A memory-based distributed vision system that employs a form of attention to recognize group activity at a subway station, *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.571-576, ISBN 0-7803-8464-4, Sendai, Japan

Morioka, K.; Lee J.-H. & Hashimoto H. (2002). Human- Following Mobile Robot in a Distributed Intelligent Sensor Network. *IEEE Trans. on Industrial Electronics*, Vol.51, No.1, 2004, pp.229-237, ISSN 0278-0046

Morioka K.; Lee, J.-H.; Kuroda, Y. & Hashimoto, H. (2007). Hybrid Tracking Based on Color Histogram for Intelligent Space. *Journal of Artificial Life and Robotics*, Vol.11, No.2, pp.204-210, ISSN 1433-5298

Nakamura, Y. (1993). Nonholonomic robot system, Part 2: Motion planning under kinematic nonholonomic constraints. *Journal of Robotics Society of Japan*, vol. 11, no. 5, pp. 655–662, ISSN 0289-1824

Niitsuma. M.; Hashimoto, H. & Hashimoto, H. (2007). Spatial Memory as an Aid System for Human Activity in Intelligent Space. *IEEE Trans. on Industrial Electronics*, Vol.54, No.2, pp.1122-1131, ISSN 0278-0046

Sato, T.; Harada, T. & Mori, T. (2004). Environment-Type Robot System "Robotic Room" Featured by Behavior Media, Behaviour Contents, and Behavior Adaptation. *IEEE/ASME Trans. on Mechatronics*, Vol.9, No.3, pp.529-534, ISSN 1083-4435

Sidenbladh, H. & Kragic, D. & Christensen H. I. (1999). A person following behavior for a mobile robot, *Proc. 1999 IEEE Int. Conf. Robotics and Automation*, Vol.1, pp. 670–675, ISBN 0-7803-5180-0, Detroit, USA

Szemes, P.; Hashimoto, H. & Korondi, P. (2005). Pedestrian-behavior-based mobile agent control in intelligent space. *IEEE Trans. on Instrumentation and Measurement*, Vol.54, No.6, pp.2250-2257, ISSN 0018-9456

Development of a Sensor System for an Outdoor Service Robot

Takeshi Nishida[1], Masayuki Obata[2],
Hidekazu Miyagawa[2] and Fujio Ohkawa[1]
[1]Kyushu Institute of Technology,
[2]YASKAWA INFORMATION SYSTEMS Corporation
Japan

1. Introduction

In general, service robots are equipped with multiple types of sensor for environmental recognition and to avoid the measurement error that occurs by various measurement noises. Especially for the robots that work in outdoor environments, cameras and LRFs (Laser Rangefinders) are the most useful sensor devices, and they have been installed into many prototype service robots. Robots can acquire texture, color, shadow, etc. of objects or a scene via cameras, and execute various tasks, e.g. landmark recognition, face recognition, target tracking etc. based on those information. Moreover, the stereovision composed by using two or more cameras can acquire 3D information on the scene. However, the distance measurement of the objects with difficulty of decision of correspondence of feature points, such as walls without texture and shadow, is difficult. Furthermore, when the strength of environmental light change that exceeds the dynamic range of the camera image sensor occurs, the measurement accuracy greatly falls (DeSouza et al., 2002). On the other hand, LRF is a device which uses a laser beam in order to determine the distance to a reflective object, and then the distance with comparatively high accuracy can be measured even in the situation in which the measurement with the camera becomes unstable. Therefore, the LRF is frequently used for localization, map building, and running route inspection of autonomous mobile robots. However, the calculation algorithm to recognize the target object by using the LRF is very complex, the calculation cost is also high, and have following disadvantages:

1. The range data sometimes involve lack of data called black spots around the corner or the edge of the objects.
2. The range data involves quantization errors owing to measurement resolution (e.g. 10 [mm]).
3. The number of data points in a range data set is large.
4. Texture and color information etc. on the object cannot be acquired.

Therefore, we have developed a novel sensor system that consists of two cameras and a LRF; various types of measurement and recognition of targets are possible according to those combinations. Moreover, the sensor system has both advantages of camera and LRF, and has the robustness against environment variations. In this chapter, we show the

methods of target recognition and 3D posture measurement using the sensor system after giving explanation about the construction of hardware. Furthermore we show the results of several outdoor experiments of the robot equipped with the sensor system.

2. Structure of sensor system

2.1 Devices

The sensor system (Fig. 1) consists of two cameras, a LRF, and three stepping motors. The maximum measurement distance of the LRF (URG-04LX) is 4 [m], and the measurement error margin is less than 1 [%]. The infrared laser is radially irradiated from the center part of the LRF, its range of the measurement is forward 170 [deg] (the maximum range is 240 [deg]), and the angle resolution is about 0.36 [deg]. The CCD color cameras capture images in 24-bit color at 640×480 resolution at rates to 30 [fps]. Moreover, these installation positions have been designed so that those optical axes and the measurement plane of LRF are parallel.
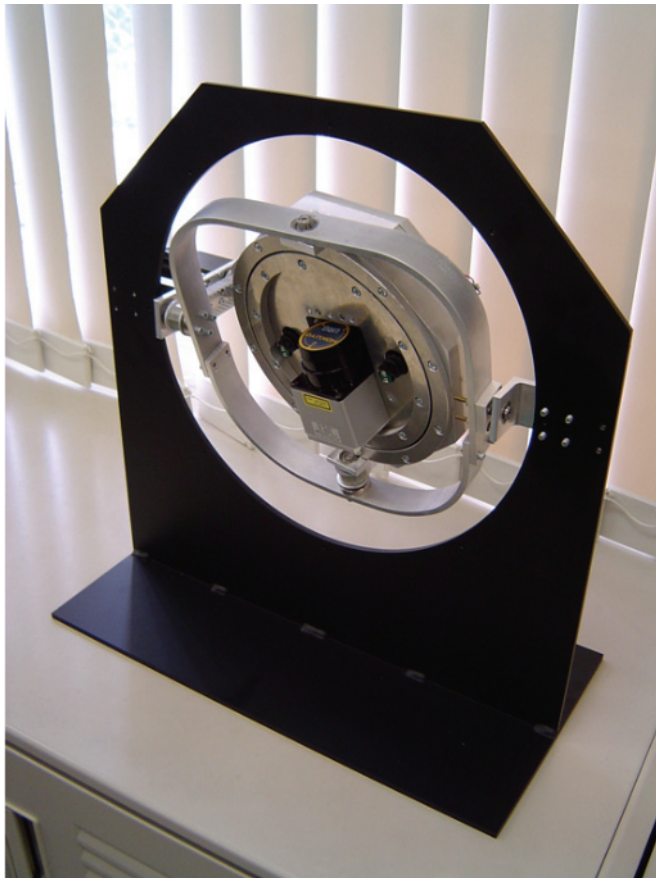
(a) Sensor system
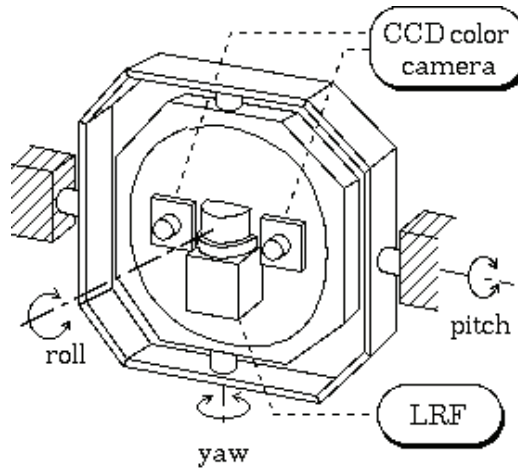
(b) Configuration of the sensor system

Fig. 1. Sensor system developed for an outdoor robot.

2.2 Coordinate systems

The definition of each coordinate system and the relations between them are shown here. At first, Table 1 shows the definition of each coordinate system, and the relations between the cameras and the LRF are shown in Fig. 2.

World coordinate system	$\Sigma_w = \left\{ {}^w x, {}^w y, {}^w z \right\}$
Robot coordinate system	$\Sigma_r = \left\{ {}^r x, {}^r y, {}^r z \right\}$
LRF coordinate system	$\Sigma_l = \left\{ {}^l x, {}^l y, {}^l z \right\}$
Left camera coordinate system	$\Sigma_{cl} = \left\{ {}^{cl} x, {}^{cl} y, {}^{cl} z \right\}$
Right camera coordinate system	$\Sigma_{cr} = \left\{ {}^{cr} x, {}^{cr} y, {}^{cr} z \right\}$
Left camera screen coordinate system	$\Sigma_{sl} = \left\{ {}^{sl} x, {}^{sl} y \right\}$
Right camera screen coordinate system	$\Sigma_{sr} = \left\{ {}^{sr} x, {}^{sr} y \right\}$
Right hand coordinate system	$\Sigma_{hr} = \left\{ {}^{hr} x, {}^{hr} y, {}^{hr} z \right\}$
Right hand-eye screen coordinate system	$\Sigma_{hsr} = \left\{ {}^{hsr} x, {}^{hsr} y \right\}$

Table 1. Coordinate systems

Let $R_X(\cdot)$, $R_Y(\cdot)$, and $R_Z(\cdot)$ be the rotation matrices for each axis of the robot coordinate system. The homogeneous coordinate transformation matrix from the sensor coordinate system Σ_r to the LRF coordinate system Σ_l is represented as follows,
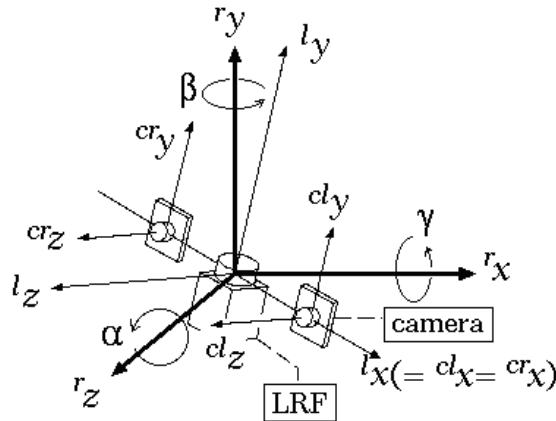
Fig. 2. Relations of coordinate systems

$$\,^r_l H \triangleq \begin{pmatrix} \,^r_l R & \,^r_l p \\ \mathbf{0} & 1 \end{pmatrix} \in \mathbb{R}^{4\times 4} \,, \tag{1}$$

where,

$$\,^r_l R \triangleq R_Z(\alpha)\cdot R_Y(\beta)\cdot R_X(\gamma), \tag{2}$$

$$R_X(\gamma) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{pmatrix}, \quad R_Y(\beta) = \begin{pmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{pmatrix}, \quad R_Z(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix}, \tag{3}$$

and $\,^r_l p$ represents a translation vector from the origin of Σ_r to the origin Σ_l. From these relations, the mapping from a measurement point $\,^l x_{ij} \triangleq \left(\,^l x_{ij}, \; ^l y_{ij}, \; ^l z_{ij}\right)^T \in \Sigma_l$ to the point $\,^r x_{ij} \triangleq \left(\,^r x_{ij}, \; ^r y_{ij}, \; ^r z_{ij}\right)^T \in \Sigma_r$ is described by

$$\begin{pmatrix} \,^r x_{ij} \\ 1 \end{pmatrix} = \,^r_r H \begin{pmatrix} \,^l x_{ij} \\ 1 \end{pmatrix} = \,^r_l H^{-1} \begin{pmatrix} \,^l x_{ij} \\ 1 \end{pmatrix}, \tag{4}$$

where, $i = 1, \cdots I, j = 1 \cdots J$ and

$$\,^r_l H^{-1} = \begin{pmatrix} \,^r_l R^T & -\,^r_l R^T \,^r_l p \\ \mathbf{0} & 1 \end{pmatrix}. \tag{5}$$

Next, the homogeneous coordinate transformation matrices from the LRF coordinate system Σ_l to the left and the right camera coordinate system Σ_{cl} and Σ_{cr} are represented as follows,

$$_{cl}^{l}\boldsymbol{H} = \begin{pmatrix} _{cl}^{l}\boldsymbol{R} & _{cl}^{l}\boldsymbol{p} \\ \mathbf{0} & 1 \end{pmatrix}, \quad _{cr}^{l}\boldsymbol{H} = \begin{pmatrix} _{cr}^{l}\boldsymbol{R} & _{cr}^{l}\boldsymbol{p} \\ \mathbf{0} & 1 \end{pmatrix}, \tag{6}$$

where, $_{cl}^{l}\boldsymbol{p}$ (or $_{cr}^{l}\boldsymbol{p}$) is translation vector from the origin of Σ_l to the origin of Σ_{cl} (or Σ_{cr}). Therefore, the posture of the left (right) camera against Σ_r is described by

$$_{cl}^{r}\boldsymbol{H} = _{l}^{r}\boldsymbol{H}\,_{cl}^{l}\boldsymbol{H}, \quad (_{cr}^{r}\boldsymbol{H} = _{l}^{r}\boldsymbol{H}\,_{cr}^{l}\boldsymbol{H}). \tag{7}$$

Here, the sensor system is designed to be $_{cl}^{l}\boldsymbol{R} \approx \boldsymbol{I}_{3\times3}$, $_{cr}^{l}\boldsymbol{R} \approx \boldsymbol{I}_{3\times3}$ (i.e. $_{r}^{l}\boldsymbol{R} = _{r}^{cl}\boldsymbol{R} = _{r}^{cr}\boldsymbol{R}$), and $_{cl}^{l}\boldsymbol{p} = -_{cr}^{l}\boldsymbol{p}$ for easiness. Furthermore, the mapped point $^{sl}\boldsymbol{x}_{ij}$ of $^{l}\boldsymbol{x}_{ij} \in \Sigma_l$ on Σ_{sl} is represented as follows,

$$_{cl}^{r}\boldsymbol{H}\begin{pmatrix} ^{r}\boldsymbol{x}_{ij} \\ 1 \end{pmatrix} = \begin{pmatrix} _{l}^{r}\boldsymbol{R}\cdot\left(^{r}\boldsymbol{x}_{ij} + _{cl}^{l}\boldsymbol{p}\right) + _{l}^{r}\boldsymbol{p} \\ 1 \end{pmatrix} \triangleq \begin{pmatrix} ^{cl}\boldsymbol{x}_{ij} \\ 1 \end{pmatrix}, \tag{8}$$

$$_{sl}^{cl}\boldsymbol{f}\left(^{cl}\boldsymbol{x}_{ij}\right) \triangleq \left(-f_{cl}\frac{^{cl}x_{ij}}{^{cl}z_{ij}}, \quad f_{cl}\frac{^{cl}y_{ij}}{^{cl}z_{ij}}\right)^{T} \triangleq {}^{sl}\boldsymbol{x}_{ij}, \tag{9}$$

where, f_{cl} is the focal length of the left camera. Moreover, to convert the unit of $^{sl}\boldsymbol{x}_{ij}$ from the distance into the pixel, the following transformation is defined.

$$F\left(^{sl}\boldsymbol{x}_{ij}\right) \triangleq \begin{pmatrix} s_x \cdot {}^{sl}x_{ij} \\ s_y \cdot {}^{sl}y_{ij} \end{pmatrix} \triangleq {}^{sl}\boldsymbol{x}_{mn}, \tag{10}$$

where, $M = \{m | m \in \mathbb{Z}_+, m \leq |M|\}$ and $N = \{n | n \in \mathbb{Z}_+, n \leq |N|\}$ are width and height pixel indexes of the $|M| \times |N|$ image, and (s_x, s_y) are scale factors to transform them from the measurement distance to the pixel number. Hence, the measurement data points of the LRF are mapped into the left camera screen pixels by Eqn. (8), (9), (10), and the measurement data between devices is mutually mapped by the above relational expressions. In addition, notes of this system are enumerated as follows. Firstly, pinhole camera model is assumed and lens distortion of the camera is disregarded. Secondly, the values of $_{l}^{r}\boldsymbol{p}, _{cl}^{l}\boldsymbol{p}, _{cr}^{l}\boldsymbol{p}$ are already-known according to a prior calibration. Thirdly, if the system configuration with high accuracy is difficult, the calibration of $_{r}^{cl}\boldsymbol{R}, _{r}^{cr}\boldsymbol{R}$ are also necessary. Lastly, the rotation angles that provides for $_{r}^{l}\boldsymbol{R}$ can be measured by counting the pulse input of the stepping motors.

3. Object detection and target tracking

The applications for the object detection and the target tracking were developed on the assumption that this sensor system will be installed in the autonomous mobile robot. Therefore, to execute those tasks, high speed execution and robustness are demanded for the processing of the data obtained from the sensor system. Although the objects detection and the target tracking by the camera image processing are executable at high speed, they are influenced easily from the change in environmental light. On the other hand the calculation cost for the measurement and the object detection by the LRF is comparatively high; however, the measurement is robust under the influence of environmental light. Therefore, these devices of the sensor system are combined to complement each other for high accuracy detection and measurement. Namely, the target is measured with only one device when a limited measurement for fast computation is required, and is measured with multiple devices when highly accurate processing is required.

In this section, we will show applications of the sensor system, and one of the flowchart is shown in Fig. 3. In this application, at first, a target object is detected and tracked by camera image processing, and after that, the range data of the object obtained by rotating of the LRF analyzed. By such measurement flow, the object detection is quickly achieved by the camera image processing, and the analysis in detail of the target object is done by using the LRF data not influenced easily from an environmental change. In the following, we consider the procedure for detection and measurement of plastic bottles for a concrete example.
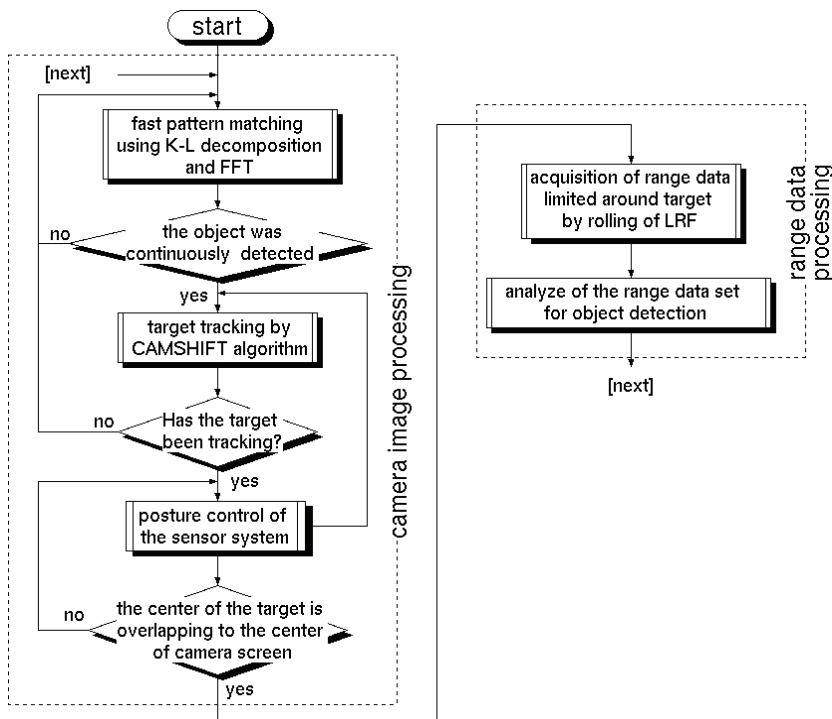
Fig. 3. Flowchart of an application of the sensor system.

3.1 Camera image processing

Since the appearance of trash changes according to position and posture, robustness to distortion and transformation of shape is necessary for the trash detection method. Therefore, from the point of view for robustness and fast execution, we employed the eigen space method (Uenohara & Kanade, 1997) for the object detection, and employed the CAMSHIFT method (Gray, 1998) for the target tracking (see (Fuchikawa et al., 2005) for detail). These methods are briefly explained as follows.

3.1.1 Karhunen-Loeve expansion

Let $q_l(x) \triangleq q_l(x, y)$, $(l = 1, \cdots, L)$ be a set of template images consisting of $r \times s$ pixels. Moreover let q_l be its $rs (= r \times s)$ dimensional vector in scan line order. The singular value decomposition (SVD) of a matrix $Q = (q_1 - \bar{q}, \cdots, q_l - \bar{q}, \cdots, q_L - \bar{q}) \in \mathbb{R}^{rs \times L}$ is given by

$$Q = UDV^T, \tag{11}$$

where $\bar{q}$ is average vector of q_l, $U = (u_1, \cdots, u_L) \in \mathbb{R}^{rs \times L}$ and $V = (v_1, \cdots, v_L) \in \mathbb{R}^{L \times L}$ are orthogonal matrices, $D = \mathrm{diag}(d_1, d_2, \cdots, d_L) \in \mathbb{R}^{L \times L}$ is a diagonal matrices, and $d_1 \geq d_2 \geq \cdots \geq d_L$.

There are some methods for detecting objects by using eigenvectors (eigen images). We assume that a robot works under outdoor environment, so robustness for changes of lighting is required. Since normalized cross-correlation is insensitive to the variation of intensity of the background, we employ a method obtaining normalized correlation approximately by using eigenvectors. Suppose we have major K eigenvectors u_i and the following vector

$$u_0 = \frac{\bar{q} - U'a^c}{\left\| \bar{q} - U'a^c \right\|}, \tag{12}$$

where the coefficient vector a^c and U' are given by

$$a^c = U'^T \bar{q}, \tag{13}$$

$$U' = (u_1, \cdots, u_K) \in \mathbb{R}^{rs \times K}. \tag{14}$$

The number of eigenvector K affects the processing speed and reliability of result. Here, the order K is decided from the cumulative proportion

$$\mu^{(K)} = \frac{\sum_{k=1}^{K} d_k}{\sum_{l=1}^{L-1} d_l}, \tag{15}$$

and experimental recognition rate. Finally, the matrix composed of u_0 and U' as follows

$$\bar{U} \triangleq (u_0, u_1, \cdots, u_K) \tag{16}$$

is used in the recognition processes. We use images of 12 kinds of plastic bottles as template images, and assume that the range of object detection is about 500 [mm] to 2000 [mm] from a robot, the images were captured as shown in Fig. 4. Moreover, we set the height of a camera to 400 [mm], the depression angle $\theta_p = 10; 20; 30; 40$ [deg], the rotation angle of a bottle $\theta_r = 0; 10; \cdots; 350$ [deg] (every 10 [deg]). Namely, we collected 432 template images with 150×110 pixels (Fig. 5 (a)). The centers of appearance of the plastic bottles and the centers of the images were adjusted to be overlapped, and the intensities of the background of the images adjusted to be zero. Moreover, for determining K, we conducted the object detection experiments by using several outdoor images that contains plastic bottles, examples of the experimental results are shown in Fig. 6. We decided $K = 20$ from these experiments, and the cumulative proportion was $\mu^{(20)} = 0.418$. Since it is advisable that the details such as difference of labels were disregarded for the generality of the object detection processing, the cumulative proportion was not set so high. The reconstructed images by using the dimension K are shown in Fig. 5(b), and we can see from these figures that the eigenvectors and coefficients kept features of plastic bottles.
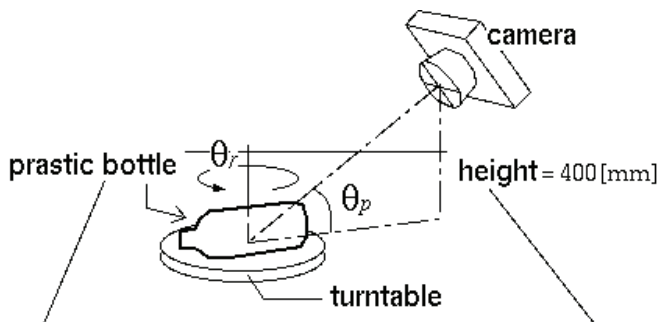
Fig. 4. Shooting conditions of template images.

(a)　　　　　　　　　　　　　　(b)
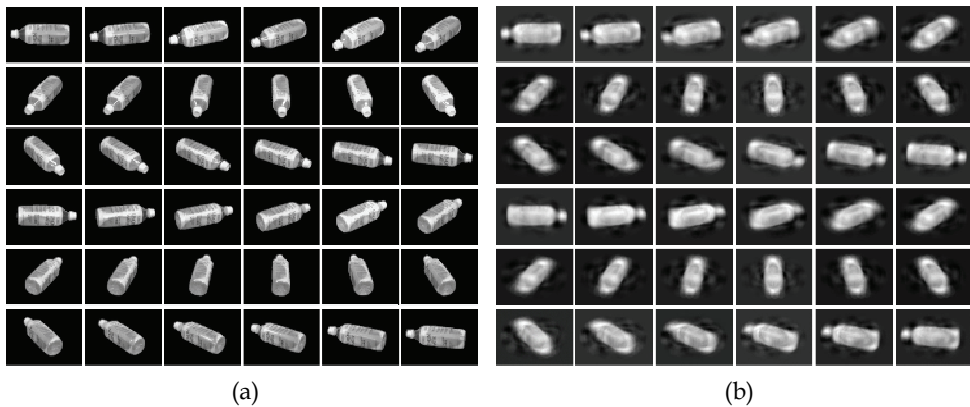
Fig. 5. Example of plastic bottle images: (a) 36 of 432 are shown; (b) reconstructed images.

Fig. 6. Examples of results of object detection experiments under outdoor environments.

3.1.2 Template matching in vector subspace

Let $p(x) \in \mathbb{R}^{r \times s}$ be an input image extracted around the pixel x from a camera image, and $p \in \mathbb{R}^{rs}$ be its vector. First, p is normalized and transformed to be zero mean vectors $\bar{p}$, i.e. the average of vector elements is zero. The pattern $\tilde{q}$ constructed by using the vector subspace is given by

$$\tilde{q} = \sum_{i=0}^{K} (u_i^T \bar{p}) u_i.$$
(17)

Furthermore, the normalized cross-correlation of $\bar{p}$ and $\tilde{q}$ is calculated by

$$R = \frac{\bar{p}^T \tilde{q}}{\|\bar{p}\|} = \frac{\sqrt{\sum_{i=0}^{K} (u_i^T \bar{p})^2}}{\|\bar{p}\|}.$$
(18)

Hence, we can obtain the normalized correlation R approximately by Eq. (18), and the computational cost is reduced greatly than calculating directly with original template images. Moreover, Eq. (18) can be calculated by using FFT (Fast Fourier Transform) efficiently. Furthermore, we introduce a value $R_{\text{th}} = 0.7$ as a threshold value of R to adjust the false detection rate. Therefore, the positions $x_d\ (d = 1, \cdots)$ where the normalized correlation values are larger than R_{th} are searched, and they are decided as targets.

3.1.3 Target tracking

The detected targets are tracked by means of CAMSHIFT algorithm (Bradski, 1998). It is a nonparametric algorithm that investigates gradients of the hue values, obtained by HSV transformation, around the targets and tracks the peak points of these gradients. Moreover, this algorithm has robustness for the occlusion and the appearance changing of the object,

the computational cost is comparatively low because the target tracking regions are limited around ROI in last frame, and it is executable by the video rate.

First, the point x_d is set as an initial point of center of the target region that has been continuously and stably detected by the eigen space method mentioned above. When the plural objects are detected, the object nearest the robot, i.e., the object with the shortest distance from the center of the lower side of the image is set as a target. Next, the target is tracked by executing the CAMSHIFT algorithm. When losing sight of the target by pedestrians or moving of the robot equipped with the sensor system, the object is detected by using the eigen space method again. The experimental results of these procedures are shown in Fig. 7. The frame rates of these processing for 512×512 input image were about 5 [fps] for the eigen space method and about 20 [fps] for the CAMSHIFT algorithm by a computer equipped with a Pentium4 3GHz.

3.1.4 Measurement of the target by stereo vision

This sensor system has a 3D measurement function based on a literature (Birchfield et al., 1999) on stereovision. Moreover, various parameters of the cameras in the sensor system were designed to meet the following requirements. Namely, the measurement is in error by less than 10 [mm] when the distance of the sensor system is within 2 [m] from the object. On the other hand, the measurement range of the LRF is from 60 [mm] to 4095 [mm], and its measurement is in error by 10 [mm] to 40 [mm]. Thus, the measurement by the stereovision is effective when the distance from the object is within 2 [m], and the LRF is effective for a longer distance. However, the measurement by the stereovision is influenced by environment light strongly, and the accurate 3D measurement is difficult by vibration while running of the robot. Therefore, in the following experiments, 3D measurements were executed by using LRF. From now on, an ingenious stabilization method of the measurement of stereovision will be needed to construct high accuracy and robust measurement method for autonomous robot under outdoor environment.

3.2 Measurement of target by LRF

A detailed three dimensional measurement is necessary for manipulation of the target by the robot. Here, the method of three dimensional measurement by the LRF without affected by environmental light is described. It is necessary to extract the data set of the target object from measurement data set for the analysis of the object. We employ PSF (Plane Segment Finder) method (Okada et al., 2001) or RHT (Randomized Hough Transform) (Xu et al. 1993, Ding et al. 2005) to extract and eliminate planes such as floor surface from the range data set. A concrete procedure is shown as follows.

First, let $^{r}x_{ij} \triangleq (^{r}x_{ij}, {}^{r}y_{ij}, {}^{r}z_{ij})^{T}$ be 3D points measured the LRF and the indices of these points $i = 1, \cdots, I$ and $j = 1, \cdots, J$ represents the measurement order. The relations of data points are shown in Fig. 8(a), and Fig. 9 shows an example of measurement range data set of a plastic bottle. As the figure indicates, only the surrounding of the target is measured in high density by the rotating of the LRF. Next, to extract and remove the floor surface from this measurement data, the above mentioned PSF method is applied to this data according to the following procedures.
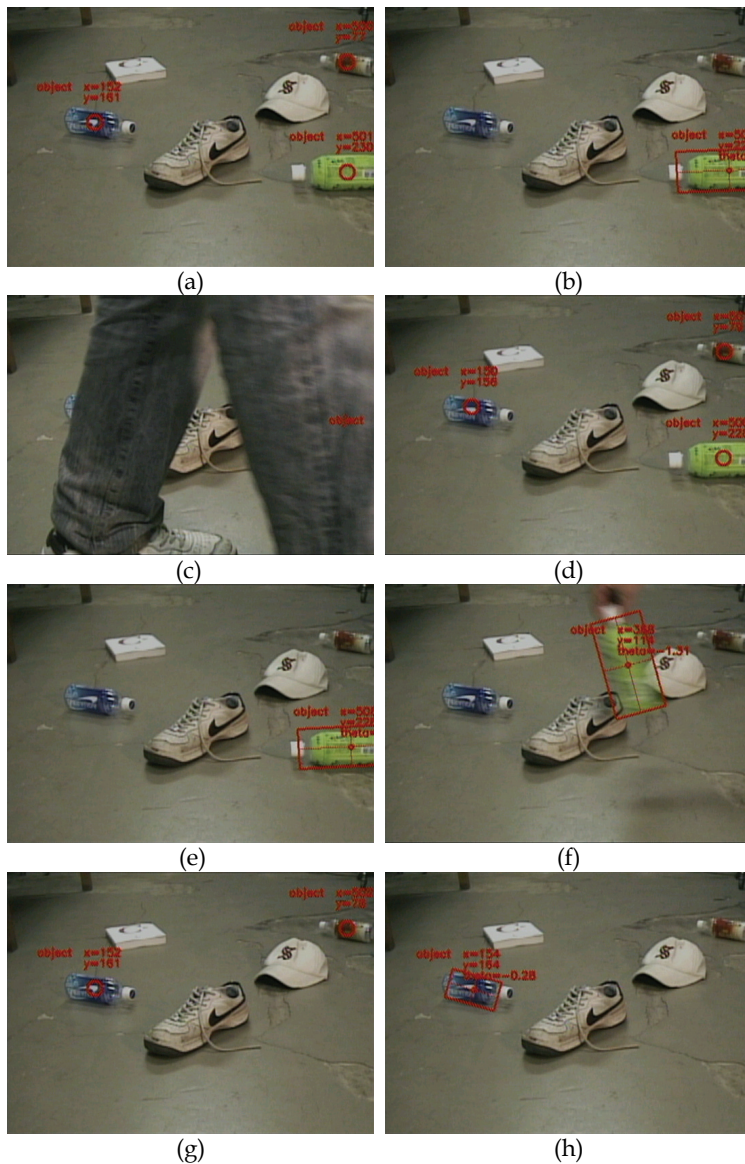
Fig. 7. Example of image processing: (a) the objects had been detected by the eigen space method; (b) the nearest object had been continuously and stably detected and it had been set as a target for tracking; (c) the target had been lost sight of by a pedestrian; (d) the objects were detected again; (e) the target was tracked by CAMSHIFT algorithm; (f) the target was correctly tracked though the object moved at high speed; (g) the target was excluded from the scene and the objects were detected again; (h) the target was tracked by CAMSHIFT algorithm.
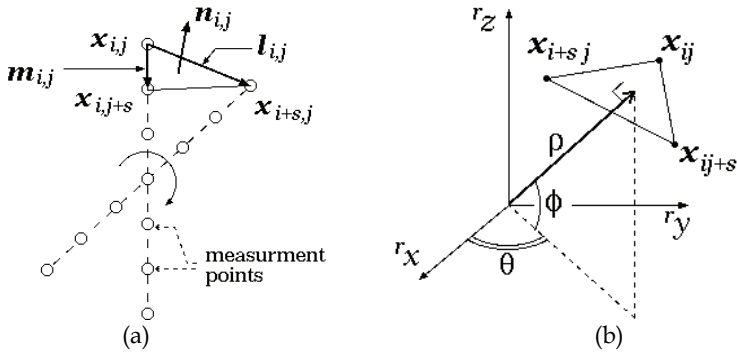
Fig. 8. Relations of 3D data points and a plane constructed by them: (a) relation of the measurement points; (b) relations of the coordinate system and Hough parameters.
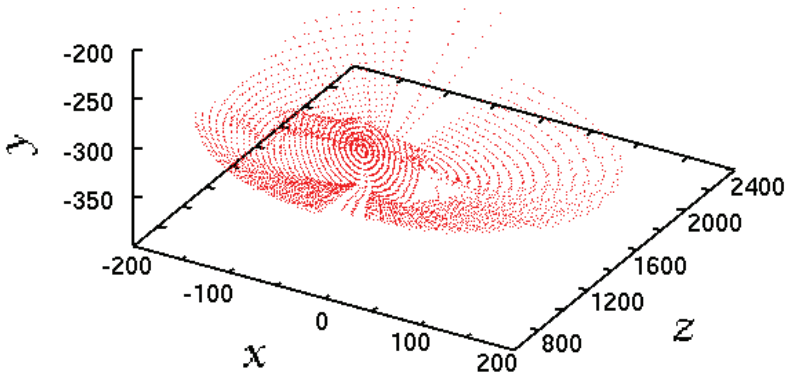
Fig. 9. Range data set of a detected plastic bottle.

Namely, the plane parameters $\theta_{ij}\ (0 \leq \theta_{ij} < \pi)$ and $\phi_{ij}\ (0 \leq \phi_{ij} < \pi)$ shown in Fig. 8(b) are calculated from the normal vector of the plane including the points $(x_{ij}, x_{i+s,j}, x_{i,j+s})$ as follows,

$$\theta_{ij} = -\tan^{-1}\left(n_{ij}^{y}\big/n_{ij}^{x}\right) \tag{19}$$

$$\phi_{ij} = -\tan^{-1}\left(n_{ij}^{z}\big/\left(n_{ij}^{x2}+n_{ij}^{y2}\right)\right)^{\frac{1}{2}} \tag{20}$$

where,

$$n_{ij} = \frac{l_{ij} \times m_{ij}}{\left\|l_{ij} \times m_{ij}\right\|} \triangleq (n_{ij}^{x},\ n_{ij}^{y},\ n_{ij}^{z})^{T}, \tag{21}$$

$$l_{ij} = x_{i+s,j} - x_{ij}, \tag{22}$$

$$m_{ij} = x_{i,j+s} - x_{ij},$$ (23)

and s represents an interval of index of data point. Here, we set $s = 2$ and calculated normal vectors n_{ij}, and the distribution of them are shown in Fig. 10(a). We see from this figure that the normal vectors were distributed around $n_{ij} = (0, 1.0, 0)^T$, however the specification of the peak of the distribution is difficult. Therefore, the values of $(\theta_{ij}, \phi_{ij}, \rho_{ij})$ in Hough space are calculated from these normal vectors. Namely, the values of ρ_{ij} are obtained by the following expression.

$$\rho_{ij} = (x_{ij} \cos \theta_{ij} + y_{ij} \sin \theta_{ij}) \cos \phi_{ij} + z_{ij} \sin \phi_{ij}$$ (24)

Since the variance of ρ_{ij} caused by the measurement noise is relatively large, it is also difficult to specify the peak (Fig. 10(b)). Therefore, the values of (θ_{ij}, ϕ_{ij}) are normalized and mapped into $\theta - \phi$ space. Moreover two dimension histogram of 36×36 division is generated and the peak cell of (θ_{ij}, ϕ_{ij}) is searched. In the histogram shown in Fig. 11(a), it is found that the peak value $C_{\theta_{pF}, \phi_{pF}}$ exists at $(\theta_{pF}, \phi_{pF}) = (90, 0)$.
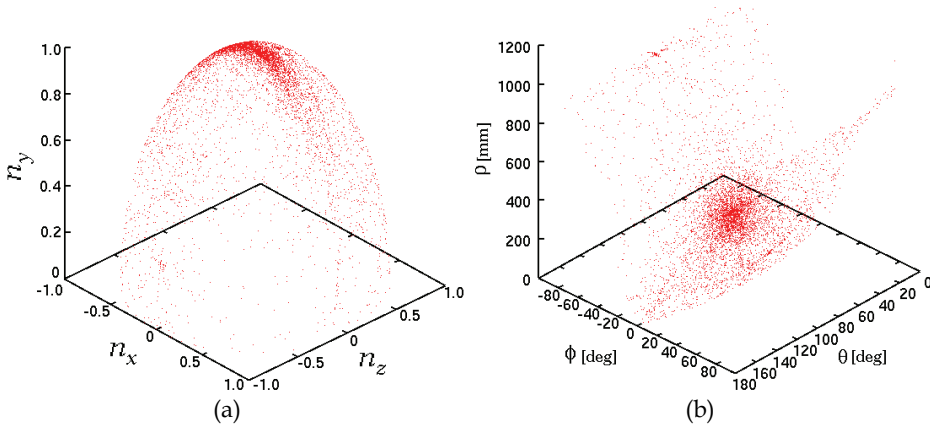
Fig. 10. Distribution of vectors; (a) normal vectors; (b) $(\rho_{ij}, \theta_{ij}, \phi_{ij})$ vectors.

Next, the following processing is executed for only the measurement points with the value within a certain range Δ_C (=50 [deg]) around the peak value. The Hough parameters of these measurement points are replaced with the peak value, i.e.,

$$\theta_{ij} := \theta_{pF} \quad \text{if } (\theta_{pF} - \Delta_C/2) \le \theta_{ij} \le (\theta_{pF} + \Delta_C/2),$$ (25)

$$\phi_{ij} := \phi_{pF} \quad \text{if } (\phi_{pF} - \Delta_C/2) \le \phi_{ij} \le (\phi_{pF} + \Delta_C/2),$$ (26)

and the following values are calculated.

$$\rho'_{ij} = (x_{ij} \cos\theta_{pF} - y_{ij} \sin\theta_{pF}) \cos\phi_{pF} + z_{ij} \sin\phi_{pF}. \tag{27}$$

The measurement points with ρ'_{ij} close to the peak value $\hat{\rho}_{pF}^{max}$ can be estimated to be on the same plane. Namely, the measurement points on the floor surface can be excluded by eliminating these measurement points. The histogram of the values ρ'_{ij} is shown in Fig. 11(b). We employed the discriminant analysis method to separate the histogram, and then it was separated at 300 [mm]. The separated points with $\rho'_{ij} \geq 300\,[\text{mm}]$, i.e. the measurement points on the floor surface are shown in Fig. 12(a), and other measurement points are shown in Fig. 12(b). It was found from these results that planes such as floor surface can be excluded from the target data set by this procedure. As a consequence, the robot can know size, position, and posture of the target object by the extracted range data.

(a)

(b)
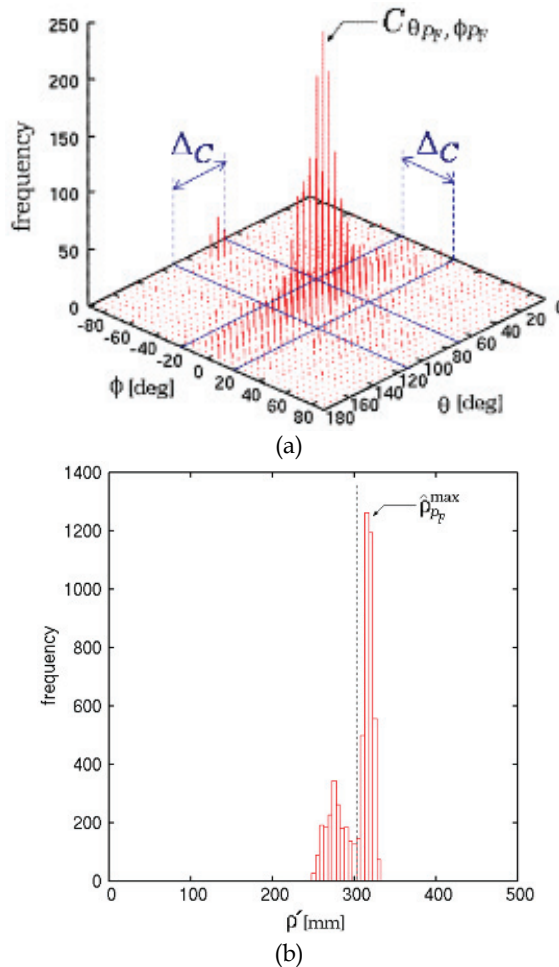
Fig. 11. Histograms: (a) normalized Hough parameter (θ_{ij}, ϕ_{ij}); (b) values of ρ'_{ij}.

3.3 Simple measurement by one camera and LRF

This sensor system can simply track and measure the distance of the object by using only a camera and the LRF. Namely, the object is detected and tracked by using only one camera image, and the posture of the sensor system is controlled to overlap the center of the object and the center of the image. At this time, since the pixels of the camera image corresponding to the measurement points of the LRF can be easily found from Eqn. (8), (9), (10), the distance to the object can be known by using a part of the measurement result of the LRF. An example of processing image of this application is shown in Fig. 13. In the rectangular at the upper right of the image, the distance to the object (D. to Object), the pixel position of center of the object (C. of pix.), the posture of appearance of the object (Theta), the size of the object (Area), and the posture angle of the sensor system (Yaw, Pitch, Roll) are displayed. Moreover, it was confirmed that these information about the target object was calculated in the video rate (30 [fps]).

(a)

(b)
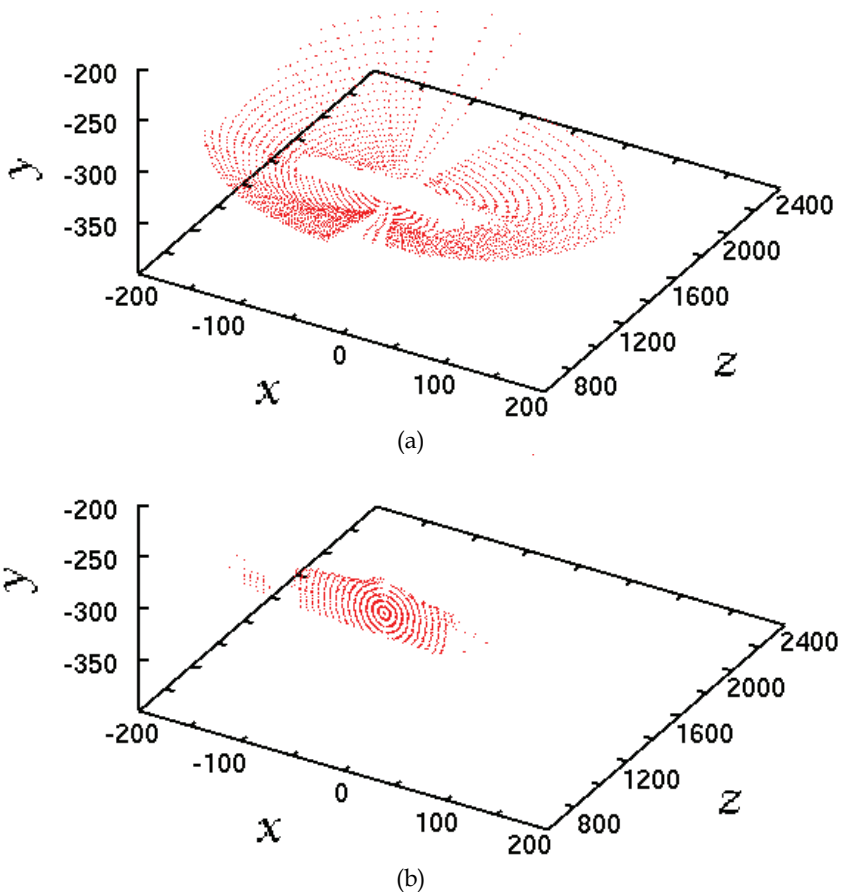
Fig. 12. Example of measurement of a plastic bottle: (a) results of extraction of the floor surface; (b) range data set of the plastic bottle.
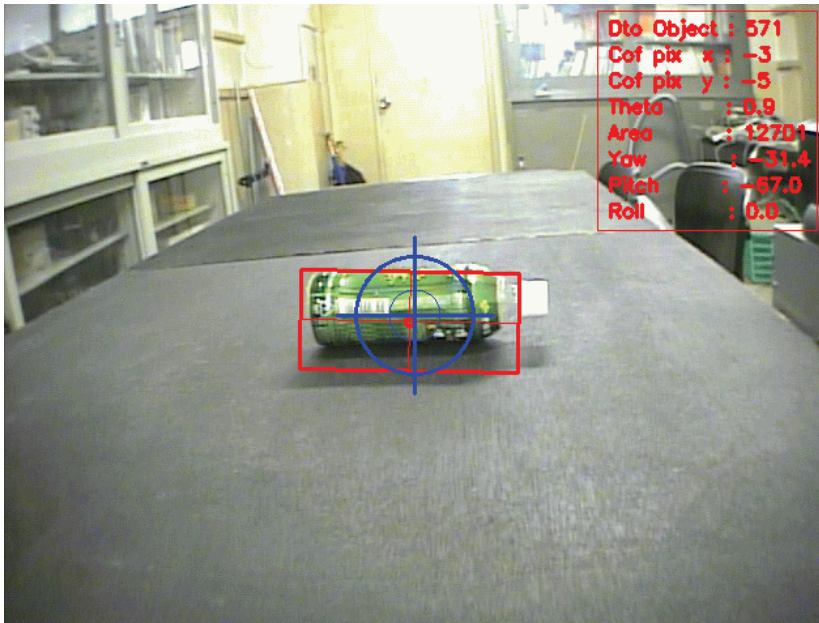
Fig. 13. Example of the detection and the measurement of a plastic bottle. The scope mark shows the center of the image, and the square around the plastic bottle shows a detected area of the object.

4. Installation for outdoor service robot OSR-02

Since November 28th of 2003, the Japanese government has designated the Fukuoka Prefecture as the one of the special zones for robot research and development. There researchers are allowed to operate their robots in public areas by applying for police permission. Thus, for example, in shopping areas in the city, we can take advantages of executing realistic experiments of outdoor service robots for cleaning up streets, welcoming customers, guiding them to a certain place, providing information about the shopping area, and so on. Under such circumstances, we have started developing the outdoor service robots called OSR-01 (Obata et al., 2006) and OSR-02 (Nishida et al., 2006) intended for cleaning up shopping streets by means of the separated collection of discarded trash, such as plastic, glass, and steel, etc. on shopping streets.

The outdoor service robot which we call OSR-02 (Fig. 14) had been developed intended for cleaning up urban areas by means of collecting discarded trash such as plastic bottles, cans, plastic bags and so on. So far, several type robots for the cleaning work have been developed; however, they only vacuum on the specified route. Since OSRs recognize and collect trash or target objects with the manipulator in shopping street etc., the technology that composes the OSRs is greatly different. Although many operations have been programmed for the above task, we, in this section, mainly show the sensor system and the manipulation system of OSR-02 for trash detection and distance measurement of the target after giving a briefing about the hardware of the robot.

(a)
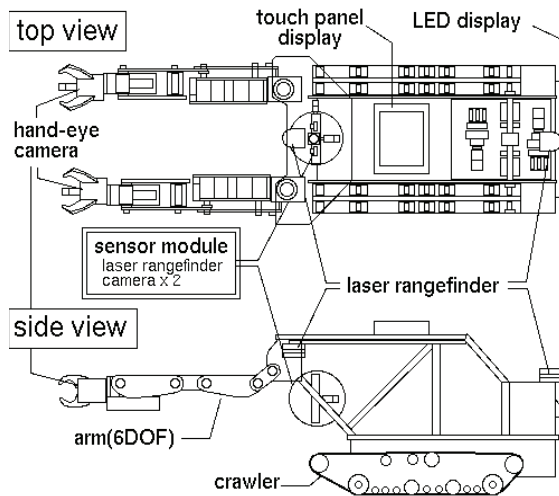
(b)

Fig. 14. Outdoor service robot OSR-02: (a) whole image of OSR-02; (b) configuration.

4.1 Structure

OSR-02 has two manipulators and crawlers. The length of the manipulators is 850[mm] and they have five degrees of freedom and a hand. The specification of OSR-02 is shown in Table 2 and its electrical structure is shown in Fig. 15. Moreover, five computers for control of the crawler, the manipulators, and the sensor system are installed in OSR-02, and they communicate via Ethernet. Furthermore, in order to recognize the surrounding environment, and not to harm surrounding persons, several sensors had been installed in the robot.

D.O.F. of manipulator		5 (x2)
hand		1 (x2)
drive wheel		2
dimension	height	600[mm]
	width	500[mm]
	depth	800[mm]
manipulator length		850[mm]
weight		90[kg]
running speed		2.0 [km/h]
sensor	LRF	3
	color CCD camera	4
	force sensor (hand)	2
	magnetic sensor (hand)	2

Table 2. Specification of OSR-02
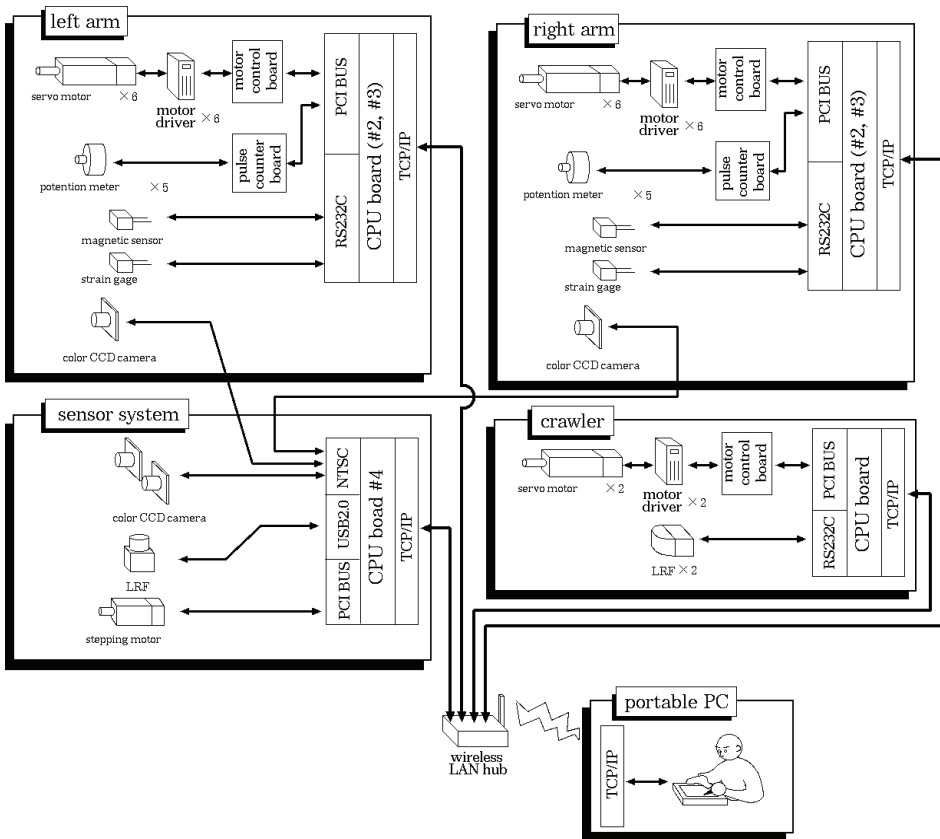
Fig. 15. Electrical structure of OSR-02

4.2 Equipped devices

The main equipped devices of OSR-02 are shown in Fig.16. Fig. 16(a) shows the installed sensor system, and it is used for the running route inspection and the trash detection. The right hand is shown in Fig. 16(b), and a hand-eye camera is installed in the center position of claws. The image obtained from the camera is used for the hand-eye system driven for the grasping of target. Moreover, a magnetic sensor and a strain gage have been installed into a claw of each hand as shown in Fig.17. The sorted collection of the grasping objects and the damage prevention of the hands by controlling of the grasping power are possible by using these sensors. The rear and the front LRF for the obstacle detection of the robot are shown in Fig. 16(c) and (d), and the horizontal scanning range of each LRF is 180 [deg] by 10 [Hz]. The results of the image processing in the robot can be displayed on the rear monitor shown in Fig. 16(d). The publicity LED panel shown in Fig. 16(e) displays the content of the job of the robot, the advertisement of the shopping street, and so on. It is possible to give the command to OSR-02 by the touch panel of removable laptop PC via wireless LAN (Fig. 16(f)).
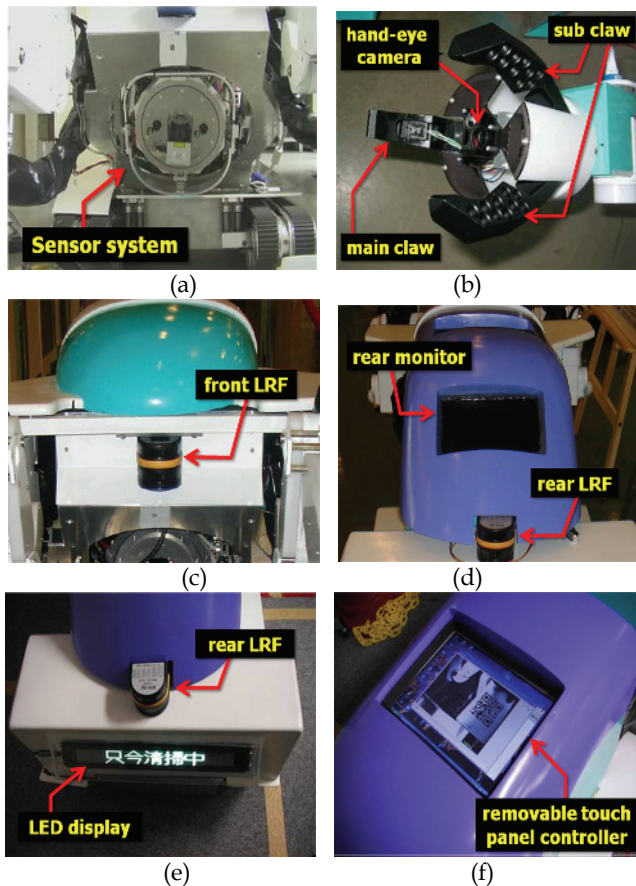
Fig. 16. Equipped devices of OSR-02: (a) installed sensor system; (b) right hand; (c) front LRF; (d) rear monitor and rear LRF; (e) LED display; (f) removable touch panel controller.

(a)

(b)

(c)

(d)
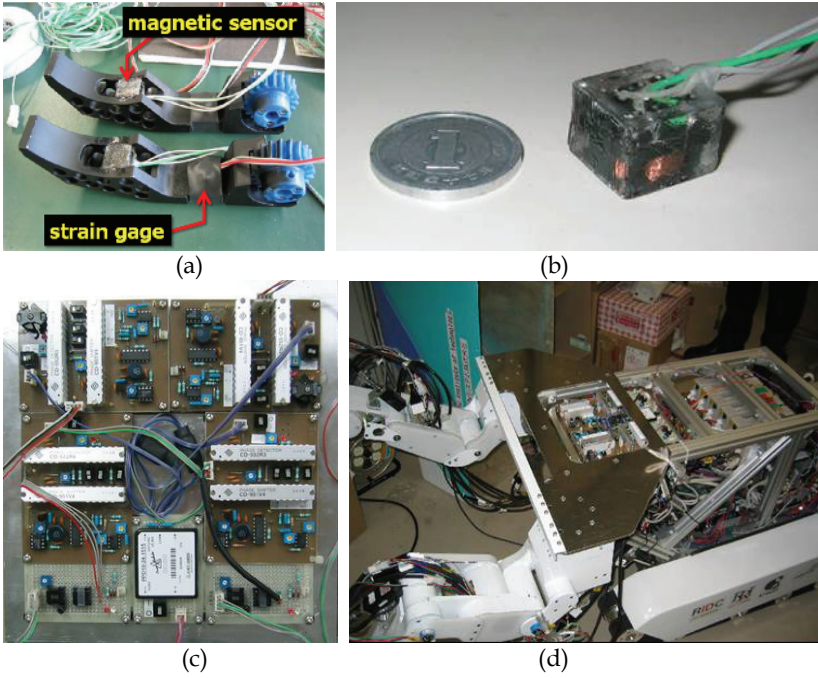
Fig. 17. Magnetic sensor for trash separation: (a) main claws of each hand with magnetic sensor and strain gage; (b) magnetic sensor; (c) sensor circuit; (d) the circuit built into the robot.

4.3 Posture control of sensor system

We constructed a visual feedback running control system by using the sensor system for OSR-02 to approach the target object. It is necessary to control the posture of the sensor system so as not to lose sight of the target until the robot approaches it. When the target is detected at $(x_d(t), \; y_d(t))^T \in \sum_{sl}$ on the left camera screen at time t, the yawing angle $\theta(t)$ and the pitching angle $\phi(t)$ of the sensor system are controlled as follows,

$$\begin{pmatrix} \dot{\theta}(t) \\ \dot{\phi}(t) \end{pmatrix} = \begin{pmatrix} -k_{01} & 0 \\ 0 & -k_{11} \end{pmatrix} \begin{pmatrix} \theta(t) \\ \phi(t) \end{pmatrix} + \begin{pmatrix} -k_{02} & 0 \\ 0 & k_{12} \end{pmatrix} \begin{pmatrix} x_d(t) \\ y_d(t) \end{pmatrix}, \tag{28}$$

where, gain coefficients $k_{01}; k_{02}; k_{11}; k_{12} > 0$ are design parameters, and the bias between the rotation center of the sensor system and the optical center of the cameras are disregarded.

4.4 Running control system

The relation of the coordinate systems on the robot and the target position is shown in Fig. 18(a). The angular velocity input $w(t) \triangleq (w_l(t), \; w_r(t))^T$ for approaching the target is controlled as follows,

$$w = R_g v, \qquad (29)$$

where,

$$R_g = \frac{1}{r_w}\begin{pmatrix} 1 & l \\ 1 & -l \end{pmatrix}, \quad v = \begin{pmatrix} v_g \\ w_g \end{pmatrix}, \qquad (30)$$

r_w is radius of drive wheel in the crawler, and l is distance between the drive wheel and the center of rotation of the robot. Moreover, v_g and w_g are the velocity and the angular velocity of the center of rotation of the robot, respectively. The velocity input in the robot coordinate system toward the target position is decided as follows,

$$v = K \cdot {}^r d, \qquad (31)$$

where ${}^r d \triangleq ({}^r d_x , {}^r d_z)^T$ is the position of the target object in which height is disregarded. Moreover, K is represented as follows,

$$K = \begin{pmatrix} k_r & 0 \\ 0 & k_\theta \end{pmatrix}, \qquad (32)$$

where, k_r and k_θ are gain coefficients about distance from the target object and about yawing angle θ of the sensor system, respectively. The values of k_r and k_θ are adjusted according to the situation of the road. Therefore, the velocity of each wheel are determined by Eqn. (29) and Eqn. (31) as follows,

$$w = R_g K^r d \qquad (33)$$

where the value of w does not exceed the maximum velocity set beforehand. A picture of the rear monitor of the robot at the time of running by the above visual feedback control system is shown in Fig. 18(b).

4.5 Target manipulation by hand-eye system

After the robot approached to a constant distance from the target, it raises the arm and takes up an initial posture to grasp the target (Fig. 19(a)). When the manipulator is in the initial posture, the robot views the target by the hand-eye camera and measures grasping position and posture of it. The relation between the hand-eye camera and the target object is shown in Fig. 19(b). The hue value obtained by the sensor system in a prior running sequence is used to identify the target region on the right hand-eye screen. Namely, the region with the specific hue value tracked in a prior sequence is searched on the hand-eye camera image, and only the region with the maximum size is set as the target region. Then the minimum square that suits this region, the center position $x_e = (x_e , y_e)^T \in \Sigma_{hsr}$ and the posture angle θ_h of the target object are calculated. An example of the result of right hand-eye image processing is shown in Fig. 19(c). Here, the height h_h of the hand in the initial posture is always constant, and the height h_t of the target object can be estimated from the kind of

trash (e.g. $h_h = 65[\text{mm}]$ for a plastic bottle). Therefore, the grasping position $x_t \in \Sigma_{hr}$ of target can be calculated as follows,

$$x_t \triangleq \begin{pmatrix} x_t \\ y_t \\ z_t \end{pmatrix} = \begin{pmatrix} -\alpha x_e z_t / f_h \\ \alpha y_e z_t / f_h \\ h_h - h_t \end{pmatrix}, \tag{34}$$

where f_h is focal length of the hand-eye camera and α is a coefficient to convert the unit from the pixel into the distance. Here, the hands had been designed so that the right hand coordinate system Σ_{hr} overlaps with the hand-eye coordinate system Σ_{hsr} as shown in Fig. 19(c), namely, rotation and bias of them are able to be disregarded. Next, the hand and claws are controlled for grasping and collection of the target object by a feed forward control system, therefore the joint angles of the right arm θ_{hrj} $(j = 1, \cdots, 5)$ shown in Fig. 19(d) are controlled to be $x_t \to 0$ and $\theta_{hr5} \to \theta_h$. Moreover, appropriate holding of the target is confirmed by the strain gauge, and the above-mentioned sequence is repeated again when the robot failed in the holding.

(a)   (b)
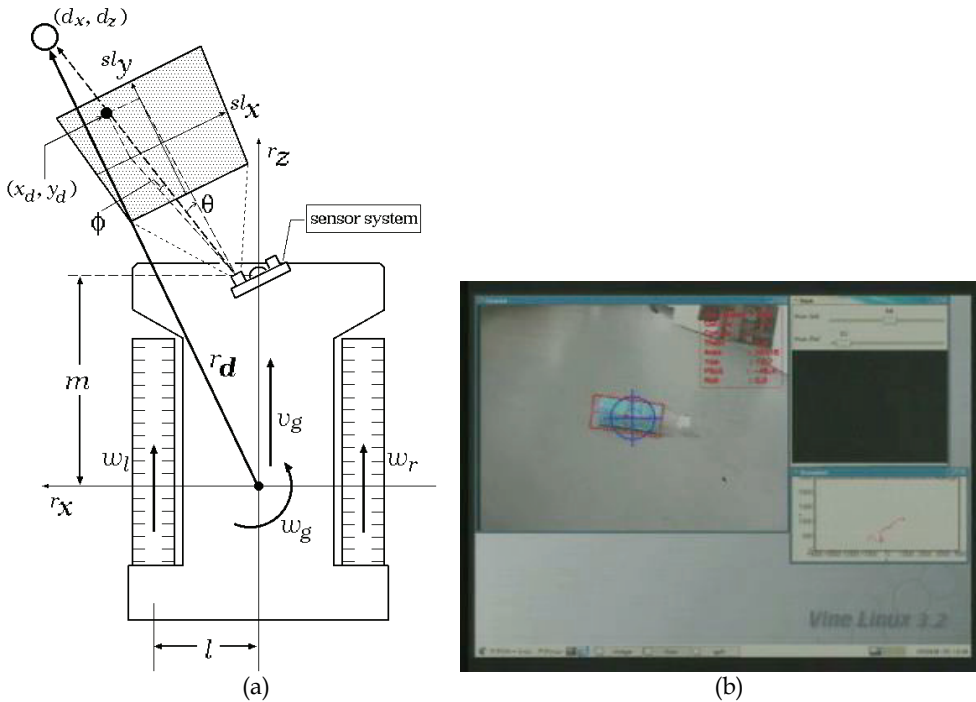
Fig. 18. Running control system of OSR-02: (a) relation of the target position and the robot coordinate system; (b) a picture of the rear monitor during approaching to the target object, upper left window is a left camera image, upper right window is a right hand-eye camera image, and lower right window is a measurement data of the LRF.

<center>(a)   (b)</center>

<center>(c)   (d)</center>
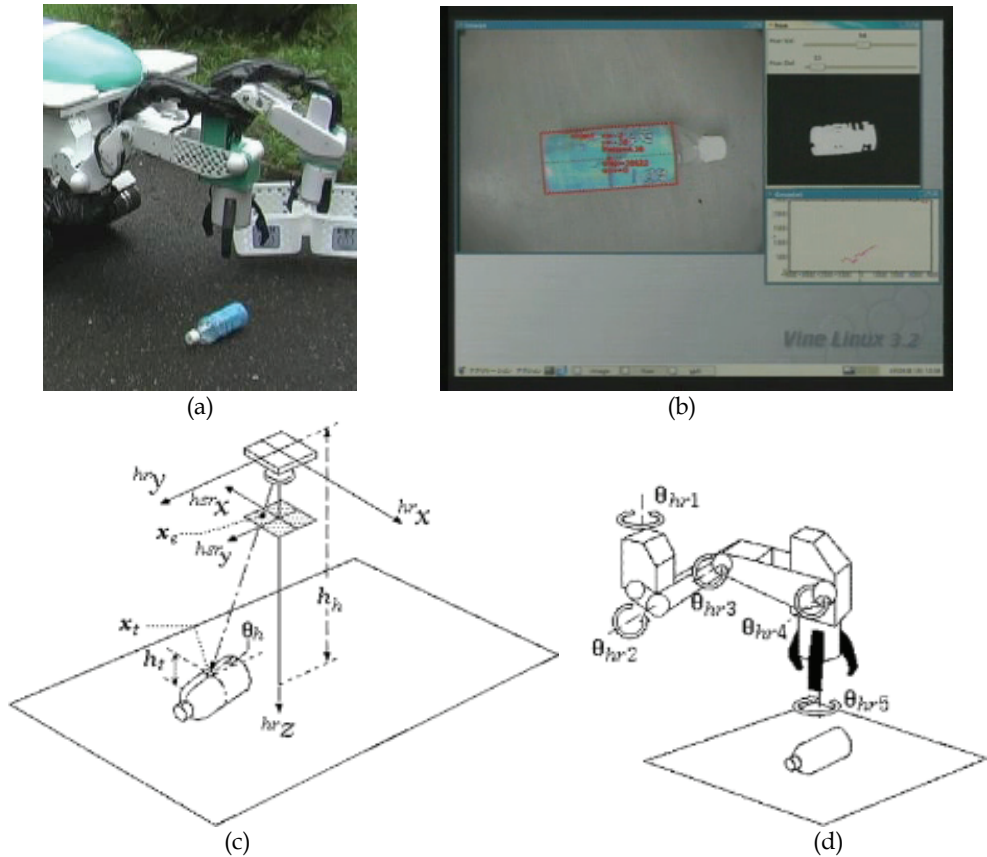
Fig. 19. Hand-eye system: (a) initial posture for grasping the target; (b) a picture of the rear monitor during the grasping action, the information about the target is displayed by the red lines in the upper left window, and the region with the target hue is shown in the upper right window; (c) relation of the hand-eye camera screen coordinate system Σ_{hsr} and the hand coordinate system Σ_{hr} ; (d) joint angles of the right arm.

5 Experiments

5.1 Trash collection task

Trash collection task of OSR-02 was experimented under outdoor environments. Plastic bottles and cans were put forward of about 2 [m] of the robot, and the collection experiments were conducted. In this experiment, the robot had been programmed to classify the target by the right hand and to store it into the trash box for the separate collection held by the left hand. The weather on the day was a drizzle and the results of the experiment are shown in Fig. 20. It was confirmed that the above mentioned various measurement methods, the control systems, and the operation sequence of them are able to be behaved appropriately by these experiments.
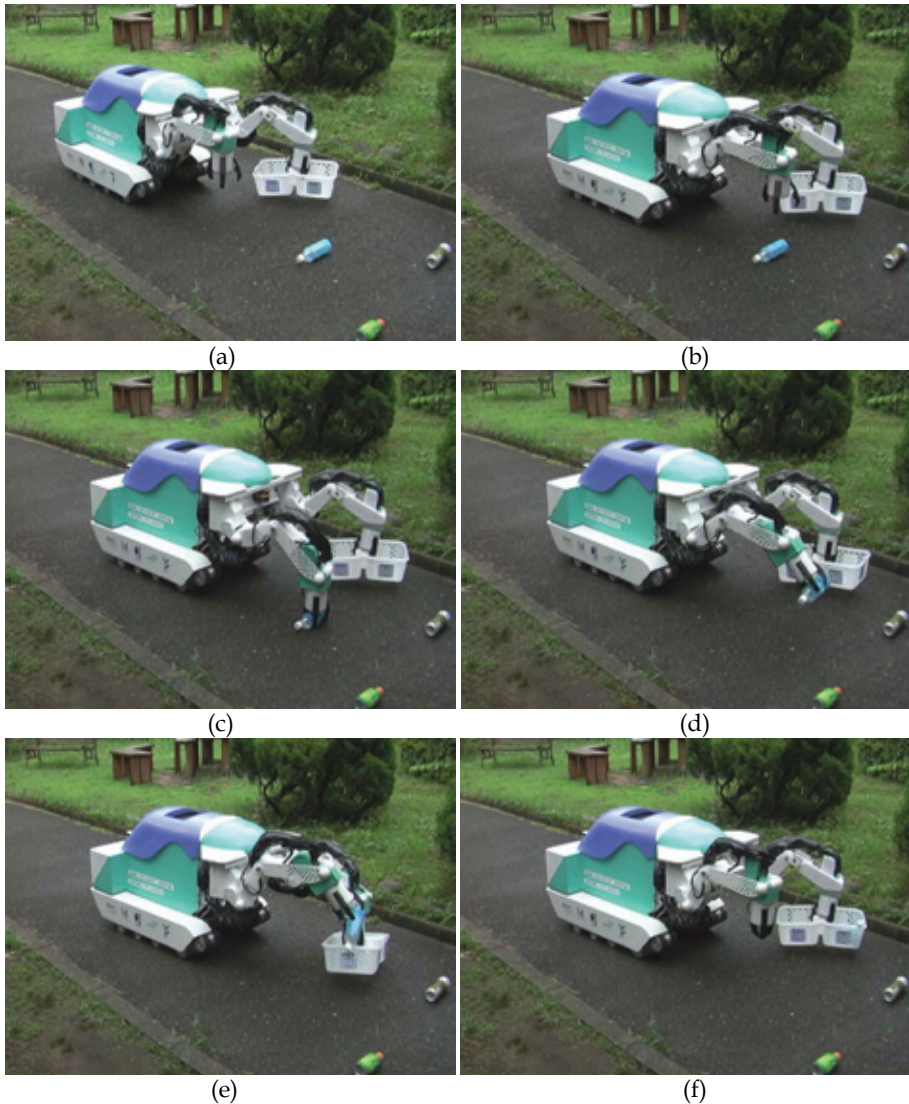
Fig. 20. Trash collection work flow of OSR-02: (a) a plastic bottle had been detected; (b) initial posture for grasping; (c) grasping the target; (d) lift operation; (e) the robot puts the grasped object into the trash box for plastic based on the classification result; (f) searching for next trash.

5.2 Experiments in public space

We demonstrated of OSR-02 several times at a shopping arcade and a playground in Kitakyushu-city (Fig.21). In the demonstrations, the robot went round the shopping arcade and the playground, and collected trash. We confirmed that OSR-02 was able to detect the

plastic bottle, glass bottle, and cans on the street, to approach, to collect accurately, and to avoid the pedestrian. Furthermore, it was found from these experiments that the calculation speed of the developed vision system was fast enough for the collection of trash.

Fig. 21. Demonstrations of OSR-02: (a) at a shopping street; (b) at a playground.

6. Conclusion

In this research, we have proposed a novel sensor system and its applications, and have shown its effectiveness by the fundamental experiments. Moreover, we installed the sensor system into the outdoor service robot OSR-02 and conducted experiments at the shopping street and the playground. Then, it was found from these experiments that the calculation speed of the developed vision system was fast enough for appropriate running and the collection of trash.

In the design of the autonomous robot, the relation between the electricity consumption and the performance is always considered, and the required measurement accuracy differs by tasks. For example, the measurement should be high speed while the robot is moving, and it should be high accuracy while the robot stops and is manipulating the object. This sensor system can achieve such demand of the measurement according to kind of task by switching the measurement method and devices though it is composed of basic sensor devices.

7. Acknowledgements

8. References

Birchfield, S. & Tomasi, C. (1999). Depth Discontinuities by Pixel-to-Pixel Stereo, *International Journal of Computer Vision*, Vol. 35, No. 3, pp. 269-293.

Bradski, R., G., (1998). Computer vision face tracking as a component of a perceptual user interface, *In Workshop on Applications of Computer Vision*, pp. 214–219.

DeSouza, N., G. & Kak, C. (2002). A. Vision for Mobile Robot Navigation: A Survey, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 24, No.2, pp. 237–267.

Ding, Y.; Ping, X.; Hu, M. & Wang, D. (2005), Range Image Segmentation Based on Randomized Hough Transform, *Pattern Recognition Letters*, No. 26, pp. 2033-2041.

Fuchikawa, Y.; Nishida, T.; Kurogi, S.; Kondo, T.; Ohkawa, F.; Suehiro, T.; Watanabe, Y.; Kawamura, Y.; Obata, M.; Miyagawa, H. & Kihara, Y. (2005). Development of a Vision System for an Outdoor Service Robot to Collect Trash on Streets, *Proc. of CGIM05*, pp.100-105.

Hihnel, D.; Burgard, W. & Thrun, S. (2003). Learning compact 3D models of indoor and outdoor environments with a mobile robot, *Robotics and Autonomous Systems*, Vol. 44, pp. 15–27.

Kondo, T.; Nishida, T.; Obata, M. & Ohkawa, F. (2005). A Research Report about The Outdoor Service Robot OSR-01, *Proc. of The 1st International Conference on Design Engineering and Science Vienna*, pp. 271-275.

Xu, L. & Oja, E. (1993). Randomized HoughTransform (RHT): Basic Mechanisms, Algorithms and Complexities, *Computer Vision, Graphics, and Image Processing: Image Understanding*, Vol.57, No.2, pp. 131-154.

Nishida, T.; Takemura, Y.; Fuchikawa, Y.; Kurogi, S.; Ito, S.; Obata, M.; Hiratsuka, N.; Miyagawa, H.; Watanabe, Y.; Koga, F.; Suehiro, T.; Kawamura, Y.; Kihara, Y.; Kondo, T. & Ohkawa, F. (2006). Development of Outdoor Service Robots, *Proc. of SICE-ICASE International Joint Conference*, pp. 2052-2057.

Nishida, T.; Takemura, Y.; Fuchikawa, Y.; Kurogi, S.; Ito, S.; Obata, M.; Hiratsuka, N.; Miyagawa, H.; Watanabe, F.; Suehiro, T.; Kawamura, Y. & Ohkawa, F. (2006). Development of Outdoor Service Robots, *Proc. of SICE-ICASE International Joint Conference*, pp. 2687-2691.

Obata, M.; Nishida, T.; Miyagawa, H.; Kondo, T. & Ohkawa, F. (2006). Development of Outdoor Service Robot to Collect Trash on Streets, *IEEJ Trans. EIS*, Vol. 126, No. 7, pp.840-848.

Okada, K.; Kagami, S.; Inaba, M. & Inoue, H. (2001). Plane Segment Finder: Algorithm, Implementation and Applications, *Proc. of Int. Conf. on Robotics and Automation*, pp. 2120-2125.

Surmann, H.; Lingemann, K.; Nuchter, A. & Hertzberg, J. (2001). A 3D laser range finder for autonomous mobile robots, *Proc. the 32nd ISR*, pp. 153–158.

Surmann, H.; Nuchter, A. & Hertzberg, J. (2003). An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments, *Robotics and Autonomous Systems*, Vol. 45, pp. 181–198.

Uenohara, M. & Kanade, T. (1997). Use of Fourier and Karhunen-Loeve Decomposition for Fast Pattern Matching With a Large Set of Templates, *IEEE trans. on pattern analysis and machine intelligence*, Vol. 19, No. 8, pp. 891–898.

Real-time Map Update Using Pose Reliability of Visual Features

Joong-Tae Park, Yong-Ju Lee and Jae-Bok Song
Korea University
Korea

1. Introduction

Autonomous navigation of a mobile robot consists of many basic techniques such as mapping, localization, path planning, collision avoidance, system architecture and so on. Among these, localization is the most important task since a robot must know its pose to reach the desired destination reliably. Localization is a method for estimating the robot pose using the environmental map and the sensor information. Therefore, localization performance increases as the differences between the map and the real environment decrease. Representative examples of map matching based localization are as follows.

MCL (Monte Carlo Localization) method [1][2], which robustly estimates the robot pose, compares the information from the sensors mounted on the robot with the environment map. The vision-based SLAM using the SIFT (Scale Invariant Feature Transform) algorithm [3] based on a stereo camera was also proposed [4] [5].

The above localization methods have been applied to many mobile robots and their performances were verified. The localization schemes, however, tend to show poor performance when the map is different from the real environment due to artificial or natural changes in the environment. If the robot can detect such changes occurring in the environment and reflect them on the map, navigation performance can be maintained even for the environmental changes. In this research, a new method for recognizing the environmental changes and updating the current map is proposed. With this approach, the robot can navigate autonomously with high reliability and thus offer better services to humans.

Despite the importance of map update, little attention has been paid to the update algorithm of the constructed map. This paper proposes a method for updating the constructed map reliably and simply. The particle filter algorithm [6], which has been used for localization, is adopted for the map update. If the robot recognizes a visual feature, new samples representing the candidates for the robot pose are drawn around the visual feature. After newly drawn samples converge, the similarity between the poses of new samples and those of the current robot samples is evaluated. The pose reliability of the recognized object is calculated by applying the similarity to the Bayesian update formula [7]. Then the object whose pose reliability is below the predetermined value is discarded. On the other hand, the new position of the moved visual feature is registered to the visual feature map if its pose reliability is greater than the predetermined value.

The remainder of this paper is organized as follows. Section 2 illustrates an overview of the navigation system which is the main framework of this research. Section 3 introduces the concept of the intelligent update of a visual map. Experimental results are shown in section 4 and finally in section 5 conclusions are drawn.

2. Overview of navigation system

This section overviews the navigation system so as to help to understand the proposed intelligent update of a visual map. The autonomous navigation system used in this research works based on a range sensor and a vision sensor. Figure 1 shows the structure of the integrated navigation system. This system is classified into two parts; a vision framework and a navigation framework. Each framework consists of general components which are segmented in a task unit and a control component which supervises general components. When a robot receives the order to move to the goal, the navigation system activates the 'Mobile Supervisor' component and the 'Vision Supervisor' component. Detection of the environmental changes and the map update are executed in the 'Localizer' component and the 'MapBuilder' component, as shown in Fig. 1. With this method, the robot is able to perceive the changes occurring in the environment by itself during autonomous navigation.
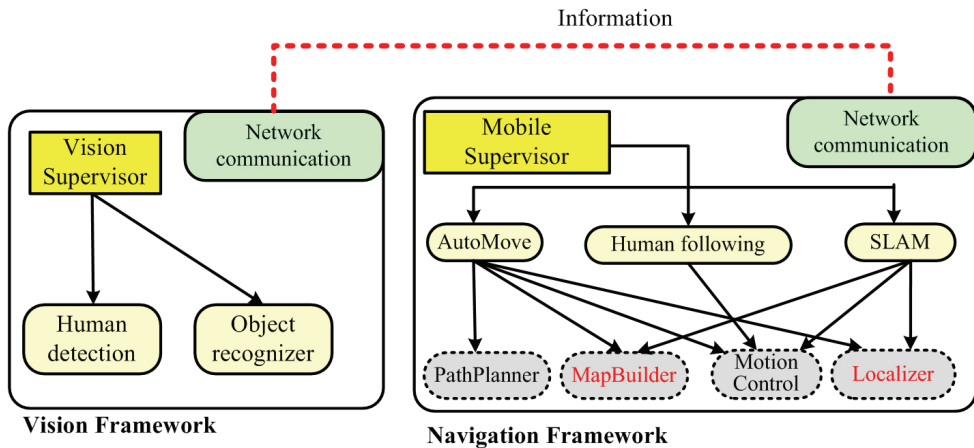
Fig. 1. Architecture of navigation system.

The operation scheme of the navigation system is as follows:
Step 1: Control component loads 'AutoMove' component.
Step 2: AutoMove component loads specific modules (Localizer, PathPlanner, etc.).
  Repeat from Step 3 to 6 until the robot reaches the goal.
Step 3: Estimate the current robot pose from 'Localizer.'
  (a) Obtain visual information from 'Object recognizer.'
  (b) Detect environmental changes.
Step 4: 'MapBuilder' constructs the map.
  (a) Update a grid map.
  (b) Update a visual map.
Step 5: 'PathPlanner' generates a path to the goal.
Step 6: Command translational and rotational velocities to 'MotionControl.'

3. Intelligent update of a visual map

3.1 Problem statement

Range-based localization tends to fail when many objects in the environment cannot be detected by range sensors. In order to overcome this problem, sensor fusion based localization, which combines range information and visual information, is adopted in this research [8]. A brief explanation on this sensor fusion is described in the following paragraph.
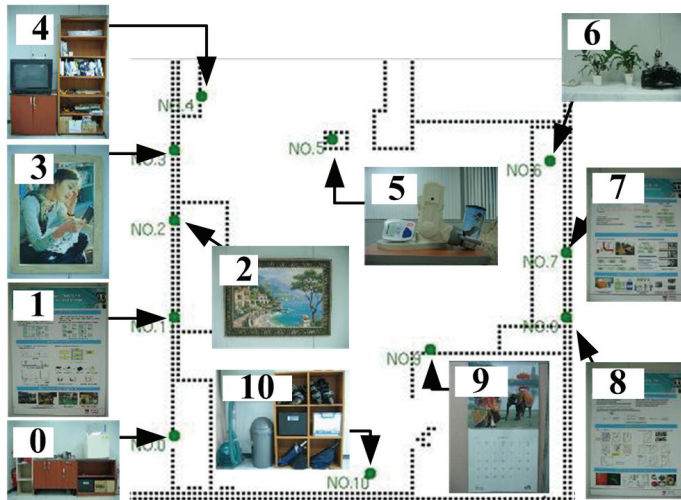
Fig. 2. Hybrid grid/visual map of environment.

(a)

(b)
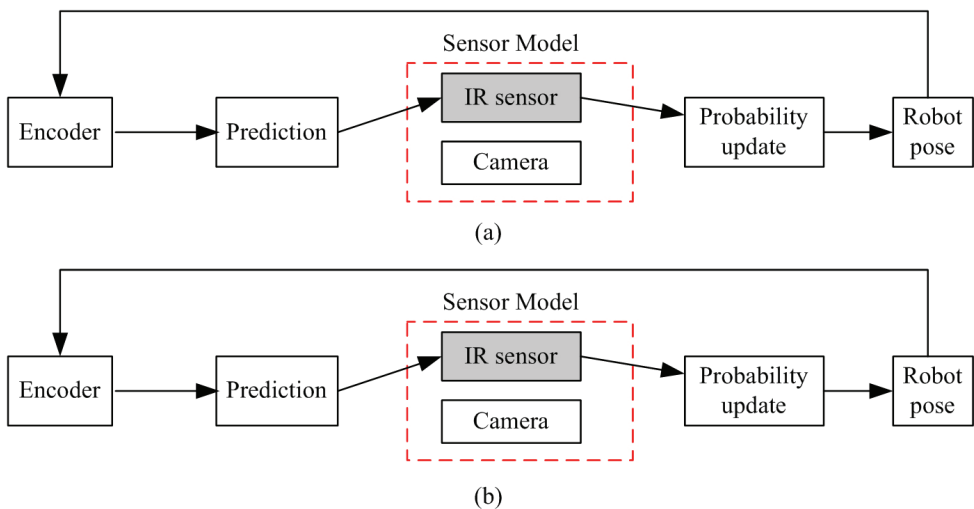
Fig. 3. Sensor models; (a) without and (b) with visually recognized objects.

With a vision sensor, a robot recognizes the objects stored in the database, as shown Fig. 2 and estimates its pose by fusing the visual and range information. However, the objects which can be used as visual features are limited in the real environment. Thus, if there is no visually recognized object, the robot has to estimate its pose with the range sensor alone, as shown in Fig. 3(a). If the robot recognizes objects stored in the database, however, the robot estimates its pose by fusing the visual and range information, as shown in Fig. 3(b). The method of object recognition used in this research is based on the SIFT algorithm, which extracts the feature points that are scale and rotation invariant. Either the range-based or vision-based scheme alone cannot overcome these sensor limitations; sensor fusion based localization should be implemented to compensate for the shortcomings of each sensor. However, if the visual information is not correct, performance of sensor fusion based localization can be worse than that of the range-based localization. For example, Fig 3(a) shows localization with information of a range sensor alone. The ellipse enclosing the robot represents its pose uncertainty. Figure 3(b) represents the case when the robot uses information of both sensors, but the object recognizer provides wrong information because of either false matching or the change in position of object 1. Note that false matching means the robot mistook object 2 for object 1. If both pieces of information were correct, the pose uncertainty would be decreased. When compared to Fig. 4(a), however, the pose uncertainty in Fig. 4(b) increased due to the wrong information from the camera.

(a)   (b)
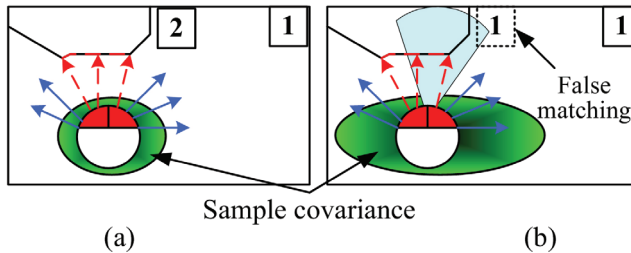
Fig. 4. Problem of localization due to wrong information; (a) localization with range information alone, and (b) localization with wrong visual information.

3.2 Detection and map update

The localizer not only estimates the robot pose, but also detects the environmental changes. The method for detecting the environmental changes is explained below in detail. The robot recognizes the object which is registered on the visual feature map. Then new random robot samples (NR_{sample}), which are the candiates for the robot pose, are drawn near the recognized object, as shown in Fig. 5(a). The area of the newly distributed samples are restricted to the circle with a radius of the measured range and centered at the recognized object. The number of samples is 300. After the new samples converge as shown in Fig. 5(b), the similarity between the poses of the new robot samples (NR_{sample}) and those of the current robot samples (R_{sample}) are evaluated. The similarity can be obtained by

$$p(R, NR, i) = \frac{r}{d} \tag{1}$$

where r is the radius of convergence bound for R_{sample}, and d is the distance between the means of R_{sample} and NR_{sample}. The probability $p(R, NR, i)$ represents the similarity between

R_{sample} and NR_{sample} when NR_{sample} converges with the information of the i-th object. If NR_{sample} exists in the convergence bound as shown in Fig. 6(a), which means $d < r$, the similarity is set to 1. As shown in Fig. 6(b), the similarity approaches 0 as the two samples become apart from each other.

(a)   (b)
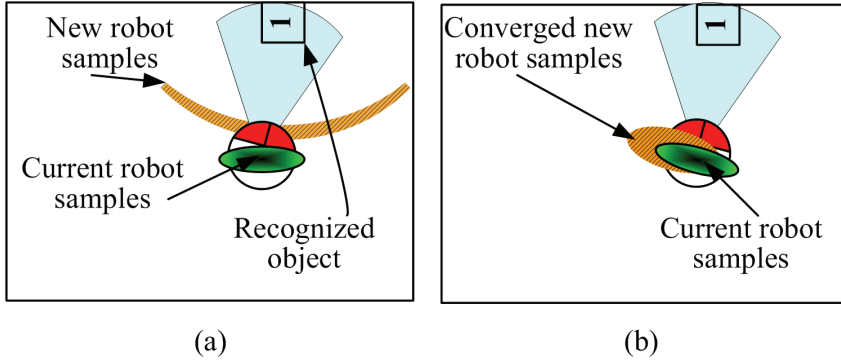
Fig. 5. Example of detecting environmental changes.

(a)   (b)
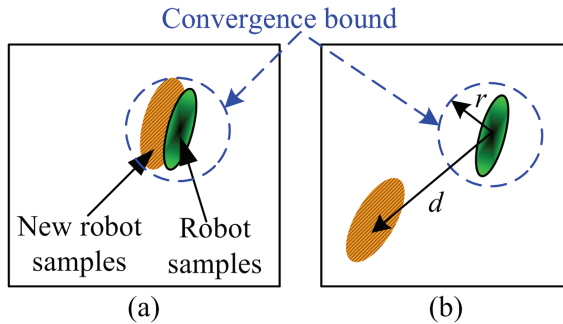
Fig. 6. Example of similarity between new and current robot samples.

The pose reliability of the recognized object is calculated by substituting the similarity into Bayesian update formula as follows:

$$p_{t+1,i} = \frac{p(R,NR,i) \times p_{t,i}}{p(R,NR,i) \times p_{t,i} + \{1-p(R,NR,i)\} \times (1-p_{t,i})} \tag{2}$$

where $p_{t,i}$ is the accumulated pose reliability of object i at time t. The pose reliabilities of all objects are initialized to 0.5 and are continuously evaluated during navigation. The pose reliability serves as a criterion which determines whether the specific visual feature is updated or not. This procedure is illustrated in Fig. 7. New samples are drawn near the recognized objects, as shown in Fig. 7(a). After the drawn samples converge, the similarity between the newly drawn samples and the current robot samples are calculated using Eq. (1). Using Eq. (1) and Eq. (2), the pose reliability of object 1 is updated in Fig. 7(b). The pose reliability of object 1 increases up to 0.9. The method which detects the environmental changes and updates the map is explained below in detail.

The pose of object 2 was changed, as shown in Fig. 7(c), and the new robot samples, NR_{sample}, are drawn near the original pose of object 2. As shown in Fig. 7(d), the similarity between NR_{sample} and R_{sample} becomes low, and thus the pose reliability of object 2 decreases due to this low similarity. Since the pose reliability of object 2 is lower than 0.1, NR_{sample} is drawn near the actual pose of object 2, as shown in Fig. 7(e). The actual pose of object 2 can be obtained with the global pose of the robot and the object information from the stereo camera (e.g., the relative range and angle to the object). Then the pose reliability of object 2 is evaluated using the similarity between NR_{sample} and R_{sample}, as shown in Fig. 7(f). If the pose reliability of the newly registered pose of object 2 is greater than 0.5, the new pose of object 2 is registered in the database and the original pose is discarded from the visual feature map.
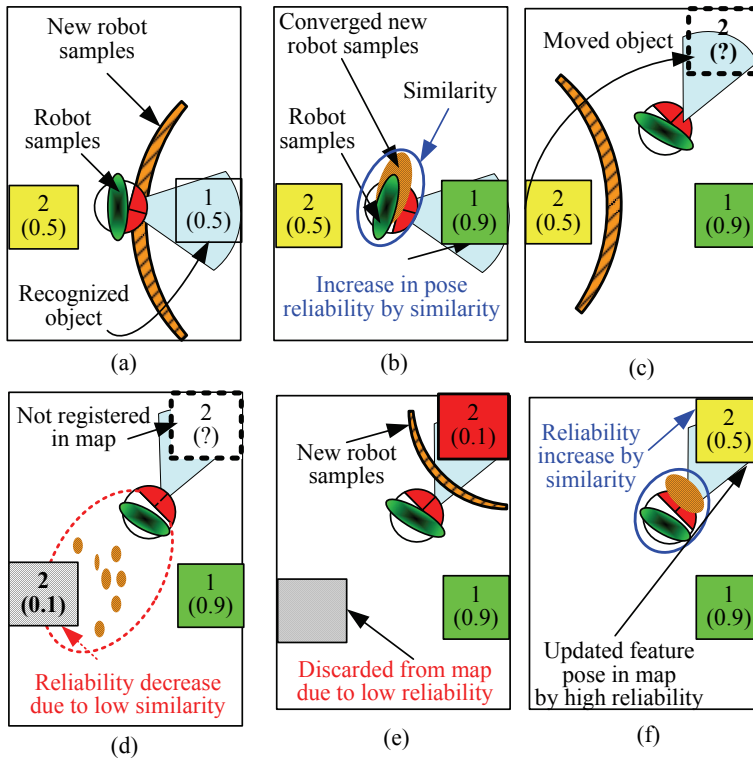
Fig. 7 Procedure of intelligent update of visual map.

4. Experimental results

Experiments were performed using a robot equipped with an IR scanner (Hokuyo PBS-03JN) and a stereo camera (Videredesign STH-MDI-C). As shown in Fig. 8(a), the experimental environment was 9m x 7m. Figure 8(b) shows the visual feature which will be moved to other places during navigation.

(a)   (b)

Fig. 8. (a) Experimental environment, and (b) typical visual feature.

4.1 Pose uncertainty due to environmental changes

(a)   (b)

(c)   (d)
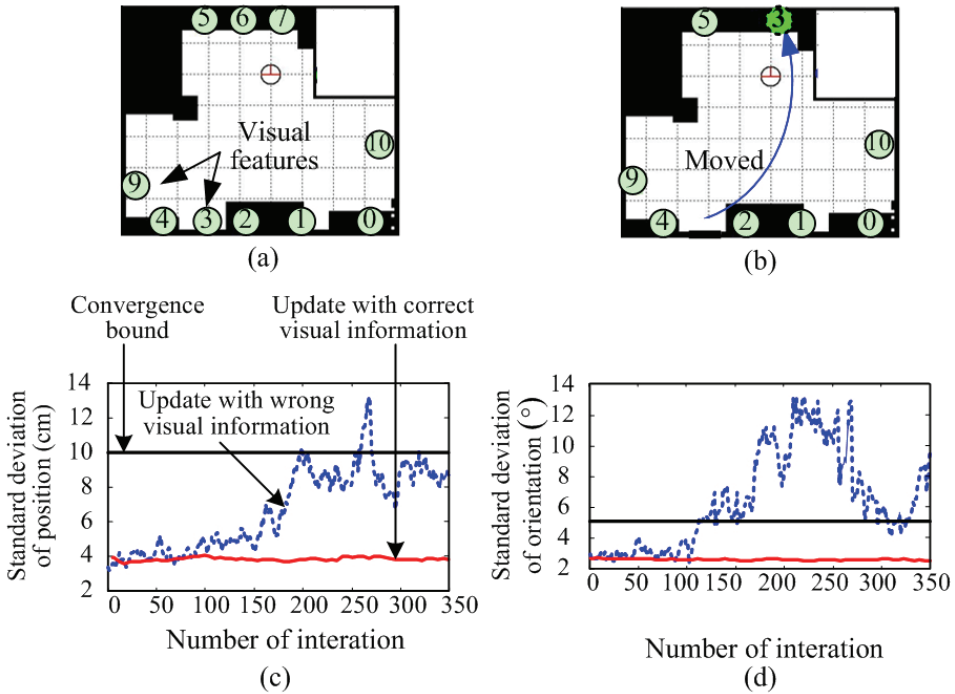
Fig. 9. Localization performance according to environmental changes; (a) experimental environment, (b) changed environment, (c) effect of changed environment on position uncertainty, and (d) effect of changed environment on orientation uncertainty.

These experiments were performed to find out the influence of the environmental change on the uncertainty of the estimated robot pose (i.e., position and orientation). No environmental change was made in Fig. 9(a), whereas the environment was changed in Fig. 9(b). In Fig. 9(c) and (d), the solid red line shows the uncertainty of the estimated pose when the map coincides with the environment. On the other hand, the dotted (blue) line indicates the pose uncertainty under the wrong visual information which means the changed position of object 3 is not updated in the map. As expected, the uncertainty of the estimated pose increases when the environmental changes are not reflected on the visual map.

4.2 Map update according to environmental changes
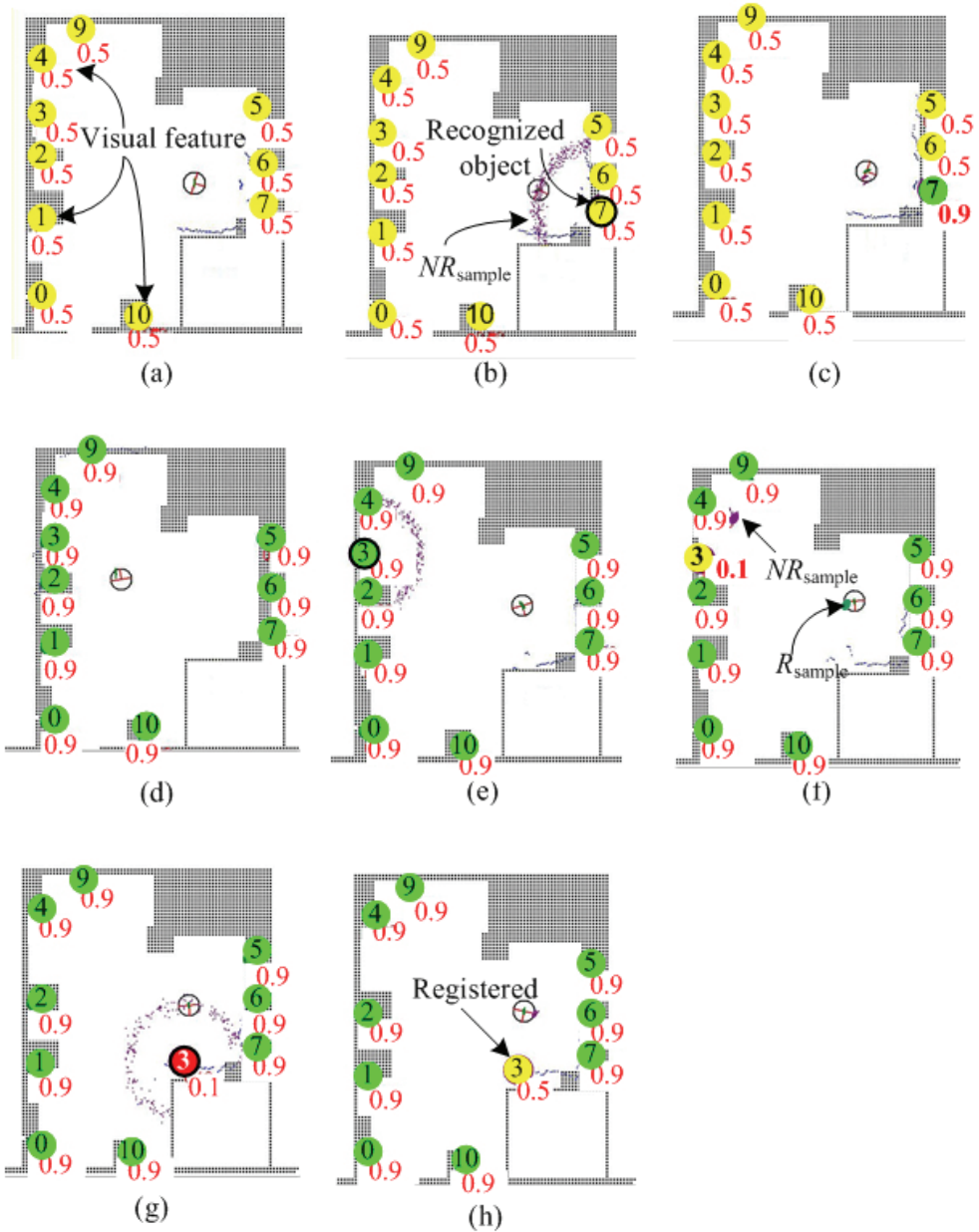
Fig. 10. Experimental results; (a), (b), (c) and (d) are procedure of increasing reliability of pose, (e),(f),(g) and (h) are procedure of intelligent update of visual map.

This experiment was performed to verify that the robot can update the visual map intelligently when the environment was changed by humans. In this experiment, the pose of object 3 registered in the visual map is changed. During navigation, the pose reliabilities of all visual features are initialized to 0.5, as shown in Fig. 10(a). In Fig. 10(b), the robot draws the new random robot samples NR_{sample} around object 7 which was just recognized. In Fig. 10(c), the pose reliability of object 7 increases to 0.9 due to the high similarity between NR_{sample} and R_{sample}, which means object 7 has a high pose reliability. All visual features have a high pose reliability through the above evaluation, as shown in Fig. 10(d). Object 3 is then moved to the place between object 7 and object 10. Fig. 10(e) shows that a robot draws NR_{sample} near the original pose of object 3, when it recognizes object 3 at the new pose. In Fig. 10(f), the pose reliability of object 3 of the visual map decreases due to the low similarity between NR_{sample} and R_{sample}. The robot deletes object 3 from the visual map if its pose reliability is below 0.1. Then NR_{sample} are drawn around the new position of object 3 and calculate the similarity between NR_{sample} and R_{sample}. The pose of the moved object is updated to the visual map if its pose reliability is greater than 0.5. Figure 10(h) shows the updated visual map. The capability of the robot which detect environmental changes and update the visual map intelligently can be verified through the above experiments.

5. Conclusions

In this paper, a probabilistic method which detects environmental changes and updates a map in dynamic environments was proposed. From this research, the following conclusions are drawn.

1. The differences between the environmental map and the real environment can be decreased through intelligent update of a visual map. It improves performance of localization and thus autonomous navigation.
2. The robot operator does not have to stop tasks of the robot because the robot autonomously reflects the environmental changes in the constructed map. In this sense, the proposed method can make a robot operate semi-permanently in dynamic environments.

6. References

D. Fox.; W. Burgard.; F. Dellaert. & S. Thrun. (2001) Robust Mote Carlo localization for mobile robots, *Int'l Journal of Artificia Intelligence,* Vol. 128, No. 1-2, May 2001, page numbers 99-141, ISSN: 0004-3702.

D. Fox.; W. Burgard.; F. Dellaert. & S. Thrun. (1999) Monte Carlo localization: Efficient position estimation for mobile robots, *Proceedings of Int'l Conf. on Artificial Intelligence*, pp. 343-349.

D. Lowe. (2004) Distinctive image features from scale invariant keypoints, *Int'l Journal of Computer Vision*, vol. 60, no 2, page numbers. 91-110, ISSN: 0920-5691.

S. Se.; D. Lowe. & J. Little. (2002) Mobile robot localization and mapping with uncertainty using scale invariant visual landmarks, *Int'l Journal of Robotics Research*, vol. 21, no.8, page numbers. 735–758, ISSN: 0278-3649.

D. Lowe. & S. Se. (2005) Vision-Based global localization and mapping for mobile robots, *Int'l Journal of Transactions on Robotics,* vol. 21, page numbers. 217-226, ISSN: 1552-3098.

C. Kwok.; D. Fox. & M. Meila. (2004) Real-time particle filters, Advances in Neural Information Processing Systems, vol. 92, no. 3.

A. Elfes. (1989) Using Occupancy Grids for Mobile Robot Perception and Navigation, *IEEE computer Archive* vol.22, no. 6, page numbers.46-57.

B.-D. Yim.; Y.-J. Lee.; J.-B. Song. & W. Chung (2007), Mobile Robot Localization Using Fusion of Object Recognition and Range Information, *Proceedings of IEEE Int. Conf. on Robotics and Automation.*

Urbano, an Interactive Mobile Tour-Guide Robot

Diego Rodriguez-Losada, Fernando Matia, Ramon Galan,
Miguel Hernando, Juan Manuel Montero and Juan Manuel Lucas
Universidad Politecnica de Madrid
Spain

1. Introduction

Autonomous service robot applications can be divided in two main groups: outdoor and field robots, and indoor robots. Autonomous lawnmowers, de-mining and search and rescue robots, mars rovers, automated cargo, unmanned aerial and underwater vehicles, are some applications of field robotics. The term indoor robotics usually applies to autonomous mobile robots that move in a typical populated indoor environment. Robotic vacuum cleaners, entertainment and companion robots or security and surveillance applications are also some examples of successful indoor robot applications.

Probably, one of the first real world applications of indoor service robots has been that of mobile robots serving as tour guides in museums or exhibitions. Such one is an extremely interesting application for researchers because allows them to advance in knowledge fields as autonomous navigation in dynamic environments, human robot interaction, indoor environment modelling with simultaneous localization and map building, etc., while also serving as a showcase for attracting the general public as well as possible investors.

We have developed our own interactive mobile robot called *Urbano*, especially designed to be a tour guide in exhibitions. This chapter describes the *Urbano* robot system, its hardware, software and the experiences we have obtained through its development and use until its actual mature stage. This chapter doesn't pretend to be an exhaustive technical description of algorithms, mathematical or implementation details, but just an overview of the system. The interested reader will be referred to more specific bibliography for these details.

The rest of the chapter is structured as follows: This section presents the related work, other existing systems, as well as our motivation to develop our own robot. Section 2 presents an overview of *Urbano*, the description of its hardware and also the software components in which the robot control is structured. These components are afterwards described in subsequent sections: Section 3 describes the feature based mapping and navigation subsystem, while the interaction capabilities including our own proprietary voice recognition and synthesis engine will be described in section 4. Section 5 briefly describes the web based remote visit that *Urbano* is also able to perform. The integration of all these components is managed through a programmable kernel that allows a high level management of all modules, described in section 6. The chapter ends with the presentation of some successful real deployments of *Urbano* in section 7, and our conclusions in section 8.

1.1 State of the art

As previously stated, *Urbano* is not the first mobile tour-guide robot. There have been many others that have served as a reference for the mobile robots research community.

Probably, the first one was Rhino (Burgard et al., 1999) robot from Bonn University, followed by Minerva (Thrun et al., 1999) from Pittsburgh, CMU and Bonn Universities. These robots meant important advances in mobile robot mapping, localization and reactive control, using commercial robot platforms as a hardware base for developments. Another approach was Sage (Nourbakhsh et al., 1999) robot that focused on a more commercial vision that was later accomplished by Mobot Inc. Sage robot uses artificial coloured landmarks in the environment to achieve a robust navigation. These robots were later followed by other derivative works as the robotic assistants for the elderly called Flo and Pearl (Montemerlo et al., 2002).

These robots were afterwards followed by many others from different universities and research centers around the globe, as Albert from Freiburg University (Germany), Lefkos from Forth institute (Greece), Tito from Valladolid University-Cartif (Spain), Carl from Aveiro University (Portugal), just to cite some examples. All of them were interactive mobile robots to serve as tour guides.

Recently, several other robotic tour guides have been developed and commercialized by some companies as the BlueBotics RoboX. Eleven RoboX were involved during 5 months in the Robotics exhibition at the International Expo 02. Industry giants have also built their own robotic guides, as Toyota TPR-Robina operating in their company headquarters. Likewise, Fujitsu developed the Enon robot that served as a guide in the Kyotaro Nishimura museum.

1.2 Previous work

Our initial trials with interactive robots were carried out with Blacky (Fig. 1), a robot for tour-guiding, tele-visit and entertainment. We developed navigation algorithms focussed on indoor, populated, complex and low structured environments.

Fig. 1. Blacky robot in a fair

Blacky was a MRV4 mobile platform from Denning Branch, Inc., with a ring of sonars, a three wheeled synchro-drive system, a radio link for wireless Ethernet connection, a

horizontal rotating laser called *LaserNav* able to identify up to 32 different bar coded passive landmarks, and loudspeakers for voice synthesis.

The robot implemented reactive behaviours such as follow corridor, go to point, escape from minimum, border by the right or the left or intelligent escape, as well as tasks such as walk along this corridor or take this corridor in this direction.

As the robot moved autonomously in a populated public environment making oral presentations and guided tours, interaction with people was an important point. Pre-defined sentences were used for *greetings, welcome, self presentations* or *asking for free way*. A very simple web server was also developed to allow remote users to remotely operate the robot.

Blacky worked in long-term experiments and was tested in exhibition-like contests, were the exhibition organizers point of view was also taken into account. Lessons learnt from that experience conditioned the posterior research of our group. Further details about Blacky robot can be found in (Rodriguez-Losada et al., 2002).

1.3 Motivation

Our main research line is the development of autonomous navigation algorithms for mobile robots, especially focusing on the Simultaneous Localization and Map Building Problem (Rodriguez-Losada et al. 2006a; 2006b; 2007; Pedraza et al., 2007). This research line was partly motivated because in the setup of Blacky a time consuming manual installation and measurement of landmarks had to be done. Nevertheless, it is also true that the main goal of *Urbano* is more than having a platform to evaluate our navigation algorithms. We also wanted to have a platform that could serve to present our advances to the general public, to help us to get funding for our projects, and very importantly, to attract people and students to get involved in research programmes in our group. We think that we have succeeded in these goals: it has helped to present our research and publish our results, we have obtained increasing marks and funding from Spanish Government research programmes and the number of people in our group has also increased. It could be said that *Urbano* is the spirit of our group.

2. System overview

This section presents the description of *Urbano* hardware, both the commercial base but also our own developments: a mechatronic face and a robotic arm for gestures. Also, the general structure of software modules is presented. Later sections will present details about these software modules.

2.1 Urbano hardware

Urbano (Fig. 2) robot is a B21r platform from iRobot, equipped with a four wheeled synchro-drive locomotion system, a SICK LMS200 laser scanner mounted horizontally in the top used for navigation and SLAM, and a mechatronic face and a robotic arm used to express emotions as happiness, sadness, surprise or anger.

The robot is also equipped with two sonar rings and one infrared ring, which allows detecting obstacles at different heights that can be used for obstacle avoidance and safety. The platform has also two onboard PCs and one touch screen. These PCs are mainly

dedicated to access the hardware, low-level control of the base, interfacing the laser rangefinder, controlling the arm and face, and performing voice synthesis and recognition.

Fig. 2. *Urbano*, our interactive mobile robot

The hardware architecture is completed with two off-board PCs as shown in Figure 3. Communication with them is implemented via wireless Ethernet. One of the external computers is dedicated to the system kernel, which handles coordination of all modules. This system kernel can, however, run also in one onboard PC, leaving the external one just as a simple monitoring and supervision tool that could be even switched off.
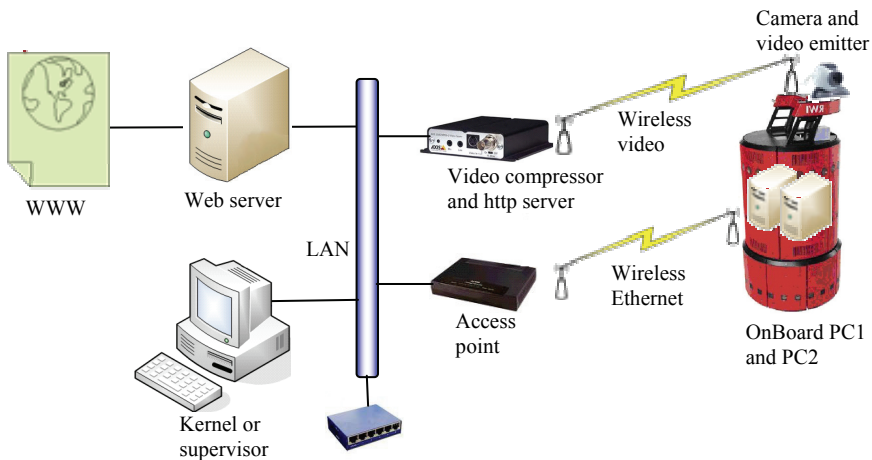
Fig. 3. System hardware architecture

The second PC is fully dedicated to the web server, which communicates with the kernel via Ethernet and TCP-IP protocol. The web server acts as interface between the robot and the

world, allowing remote users connection to operate the robot and visualize dynamic information coming from its sensors. We can also keep the web server at our laboratory, while the rest of the equipment is physically present at the exhibition site. The video stream is served trough a dedicated http video compressor and server, which output is just redirected by the web server.

2.2 Robotic face

The robot was thought to interact with people. In fact, all the environments in which the navigation tests were done were plenty of people. In order to achieve a satisfactory interaction with the public, it was strictly necessary the design and implementation of a robotic face. People find in it an attraction point to look when talking to the robot. At the same time, the face allows the robot, in combination with the voice, to express basic emotions.

The first step was the face design, analyzing its ability to express emotions, and taking into account that the design should be simple enough to be build by ourselves. Other existing robots faces were also analyzed, like Kissmet (developed by MIT) which was too complex for our purposes. Albert's face, a robot from University of Freiburg was finally our referent. In an initial version we developed a face with 5 degrees of freedom, 2 to control the mouth, one for each eyebrow and another one for closing both eyes. The actuators were model servomotors S3003 from Futaba. In this version the eyes were completely static, the mouth could not be opened and both eyelids had to be moved together. Nevertheless, as shown in Figure 4, it was perfectly able to show basic emotions.
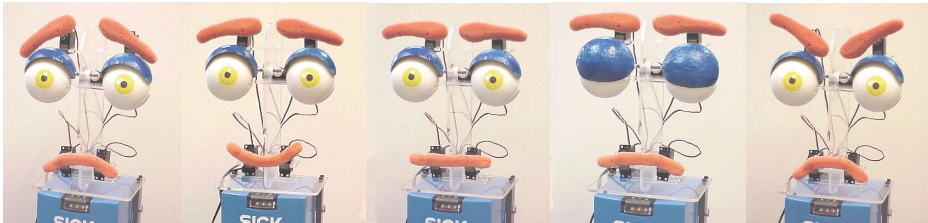
Fig. 4. *Urbano* face initial version with 5 degrees of freedom, showing happiness, sadness, being neutral, angry or asleep.

A simple board, the Mini SSC II, allows controlling the servos in an easy manner through a serial port, just by sending chains of three characters. This controller has low consumption and low dimensions.

This face is the one we have currently mounted in *Urbano*. Nevertheless, we have recently developed a new one (Fig. 5) with increased interaction capabilities. We have incremented the count of servos up to eleven, including four of them to control the mouth that can be opened and closed, simulating speech in a much more realistic way. Four servos are used to move both eyes left and right independently (cross-eyed possible) and independent eyelid closing for winking. Another servo moves both eyes together up and down. The eyebrows are controlled by two more servos. Jamara Mini-blue and Micro-blue servos are used in this version.
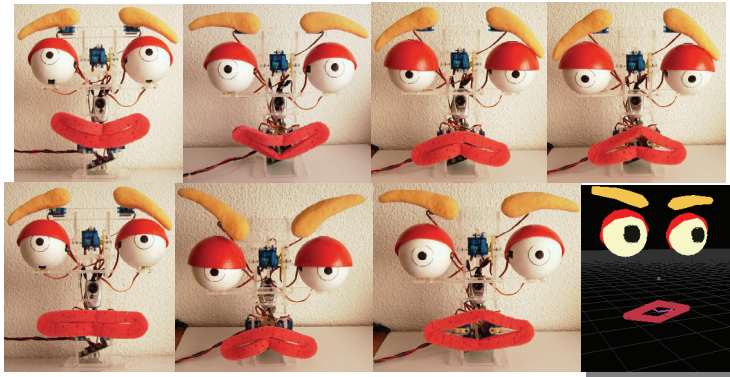
Fig. 5. *Urbano* face with 11 degrees of freedom, showing happiness, sadness, being neutral, angry, and moving the mouth.

2.3 Wired robotic arm

Since the beginning of *Urbano* development the need of a gestural interaction system between human and robot was considered. While showing the programmed tour, without gestural communication, the attention is easily missed due to the hieratic interaction between the robot and the visitors. With the inclusion of a robotic arm in the system, a more direct interaction with the environment is achieved. Moreover, it increases the ability to attract attention and helps to give emphasis and to include emotional aspects to the speech. Clearly, the use of sign language makes the interaction more natural, friendly and attractive. All this implies that the robotic arm must have a set of specific qualities and characteristics. Because it must reflect the common gestures in a speech, the structure, proportion and the dimensions of the robotic arm should be similar to the human arm. The arm movements should have similar dynamics, which requires that the movements should be stiff-less, quick and as natural as possible. As a consequence, the absolute accuracy and repeatability are not significant within a range, since the relative motions are more important in order to gesticulate than the absolute positions.

Urbano is conceived as a tour guide robot. Therefore it would be moving close to people and in a non structured environment. Safety issues have special relevance both for the humans and the robotic system. Due to its application, the system even has to allow contact with people without risk to them or to the robot hardware. However, as it has been specified before, the robot arm must move with agility and fast movements become more dangerous as the mass of the arm increases. As a consequence, a major requirement of the robot arm is that it has to be as light as possible. Reducing the inertia simplifies the actuator complexity and reduces the safety problems. Moreover, the actuation system should be somewhat reversible, so if the arm is moved manually it has to allow such movement or adopt a compliant behaviour.

The adopted solution in order to accomplish these requirements is to extract the drives from the arm and place them in the base of the robot, which is the equivalent to the shoulder blade. Placing the actuators in such way entails the problem of transmitting the mechanical power to the different joints and in particular to the elbow. Figure 6 shows the robot arm

kinematics. It has four degrees of freedom (dof), three on the shoulder articulation and the forth in the elbow.
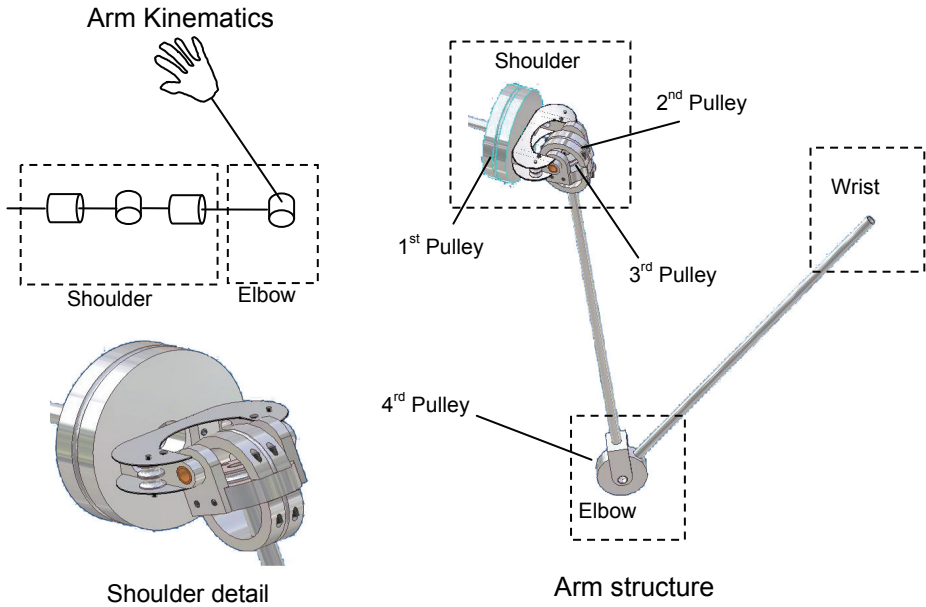
Fig. 6. Kinematics and Joint drive pulleys of the robotic arm.
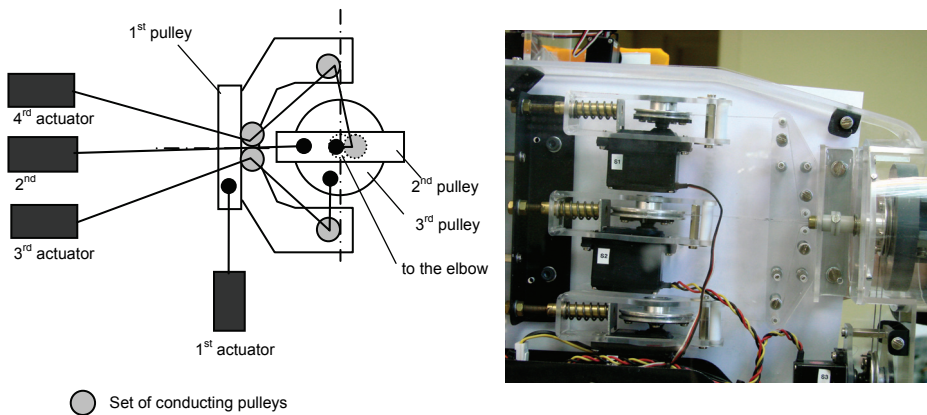
Fig. 7. Left) Schematic representation of the drive cables that are going through the shoulder. Right) Picture of the robot shoulder blade.

A cable based transmission system has been chosen as solution rather than gears. Using a gear based transmission system would be more expensive and complex from the mechanical and control point of view. In such systems a fine and complex control loop have to be used

in order to cancel the coupling among the different joints, due to the effect that a joint movement has on the subsequent articulations.

Each joint has a pulley where two wires are attached. Those cables turn the joint in opposite directions, and therefore, the length of wire that is winded must be the same that is released. Wires that run through the articulations have to be not affected by the joint movements. These can be accomplished conducting the cables through the axis of the previous joints. An added difficulty is the emergence of a friction that increases exponentially with the number of turns that the cable performs. Therefore, all the turns are made through a set of small polyester pulleys because of its low friction coefficient with nylon. Figure 7 (left) shows an scheme of the different drive cables that are going through the shoulder. The two drive cables for the elbow articulation have to go trough the three previous joint axes. Therefore it is conducted by three different sets of pulleys as is represented in the figure.

Finally, the winding and releasing of the wires is done by four servo based drive units. The Figure 7 (right) shows these units placed on the shoulder blade of the robot. In order to keep the cables tensed each unit tighten them by an adjustable spring attached to the servo and the winding pulley. Given that the absolute accuracy is not a major requirement; the position feedback is done in the drive itself, not in the joint. Therefore there is no electronic components on the arm neither signal or power wires.
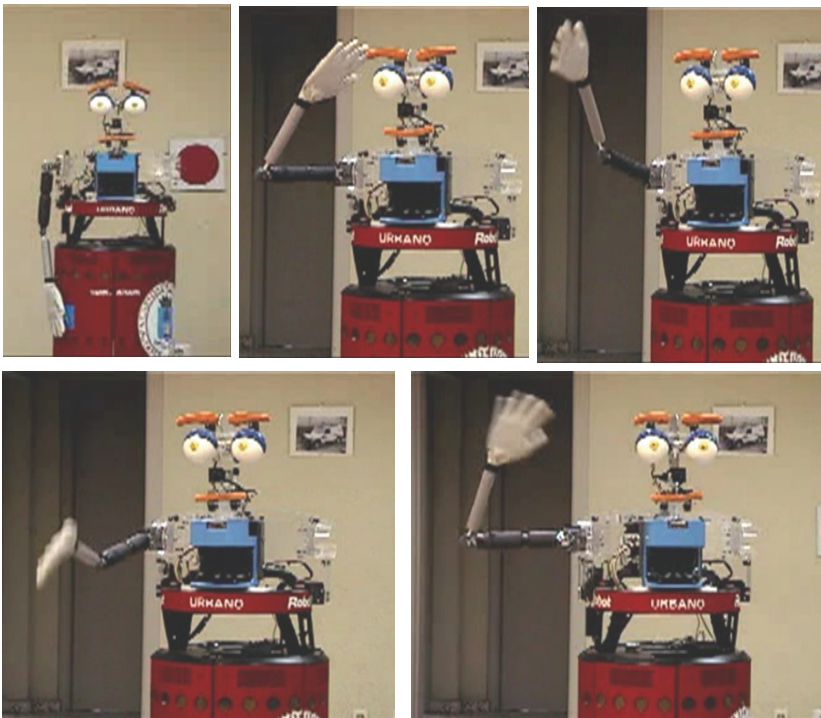
Fig. 8. Several video frames captured during a speech of *Urbano*.

The actuator units are controlled through a microcontroller based control board, that is linked through a serial RS232 connection to the onboard robot computer. This computer has

a server process that is responsible of the execution of the different commands received by a TCP/IP socket.

Figure 8 shows several video frames captured during a speech of *Urbano*. Predefined sequences of joint movements are able to express messages like this, goodbye, at your commands, everybody, etc.

2.4 Software components overview

The software is structured in several executable modules (Fig. 9) to allow a decoupled development by several teams of programmers, and they are connected via TCP/IP. Most of these executables are conceived as servers or service providers, as the face control, the arm control, the navigation system, voice synthesis and recognition, and the web server. The client-server paradigm is used, being the only client a central module that we call the *Urbano* kernel. This kernel is the responsible of managing the whole system, issuing requests to the services based on the input data defined by the exhibition database and the established robot behaviour, that is defined in a high-level programming language that will be described later.
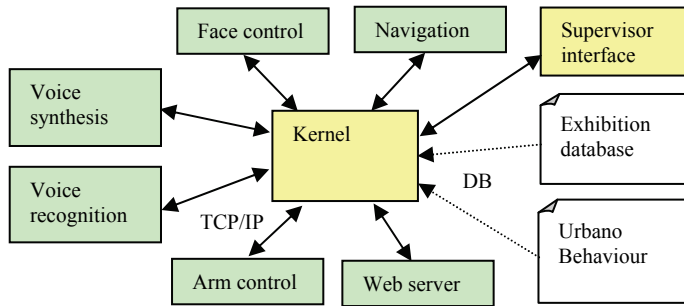
Fig 9. *Urbano* software control modules overview.

The supervisor interface acts as a client of the kernel, that reflects all necessary information to the user. Although this is the common use, the supervisor is also able to directly connect the server modules to check low level functionality.

3. Feature based mapping and navigation

We realized from our experiences with Blacky that automatic map building was required for an easy deployment of *Urbano* in new environments. The analysis of the exhibitions and the setup procedure indicated that a feature based approach could probably achieve better results and more robustness could be obtained both in the mapping procedure and the later localization in the built map. We noted that the environments were plenty of representative geometric entities, mainly straight walls, but they were also crowded because the setup procedure had to be performed while the exhibitions were open to the public. The most extended approach for feature based SLAM is the EKF algorithm, but this filter is difficult to apply when the features of the environment cannot be completely observed, e.g. when a wall is observed partially because of occlusions. Most of our recent research has been

focused in the feature based SLAM problem under an EKF approach. We developed our own version of the SPMap (Castellanos et al., 1999) algorithm, which is probably the best existing solution to handle the problem of partial observations. Our algorithm (Rodriguez-Losada et al., 2006a) efficiently handles the edges information, which is extremely important when navigating in corridors.

We soon realized that the SLAM-EKF filter was quite optimistic due to the intrinsic inconsistency (Rodriguez-Losada et al., 2007) that arises due to EKF linearizations. We proposed the use of perfectly known shape constraints (parallelism, orthogonality, co-linearity) between segments of a map to reduce the angular uncertainty of the robot that is the main source of linearizations (Rodriguez-Losada et al., 2006a; Rodriguez-Losada et al., 2007). With this solution, medium size maps with loops can be built in real time, which is more than enough for all the environments were *Urbano* has been deployed. Nevertheless, we also developed an algorithm based on the use of local maps (Rodriguez-Losada et al., 20076b), that allows multirobot mapping of large environments in real time.

 The setup procedure is usually performed with a laptop connected to the robot base, used to manually drive the robot around the environment, while the SLAM-EFK algorithm runs, building the map in real time that is showed to the operator. Nevertheless, the system can also serve for remote exploration and autonomous return, in a fashion similar to (Newman et al., 2002) as we showed in (Rodriguez-Losada et al., 2007). Once the map is built, it is downloaded to the robot, so it can automatically start a simple pose tracking algorithm. This continuous localization or pose tracking is just a simplified version of the SLAM-EKF algorithm, with the map of the environment considered as perfectly known and static. Thus the estimation is only done over the robot position and orientation, resulting in a fast and robust algorithm.

Path planning in a feature based map is not recommended, as not every obstacle is represented in the map. Grid maps could be used, but the problem of obstacles at different heights still remains. Consider the existence of tables, stairs, fences, etc, which are basically undetectable by the robot perception system. The only way to achieve a safe navigation is to constraint the robot to certain areas supervised by the installer of the system. We used a graph based approach. While exploring, *Urbano* automatically builds a graph of the environment deploying nodes in the virtual map, that are connected by branches only when revisiting them is showed to be possible. Path planning is computed in this graph with an A-star heuristic, giving as a result a sequence of ordered nodes or waypoints to the final goal. The reactive controller moves the robot to the next waypoint with a simple regulator, but also avoiding obstacles with a deviation from the direction provided by the regulator. Safety is obtained by permitting only a limited distance to the actual branch. Usually, the graph computed by the robot is not enough to allow guided tours, so a graphical user interface allows the installer to add, delete, edit, move nodes and branches, as well as assigning tags to places that can be used to identify particular exhibits *Urbano* can show.

To allow the supervision of the map building procedure and the navigation performance, a GUI application has been developed. The SLAM and navigation kernel has been implemented in portable C++ for efficiency, and the interface has been developed (Figure 10) in a multidocument-view MFC application, using OpenGL for 3D rendering. This application has been proved to be of critical importance for an easy deployment of *Urbano*.
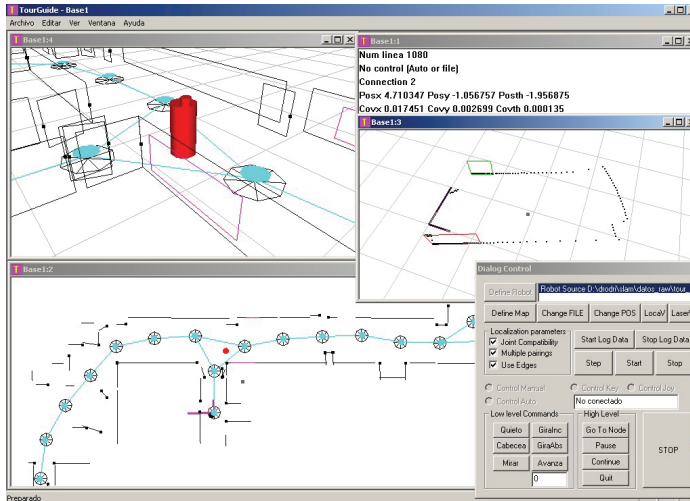
Fig 10. Map building and navigation GUI tool.

This navigation software has been also used in a different robot: the robotic smartwalker Guido of Haptica Ltd. (Dublin, Ireland), an assistive walker to support and guide the frail blinded elderly. The feature based mapping and navigation approach proved to be an improvement in Guido control, as shown in (Lacey & Rodriguez-Losada, 2008).

4. Interactive subsystem

4.1 Interaction capabilities
As described above, *Urbano* possesses several features that could be used for interacting with the people. If we conceive *Urbano* as a system, the interaction capabilities could be classified in inputs and outputs:

Outputs:
-    The robotic arm is only able to perform gestures, but not force feedback is allowed. Thus the arm is not able to sense the environment or feel any contact. Although this would be a very interesting feature, it would also be quite complex and expensive.
-    The face is able to show basic emotions, to move the mouth while speaking and to direct the eyes to any point.
-    The robot base itself is an element that can interact with the people. It can move faster or slower, to look at the closest person, it can perform basic movements as steps, nodding, quick rotations, that can be used for complementing interaction.
-    The voice synthesis is the most powerful and versatile output, being able to transmit any kind of information but also to change voice parameters (volume, speed, tone) and speak with different emotional pronunciation, but on the other hand it also requires a more complex handling.

Inputs:
-    Voice recognition is the main input for interaction, despite its complexity. Both the difficulty of understanding the speaker in a noisy environment like an exhibition, and the management of textual information, makes impossible a general dialog manager. Nevertheless, it is still quit a powerful tool when the dialog is managed by the robot.

- The robot navigation system provides useful information, about close obstacles and people blocking its path, that can be easily used for initiating interaction.
- The face webcam is used for automatic face tracking (Figure 11) with the robot eyes, with a simple threshold of the image in the hue space, plus a geometric analysis of the binarized image.
- Some other information can be used as modifiers of the interaction with the user, as the battery level that can be associated to fatigue, or the time employed to perform a task that can produce stress to *Urbano*
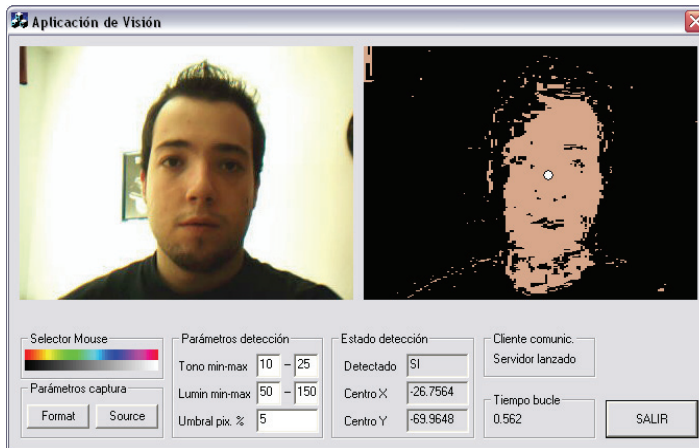
Fig. 11. Face detection for tracking with the robotic eyes.

4.2 Proprietary voice synthesis and recognition

In order to provide *Urbano* with an appropriate human-robot interface, a speech synthesizer and a speech recognizer must be designed and developed. The proposed interface would allow a natural but reliable speech dialogue between visitors and the robotic guide. Although speech technology has progressively become a mature engineering area with several commercially available products, the development of robust applications in real-life ever-changing environments is still a topic of comprehensive research.

4.2.1 Speech recognition and understanding

Commercial speech recognition products are mainly oriented to classical speaker-dependent dictation products developed by Dragon Systems and IBM, telephone-based systems (the market is currently dominated by Nuance) or restricted-domain applications (Philips has developed several products mainly for hospitals). These systems come with limitations, as they cannot be used in open-access museums or trade-fairs without a significant reduction in their performance, because human spontaneity and limited linguistic coverage minimize the potential benefits of commercial products (Fernandez et al, 2006).

In addition to this, available systems do not provide automatic systems for speech understanding, but just speech recognition. The mobile robot needs procedures to extract concepts and values from the text that outputs the recogniser, being able to cope with recognition errors and ambiguities.
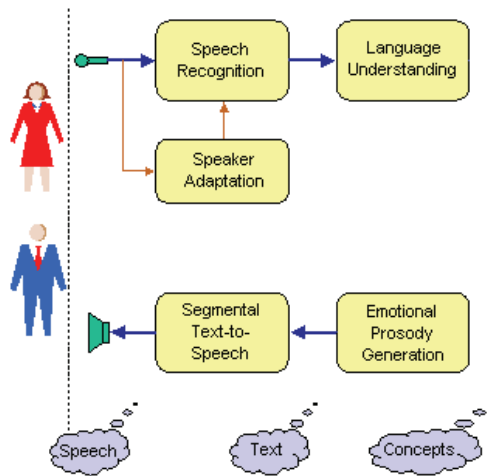
Fig. 12. Speech processing architecture in *Urbano*

Finally, commercial products generally do not provide a confidence measure on the result of the recognition process, a measure that allows a robust behavior on noisy working conditions. For example, when there are many children surrounding the robot.

Considering all these limitations, we have developed speech recognition software customized for use with a robot, as have we developed adapted modules for an air-traffic control domain or for controlling a HIFI system (Cordoba et al, 2006).

The standard speech recognition technique (Hidden Markov Models) is based on stochastic modeling of each phoneme in its context and trainable language model (bigram) that uses the probability of two words to be consecutively uttered in the specific application domain. We have trained our system with a 4000-speaker speech database to achieve speaker-independent models. In a 500-word command-and-control task, recogniser's word accuracy is greater than 95% (word accuracy takes into account speech recognition errors due to word substitutions, insertions and deletions).

Although any microphone can be used successfully, close-talk head-microphones are the best choice, due to immunity to ambient noise (which can be high in children-oriented museums, for example).

As for improving the performance of the recognition for certain special speakers, a speaker-adaptation module has been included. This module significantly improves the general models trained with 4000 speakers). Error reduction can be as high as 20%, especially for female speakers.

As speech recognition is only the first component of the speech processing, we have also developed an automatic speech understanding module. In order to adapt the system to a new exhibition or trade fair, we must provide the system with a set of samples that should be recognized by the robot, and the set of concepts and values involved in each sentence. The system automatically learns a set of understanding by induction. The rules that are learnt can convert the recognized speech into the suitable sequence of concepts and values, without the need of a human expert. Nevertheless, if the set of examples is reduced, new rules can be manually added.

4.2.2 Emotional speech synthesis

While in recent years many speech synthesizers have managed to achieve a high degree of intelligibility, one important problem remains, which is the inability of simulating the variability in human speech conveyed by factors such as the emotional state of the speaker.

The approach of this work has been based on formant synthesis, including four primary emotions, namely happiness, sadness, anger and surprise as well as a neutral state. Although this approach produces less natural speech when compared to other approaches such as concatenate synthesis, it provides a high degree of flexibility and control over acoustic parameters.

To improve the results of previous approaches, we have optimized the prosodic models that mimic the rhythm and intonation of the reference professional speaker we have recorded. Each emotion is prosodically modeled as a deviation from the neutral way of speaking. To simulate sadness we have included an artificial tremor that, although not used by the professional actor, has significantly increased the identifiably of this emotional synthetic speech.

Our actor has simulated cold anger (instead of hot anger), which is a very-controlled but menacing emotion. To simulate this kind of anger, he created a special noise during the articulation of most of the sounds, without modifying the quite neutral prosody. This non-prosodic anger is very difficult to be completely simulated on formant synthesis. This time, the significant improvement was obtained by combining an artificial articulation noise with an intensity pattern that progressively simulates hot anger.

Finally, to improve happiness, the most difficult emotion, we have increased the amount of high frequency in synthetic speech, to provide it with richer sound that is easily associated with a happy state.

The simulation of emotions in synthetic speech was tested by a group of 24 non-trained listeners. The confusion matrix obtained for the 25 sentences that composed the test is showed in next table.

Simulated Emotion	Identified emotion (%)					
	Happiness	**Cold Anger**	**Surprise**	**Sadness**	**Neutral**	**Other**
Happiness	**53.9**	9.6	20.9	0.0	7.8	7.8
Cold Anger	7.0	**70.4**	14.8	2.6	3.5	1.7
Surprise	17.4	2.6	**79.1**	0.0	0.0	0.9
Sadness	0.0	1.7	0.0	**87.0**	10.4	0.9
Neutral	1.7	3.5	2.6	7.8	**83.5**	0.9

Table 1. Confusion matrix from emotion identification experiments on speech synthesis

We can observe that all the emotions present a recognition level above 50%, and for all the emotions with the exception of happiness, this level exceeds 70%. The mean identification rate in the new perceptual test is 75%, in this semantically-neutral short-sentence emotion identification experiment. When compared to previous formant-based results on the Spanish work package in VAESS project, (Montero et al, 2002) the improvement ranges from 65% for anger, 42% for neutral, 15% for happiness, to just 5% for sadness.

These results are even better (>65%) for the last 10 sentences that composed the test, in spite of the fact that listeners did not receive information about the identification success or failure they were getting

4.3 Emotional manager

Many investigations in the emotional model area have been done and many others are currently under way. It is quite a new field and it involves many different sciences, for that reason it is not common to find fix structures for studying or developing artificial emotional models. One of the most significant studies is (Picard, 1997). From a pure scientific point of view, emotional models are studied in psychology, neuroscience, biology, etc. Humaine Network of Excellence (http://emotion-research.net) aims to create an investigation community to study emotions in the frame of human-robot interaction.
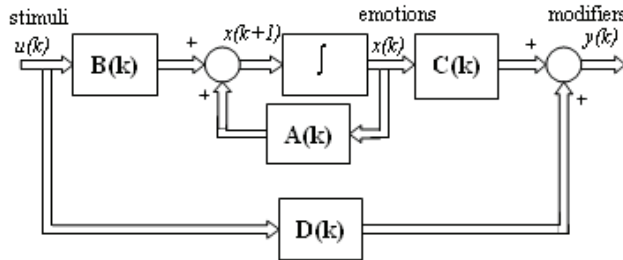
Fig. 13. Emotional state model

In order to reach a nearer approximation to human emotional system, the *Urbano* model makes use of dynamic variables to represent internal emotional state. The model follows the classic diagram showed in Fig. 13, being the system stimuli *u(k)* considered as inputs variables, emotions *x(k)* as state variables and task modifiers *y(k)* as output variables. In the following paragraphs the concepts used to build the emotional model are introduced more accurately.

Trying to define an emotional state in a human being, an emotion and its magnitude are used. For example, the statement "I am very happy" includes qualitative information, the emotion "happy", and quantitative information that is expressed with terms that give an idea about the intensity of the emotion "very".

Based on that, the *emotional state* at the time *k* is defined as the set of considered emotions with their intensity levels. Intensity levels of each emotion change continually, giving dynamics to emotional state. Emotional state tends naturally to a nominal emotional state where a balance of emotion intensities exists. An emotion is an internal variable.

A *system stimulus* is any event that has an influence in the system producing an emotional state change. There are many events that may stimulate the system, the only limitation is the system ability to sense, i.e. sensors, cameras, etc. Robotic stimuli may be internal or external. An example of internal stimuli is the life or energy the robot has, usually considered as the battery state.

Urbano has scheduled tasks; such schedule can be modified because of instantaneous emotional state. All these changes are considered as *system task modifiers*. An example of scheduled task is the tour in a museum, which a guide robot has to direct. Modifiers for this task could be the tour tempo, information to give, jokes used to build a better connection with public, etc.

Following the classic state variable model, four matrices have to be defined: A-matrix *emotional dynamic matrix* represents the model dynamic, the influence of each emotion over

itself and over the other emotions. B-matrix is the *sensitivity matrix*. C-matrix has the information of how emotional state influences modifiers. Let us call this matrix the *emotional behavior matrix*. D-matrix is the *direct action matrix*.

Due to the difficulty of finding an analytic calculation for the matrices coefficients, a set of fuzzy rules is used to obtain each coefficient. The matrices coefficients are function of time k, giving dynamics to the system. Because of that coefficients are calculated for each time k. To define fuzzy rules is a simple task; the information contained in the rules can be obtained from experts in emotions. The use of fuzzy knowledge bases opens the opportunity to a future automatic adjustment, e.g. genetic algorithms.

5. Web based remote visit

One of the project goals was the development of a Web server to allow users to visit remotely an exhibition, navigating through the robot movement and watching through its sensors. The user can be a normal citizen that enjoys connecting from his home, or a business man that connects from his office. This allows saving the displacement costs derived from travelling physically to the exhibition site, especially when the visitor lives or works in another city or country. Three kinds of users are allowed:

- The standard visitor, which can navigate through the web page accessing general information and watching the behaviour of the robot, or ask for an account.
- Privileged visitor, which can operate the robot and interact with the remote site, as well as with other connected users.
- Administrator, which can manage users, creating new accounts and assigning access privileges.

A privileged user can:

- Set a destination goal for the robot (high level command). The navigation system works in autonomous mode.
- Chat with other users.
- Command the robot sending low level commands (move forward or backward, turn). A security system avoids the robot to crash.
- To receive dynamic information of the surroundings of the robot.
- Visualize the robot environment through its camera.
- To receive the audio signal present at the remote site.
- To write down sentences to be synthesized by the robot.
- To select emotions to be expressed by the robot face.

The web server was developed using Jakarta Apache Tomcat 4.0, which includes Java support, over a Linux operating system (Debian 3.0 release1). The programming tools used were those included in Java 2 Platform, Enterprise Edition, J2EE (Java Server Pages -JSP-, JavaBeans, JavaXML), server and applets applications, and every program was written in standard Java 2. The Web pages format is standard HTML 4.0. Figure 14 shows the typical frames displayed during normal operation (map, camera, chat and control windows).

All data is stored in a mySQL data base. Information exchanged with the database is carried out using SQL (Structured Query Language) through queries to a data base server (mySQL Server) resident in the same PC. The development application was the programming environment supplied by Sun MicroSystems, SunOne Studio 4.1 Community Edition.

The web server was deeply tested at INDUMATICA 2004 fair celebrated at UPM. The server worked for 3 days, a total of 16 hours. 63 users registered, being 18 professors, 31 students

and 14 people form outside of the university. The web Server was also successfully proven at the Science Museum Príncipe Felipe of Valencia, and allows carrying out remote tours to our laboratory at UPM.
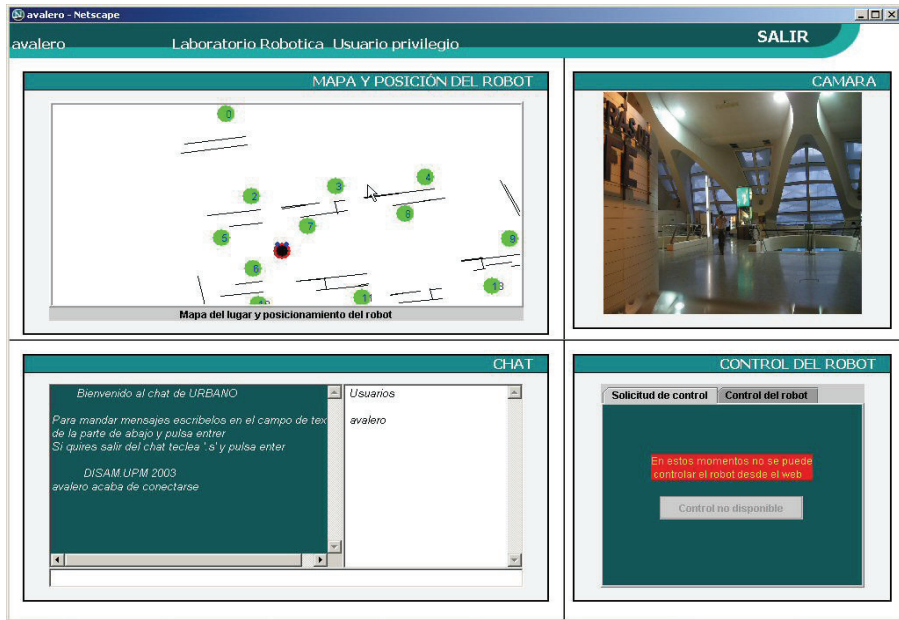
Fig. 14. *Urbano* Web based remote visit

6. Integration of components: Multitask Kernel

Front the point of view of *Urbano*'s software components, it is an agent based architecture. A specific CORBA based mechanism is used as integration glue. Every agent is a server and there is only one client, the *Kernel* module. Each computer has a *Monitor* program that interacts with the Operating system to start, suspend or kill the applications assigned to this machine.

Watchdog supervision mechanism are used to detect blocks in every client and if it is necessary to restart it. Some agents need to save a safe state in order to recovery the whole functionality (robot's recent position).

There are different kinds of information involved in *Urbano*:

- Configuration. All necessary configuration data (IP address, file names, etc.)
- Working data. Each agent can uses specific information usually data files (sequence of movements for the "Hello" action in Arm agent)
- General information. About social, humoristic, sportive information that *Urbano* uses to interact with the public
- Corpus. About the specific domain which *Urbano* works (Museum or fair contents).

A relational database was implemented to support general and corpus information, and specific files for working and configuration data. There is not redundant or shared information. The agents and their function are described in Table 2.

Agent	Function	Computer	Activity
Kernel	Task scheduler, knowledge	OnBoardPC2 (win)	Client, Server
Speech	Voice synthesis	OnBoardPC2 (win)	Server
Listen	Voice recognition	OnBoardPC2 (win)	Server
Face	Face expression control	OnBoardPC1 (linux)	Server
Arm	Arm movements control	OnBoardPC1 (linux)	Server
Navigation	Base movements control	OnBoardPC1 (linux)	Server
Emotional	Emotional model control	OnBoardPC2 (win)	Server
Supervisor	Monitoring of kernel and modules	External PC1 (win)	Client of kernel
Web server	Serve web pages	External PC2 (linux)	Server Http server

Table 2. Agents and functions

Some other programs have been developed for different needs. *Mapper* was designed to elaborate and managing maps and graphs for path planning. *UDE* the *Urbano* development environment is a complex program designed to help the end user in the maintenance and task development, also is a supervisor program of the whole architecture. Figure 15 shows the main window of this program.
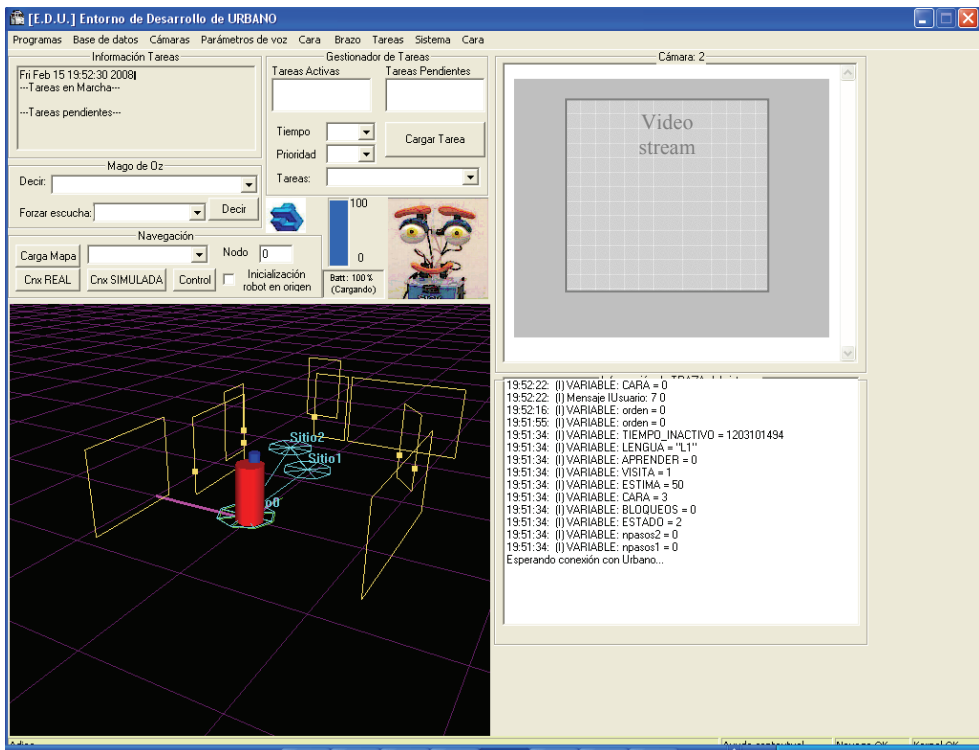
Fig. 15. *Urbano* Development Environment

The Kernel agent is a scheduler that executes *Urbano* tasks. Each task has a starting time and a priority. High priority tasks interrupt lower priority tasks. Tasks are coded by the user in a high-level programming language designed for this purpose. The tasks are compiled with yacc-lex technologies to avoid errors and to simplify the execution by the Kernel.

6.1 High-level programming

High level programming language designed is C-like. Variables can be numerical or string and the first assign defines the type. Expressions and execution control sentences are available in the same syntax that C language.

There are an important set of functions related with database access, string operations, global variables, system, etc. There are also functions to control the robot. The following table 3 shows some of these functions:

Function	Description
listen	Waits for a specific sentence from de voice recognition module
listendb	Waits for a sentence defined in Database
say	Synthesizes a sentence
saydb	Synthesizes a random sentence of a category from the database.
face	Shows a specific expression in the face
arm	Does a set of arm movements that was defined as a expression.
play	Shows a multimedia movie in the Touch-Window
image	Shows an image in the Touch-Window
buttons	Returns the identification of the selected button in the Touch panel
feeling	Does an evaluation of robot emotions
go	Goes to a specific point
where	Returns where the robot is
turn	Turns some degrees
isblock	Returns true if the robot is blocked

Table 3. Control Functions of *Urbano* programming language

The following text shows an example of task. The robot is walking and helloing around the available 14 places in the map. Task starts with an order to go to the next place. While is moving call to another task to verify if the robot is blocked in his path by objects or people and say a random helloing phrase from the database and wait 30 seconds. When the robot is in the next place, put itself in the agenda as a new task with 5 seconds of delay to start and a priority of 20.

```
// Walking !
     destination=where()+1;
     if(destination ==14)  destination = 0;  endif
     go(destination);
     while (where() != destination)
          jump("blocks");
          saydb("Hello");
          sleep(30);
     endwhile
task("walking",5,20);
end
```

In this new task example, the robot wait for a question (recorded in the database), then the *listen* function returns a keyword and a SQL query is performed to obtain from the 'explains' table all records with this keyword. In every record there are a Text, a voice type, an arm movement and a face expression that are used to give the answer.

```
// Questions
say("What is your question?");
Theme=listendb();
pTable=dbsql(format("SELECT * FROM EXPLAINS WHERE KEYWORD = '%s'
ORDER BY ORDEN", Theme));
if (pTable>=0)
      ndatos=dbgetcount(pTable);
      while (ndatos>0)
            arm(dbgetint(pTable,"ARM"));
            setspeakingvoicetype(dbgetint(pTable,"VOICE"));
            face(dbgetint(pTable,"FACE"));
            say(dbgetstr(pTable,"TEXT"));
            dbnext(pTable);
            ndatos=ndatos-1;
      endwhile
      dbclose(pTable);
      gestobrazo("POSICION_CERO");
else
      say("I don't now anything about this theme!");
endif
end
```

6.2 Managing visits

Urbano database has and inventory of objects. Each object is included in several categories, for example a Picasso's picture is a picture, modern art, cubism style, big size, etc. Each object is in a place in the map and the order of visit is important in order to avoid comings and goings. About each object there are different kinds of information: general description, specific for expert, specific for child, components, history, details, anecdotes, etc.

Urbano as tour guide robot must guide to a people group in a museum or fair in a visit. For *Urbano* a visit is defined as a set of categories to explain in limited time for some kind of visitors defined by some topics: Expert, Normal, Child, etc. Some SQL queries to database select the objects and the information about each object to be explained. If there isn't enough time for the exposition of all selected objects, a prune process is executed to reduce the number of explanations of each object (a priority value). This work is previous to the visit and can produce a test of visit, the robot makes the visit and controls the moving time and the explanation time in each object.

During the real visit timing can vary depending on questions or moving time (visitors blocks *Urbano*) if time lacks a prune process is used. Free time can be used by *Urbano* to tell jokes or recent social news.

7. Urbano successful deployments

Urbano robot has been successfully deployed in several environments, and has operated as tour guide in many occasions:

- Lab Tour: guided visit to our laboratory

- Indumatica 2004 (ETSII, Madrid, Spain): industrial trade fair
- Indumatica 2005(ETSII, Madrid, Spain): industrial trade fair
- Fitur 2006 (IFEMA, Madrid, Spain): international fair of tourism
- Principe Felipe Museum (CACSA, Valencia, Spain). Science museum.
- Demonstration at UPM.

A demonstration was performed at our university that started with a teleoperated real time exploration and mapping at rush hour with the environment crowded with students. The installer used the GUI tool to teleoperate the robot with the reference of the map graphical render, and the assistance of the robot reactive control for safety and automatic graph building, path planning and execution for convenience and comfort. In this way, the installer teleoperated the robot while exploring, but *Urbano* could go to any previous explored area fully autonomously, releasing the user from direct control most of the time. The duration of the experiment was 22'15'' with a travelled distance of 134 meters. With an experiment duration similar to the "Explore and return" experiment (Newman et al., 2002) the explored and mapped (in real time) environment is much bigger (Figure 16).
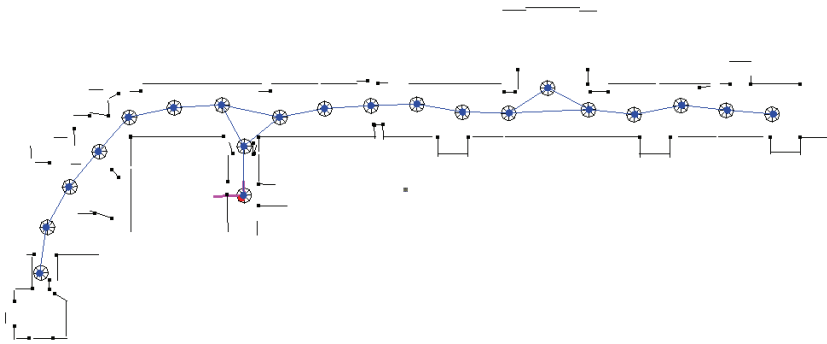
Fig. 16. Map of UPM built in an "Explore and return" experiment

Urbano robot was also deployed in the Indumatica trade fair (Figure 17) in two occasions 2004 and 2005. In both occasions it had to be installed while the fair was open to the public, and the map building was accomplished with the exhibition plenty of people. Next figure shows the map provided by the organizers; as it can be seen it is useless for navigation, as it does not resemble the actual environment. The built map accurately represents the features of the environment.
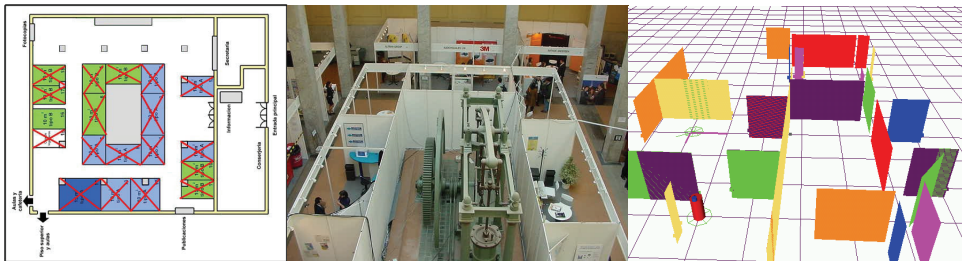
Fig. 17. Indumatica 2004 trade fair. Left) Map provided by organizers. Center) Actual environment. Right) Partial view of the built map.

The map of the environment was built in real time while manually driving *Urbano* in a 102 meters long trajectory in less than five minutes. The complete map and navigation graph is shown in Figure 18, as well as *Urbano* guiding two visitors around the fair.
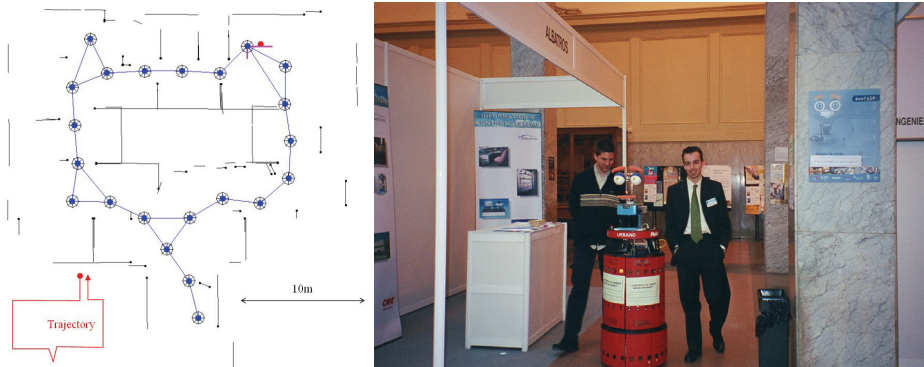
Fig. 18. Indumatica 2004 trade fair. Left) Map built by *Urbano* in real time. Right) *Urbano* guiding two visitors.

The *Urbano* project has been supervised by the "Principe Felipe" museum at the City of Science and Arts of Valencia (CACSA), one of the biggest museums in Spain, as partner and potential end user of *Urbano*. A demonstration of the system deployment was performed (Figure 19), as well as the functionality of *Urbano* as a tour guide. The map of the exhibition was correctly built in real time along a 130 meters long trajectory in approximately 16 minutes.
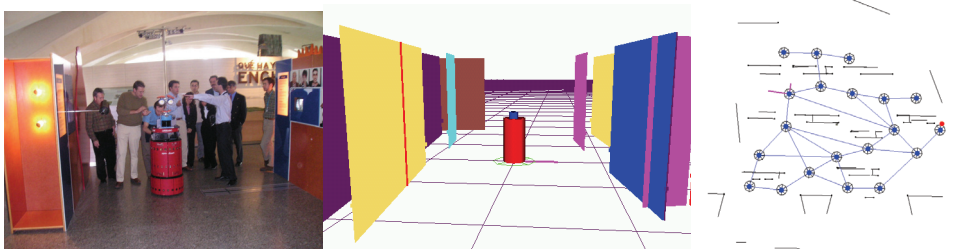
Fig. 19. Map building at Principe Felipe Museum. Left) Manually operating the robot. Center) Real time map building as seen by the installer. Right) Resulting map and navigation graph.

8. Conclusions and future work

The *Urbano* service robot system has been presented in this chapter, with an overview of both its hardware and control software. The hardware used for interaction (robotic face and arm), that has been specifically designed and built for *Urbano* following performance and cost criteria, has been showed to successfully accomplish its task. All the control, navigation, interaction (including speech) and management software has been developed from scratch according to our research lines. These developments have served to increase our scientific

publication records, but have also resulted in the attainment of a quite mature service robot system that has been successfully deployed and tested in several occasions in different scenarios. Moreover, due to its success, we have been requested many times to rent *Urbano* for several days in exhibitions by several institutions and private companies. The only reason we couldn't go on with this renting, was the lack of support in the University for this purpose, as our University is public and a non-profit organization. We are currently considering forming a spin-off to continue with *Urbano* in a more commercial line.

We are currently working in 3D data acquisition, modelling, mapping and navigation in order to achieve a much more robust system (able to detect stairs, obstacles at different heights), that wouldn't require any human supervision (navigation graph editing) for a more automated setup. In fact our goal (Robonauta project, see Acknowledgement) is the fully automated deployment of *Urbano* by showing it the environment, guiding it with natural language, just as it would be done with a new human guide in a museum staff. The interaction capabilities of *Urbano* are also being expanded, implementing some people tracking and following behaviours, as well as an improved image processing system.

The software distributed architecture will also be improved by the standarization of modules interfaces using XML technologies and the (Web Services Definition Language) WSDL specification. In this way, the modules will not require to have the interfaces hard-wired, and more flexibility and simplicity will be allowed for a more fast and error-free development. Also, the programming language will be substituted by some standard as the State Chart XML (SCXML), that could result in a more powerful and simpler to manage tool that took full advantage of the new architecture.

9. Acknowledgment

10. References

Burgard W., Cremers A.B., Fox D., Hähnel D., Lakemeyer G., Schulz D., Steiner W., Thrun S. (1999) Experiences with an interactive museum tour-guide robot. *Artificial Intelligence.* Vol. 1-2 N. 114. pp. 3-55.

Thrun S., Bennewitz M., Burgard W., Cremers A.B., Dellaert F., Fox D., Hahnel D., Rosenberg C., Roy N., Schulte J., Schulz D. (1999). MINERVA: A Second-Generation Museum Tour-Guide Robot. *IEEE International Conference on Robotics and Automation.* Vol.3, pp. 1999-2005.

Nourbakhsh I., Bobenage J., Grange S., Lutz R., Meyer R., and Soto A. (1999). An Affective Mobile Educator with a Full-time Job. *Artificial Intelligence*, Vol. 114, No. 1 - 2, pp. 95-124.

Montemerlo M., Pineau J., Roy N., Thrun S., and Verma, V., (2002). Experiences with a Mobile Robotic Guide for the Elderly. *Proceedings of the AAAI National Conference on Artificial Intelligence*, Edmonton, Canada.

Rodriguez-Losada D., Matia F., Galan R., Jimenez A. (2002). Blacky, an interactive mobile robot at a trade fair. *IEEE International Conference on Robotics and Automation.* Vol. 4. Washington DC, USA. pp. 3930-3935.

Rodriguez-Losada D., Matia F., and Galan R. (2006a) Building geometric feature based maps for indoor service robots. Robotics and Autonomous Systems, vol. 54, pp. 546-558, 2006.

Rodriguez-Losada D., Matia F., Jimenez A., Galan R. (2006b). Local map fusion for real-time indoor simultaneous localization and mapping. Journal of Field Robotics. Wiley Interscience. Vol 23, Issue 5, p 291-309, May 2006

Rodriguez-Losada D., Matia F., Pedraza L., Jimenez A., Galan R. (2007). Consistency of SLAM-EKF Algorihtms for Indoor Environments. Journal of Intelligent and Robotic Systems. Springer. ISSN 0921-0296, Vol. 50, Nº. 4, 2007, pags. 375-397.

Pedraza L., Dissanayake G., Valls Miró J., Rodriguez-Losada D., and Matía F. (2007). BS-SLAM: Shaping the world. In Proc. Robotics: Science and Systems, Atlanta, GA, USA, June 2007.

Castellanos J.A., Montiel J.M.M., Neira J., Tardos J.D. (1999). The SPmap: A Probabilistic Framework for Simultaneous Localization and Map Building. *IEEE Transactions on Robotics and Automation.* Vol. 15 N. 5. pp. 948-953.

Newman P., Leonard J., Tardos J.D., Neira J. (2002) Explore and Return: Experimental Validation of Real-Time Concurrent Mapping and Localization. *IEEE International Conference on Robotics and Automation.* Washington DC, USA. pp. 1802-1809

Lacey G. and Rodriguez-Losada D. (2008) The evolution of Guido: a smart walker for the blind. Accepted for publication in *IEEE Robotics and Automation Magazine.* To appear in 2008.

Fernández, F.; Ferreiros, J.; Pardo, J.M. ; Sama, V.; Córdoba, R. de ; Macías-Guarasa, J.; Montero, J.M.; San Segundo, R.; D´Haro, L.F.; Santamaría, M. & González G. (2006). Automatic understanding of ATC speech. *IEEE Aerospace and Electronic Systems Magazine*, Vol. 21, No 9, pp. 12-17, ISSN: 0885-8985

Córdoba, R. de ; Ferreiros, J.; San Segundo, R.; Macías-Guarasa, J.; Montero, J.M.; Fernández, F.; D´Haro, L.F. & Pardo, J.M. (2006). Air traffic control speech recognition system cross-task & speaker adaptation. *IEEE Aerospace and Electronic Systems Magazine*, Vol. 12, No 9, pp. 12-17, ISSN: 0885-8985

Montero, J.M.; Gutiérrez-Arriola, J.; Córdoba, R.; Enríquez, E. & Pardo, J.M. (2002). The role of pitch and tempo in Spanish emotional speech: towards concatenative synthesis. In: *Improvements in speech synthesis*, Eric Keller y Gerard Bailey, A. Monahan, J. Terken, M. Huckvale (Ed.) pp. 246-251, John Wiley & Sons, Ltd.

Picard R. W., (1997). *Affective Computing*, The MIT Press, Massachusetts, USA. ISBN:0-262-16170-2

Localization and Mapping for Service Robots: Bearing-Only SLAM with an Omnicam

Christian Schlegel and Siegfried Hochdorfer
University of Applied Sciences
D-89075 Ulm
Germany

1. Introduction

Localization and mapping are fundamental problems in service robotics since representations of the environment and knowledge about the own pose significantly simplify the implementation of a series of high level applications. For instance, nearly all relevant applications of service robots require navigation skills that allow for purposeful motions. Typical examples are fetch-and-carry tasks or floor coverage tasks. These are best implemented based on pose knowledge and on continuously updated maps of the environment.

Thus, a key component towards widespread use of service robots is a localization capability that can vary from pose tracking over relocalization to even solving the most demanding so-called kidnapped robot problem. In the latter case the robot is carried to an arbitrary location during its operation and is expected to detect this and then relocalize itself.

Of course, the difficulty of the localization problem significantly depends on the available information. Normally, localization requires some kind of map as reference and map building requires pose knowledge to consistently insert artifacts. A SLAM (simultaneous localization and mapping) problem arises when the robot does neither have access to a map of the environment nor does it know its own pose. The SLAM problem is more difficult than the mapping with known poses and it is more difficult than the localization problem based on a given map.

A successful approach to overcome the chicken-and-egg problem of concurrently building a map and maintaining the robot pose is based on a probabilistic representation. The online SLAM problem maintains the robot pose and the map in a single state vector. The remaining challenge is to estimate the posterior over the current pose along with the map given all the measurements and controls. SLAM is of particular importance for service robotic applications since it significantly reduces deployment efforts and ensures continues updates as needed in dynamic environments. However, one cannot neglect the specific demands on service robots. For instance, in most applications of service robots the consumer neither accepts modifications of the environment (like artifical landmarks) nor complex and time consuming deployment efforts. Although a large body of work already proved that the SLAM problem is solvable even without deploying artifical landmarks, most approaches are based on range measuring devices. For most of the service robotics applications like floor

cleaning or lawn mowing, these devices are either still too expensive (e.g. laser range finders) and do not fit into the budget or do not show the required performance (e.g. ultrasonic sensors in large open spaces like gyms or lobbies). Thus, a SLAM component based on cheap sensors requiring no artifical landmarks is a desired technology for many service robotic applications.

A solution are bearing-only SLAM approaches since these can be used with cheap sensors like omnicams. As long as no calibrated system is needed, omnicams are cheap and small and thus suitable for service robots. Although omnicams provide feature rich information on the surrounding of the robot with high update rates, they do not provide range information. Thus, one has to modify the sensor models of the well-known SLAM approaches such that observation angles of landmarks are sufficient to generate pose estimates. The problem is that one needs several observations of the same landmark from different poses to intersect the line of sights. Thus, one has to solve the so-called problem of a delayed initialization of a landmark. The problem results from the fact that the estimates of the observation poses of not yet initialized landmarks have to be corrected with each reobservation of an already known landmark. Thus, one extends the state vector such that it not only contains the robot pose and the initialized landmark poses but also the observation poses of not yet initialized landmarks.

Fig. 1. The Pioneer-3DX robot with the omnicam in our everyday indoor environment.

In this chapter, we present a bearing-only SLAM system based on an omnicam. After introducing some theoretical foundations, we describe the general approach of bearing-only SLAM with an omnicam based on artificial landmarks. This setting is then extended to get rid of artificial landmarks by exploiting SIFT features (Lowe, 2004). We propose several preselection and landmark identification mechanisms that are pivotal towards the robust application of SIFT features within a bearing-only SLAM approach based on the EKF (Extended Kalman Filter). For example, exploiting viewing areas massively reduces ambiguities and mismatches in SIFT feature reobservations and thus significantly reduces false identifier assignments. The various approaches have been successfully evaluated on a Pioneer-3DX platform in a demanding indoor environment. The experimental evaluation covers characteristic requirements of advanced service robotics environments.

2. Related work

The basic idea of bearing-only SLAM with an Extended Kalman Filter (EKF) is described in (Bailey, 2003). The focus is on solving the delayed landmark initialization problem in an

EKF framework. However, the approach has been evaluated only in simulation with known landmark assignments.

The approach of Bailey is also used by Hesch & Trawny (2005). However, they apply a line filter to an omnicam image and then use vertical lines as landmarks. Again, the evaluation has been performed in simulation only.

We used the approach of Bailey as starting point for real-world evaluations of bearing-only SLAM with an omnicam. The first step was to use artificial landmarks (Schlegel & Hochdorfer, 2005). In a second step, SIFT features in an omnicam image have been evaluated for bearing-only SLAM (Hochdorfer & Schlegel, 2007). Further refinements improve the landmark validation and thus make another step towards suitability for all day use.

Other approaches for bearing-only SLAM address the problem of landmark initialization differently. Immediate initialization methods to bearing-only SLAM have also been introduced. These are called undelayed methods. Kwok et al. (2005) presented a computationally efficient multiple hypothesis approach using Gaussian sum filters to represent the initial feature state. Sola et al. (2005) then gave new insights by introducing an approach that initializes the whole vision cone. Lemaire et al. (2005) use visual features in 3D for bearing-only SLAM. They also applied an undelayed initialization method. Pros and cons of a delayed landmark initialization are discussed by Ortega et al. (2005).

Fitzgibbons & Nebot (2002) report on SLAM with a forward looking camera. The focus there is on color-based feature tracking. Davison et al. (2007) perform 3D SLAM based on a single freely-moving camera. It is also based on a standard full covariance EKF and shows impressive performance.

The main focus of the iterative SIFT feature approach of Tamimi et al. (2006) is on reducing the enormous amount of SIFT features per image. The number of keypoints can be defined in advance and the computation time is proportional to this number. The approach has been evaluated with a SLAM system using an omnicam and a particle filter.

Another improvement of the SIFT feature calculation and an evaluation what are good image features for bearing-only SLAM is presented by Wang & Zhang (2006).

An extensive experimental comparison of state-of-the art techniques for the online bearing-only SLAM problem has been performed by Bekris et al. (2006).

Meanwhile, SIFT features are widely used for SLAM problems. For example, Gil et al. (2006) use SIFT features with a stereo vision system where depth information is available with each feature. However, it is not bearing-only SLAM.

3. SLAM with an extended Kalman filter

3.1 The Kalman filter

This section gives a brief introduction into probabilistic mapping and localization. The basic model is subsequently extended for use with bearing-only SLAM. The presentation is based on (Thrun, 2005).

The Kalman filter is essentially a set of mathematical equations that implement a predictor-corrector type estimator. The Kalman filter is optimal in the sense that it minimizes the estimated error covariance given some presumed conditions are met.

The discrete Kalman filter estimates the n-dimensional state x of a discrete-time controlled process. There is typically a *process model* that models the transformation of the process state. This can usually be represented as a linear stochastic difference equation.

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t \tag{1}$$

The $n \times n$ matrix A_t describes how the state x evolves from $t-1$ to t without controls or noise. The $n \times l$ matrix B_t relates the optional control input u_t of dimension l to the state x. In addition there is a *measurement model* that describes the relationship between the process state and the measurements. This can usually be represented with a linear expression.

$$z_t = C_t x_t + \delta_t \tag{2}$$

The $m \times n$ matrix C_t relates the state to the measurement z_t. Measurements do not have to be of elements of the state but can be any linear combination of the state elements. The random variables ε_t and δ_t represent the process and measurement noise. They are assumed to be independent of each other and normally distributed with zero mean and covariances Q_t and R_t respectively.

The Kalman filter estimates a process by using a form of feedback control. It estimates the process state at some time and obtains feedback in the form of measurements. The *prediction step* projects forward (in time) the current state estimate and the error covariance to obtain the *a priori* estimate for the next time step. The *correction step* is responsible for incorporating a new measurement into the *a priori* estimate to obtain an improved *a posteriori* estimate. Now one can run an ongoing cycle of prediction and correction steps. This recursive nature of the Kalman filter is one of its very appealing features that makes its practical implementation feasible.

The prediction step calculates the *a priori* estimate and the corresponding covariance matrix at time step t according to the following equations. Based on the *a posteriori* estimate of the previous time step $t-1$, the state transformation matrix and the control input, the *a priori* estimate of time step t is determined.

$$\hat{x}_t^- = A_t \hat{x}_{t-1} + B_t u_t \tag{3}$$

$$P_t^- = A_t P_{t-1} A_t^T + Q_t \tag{4}$$

The measurement step incorporates new observations. The first task is to compute the Kalman gain K_t. The Kalman gain sorts out the weights of the measurement and the prediction. It depends on the error covariance of the prediction P_t^- and the measurement error covariance R_t. The measurement and the prediction are then combined in a weighted manner to form the *a posteriori* estimate. Finally, the error covariance matrix belonging to the new estimate is calculated.

$$K_t = P_t^- C_t^T \left(C_t P_t^- C_t^T + R_t \right)^{-1} \tag{5}$$

$$\hat{x}_t = \hat{x}_t^- + K_t \left(z_t - C_t \hat{x}_t^- \right) \tag{6}$$

$$P_t = (I - K_t C_t) P_t^-$$
(7)

3.2 The extended Kalman filter

We so far considered a discrete-time controlled process that is governed by a *linear* stochastic difference equation. However, most realistic robotic problems involve non-linear functions in the process model and/or the measurement model.

$$x_t = g(u_t, x_{t-1})$$
(8)

$$z_t = h(x_t)$$
(9)

The general approach is a first order Taylor series expansion of the process model and the measurement model. A Kalman filter that linearizes about the current mean is referred to as an *Extended Kalman Filter (EKF)*.

The first order Taylor series expansion of the process model is given by:

$$g(u_t, x_{t-1}) \approx g(u_t, \hat{x}_{t-1}) + \frac{\partial g(u_t, \hat{x}_{t-1})}{\partial x_{t-1}}(x_{t-1} - \hat{x}_{t-1})$$
(10)

$$g(u_t, x_{t-1}) \approx g(u_t, \hat{x}_{t-1}) + G_t(x_{t-1} - \hat{x}_{t-1})$$
(11)

The measurement model is linearized as follows:

$$h(x_t) \approx h(\hat{x}_t^-) + \frac{\partial h(\hat{x}_t^-)}{\partial x_t}(x_t - \hat{x}_t^-)$$
(12)

$$h(x_t) \approx h(\hat{x}_t^-) + H_t(x_t - \hat{x}_t^-)$$
(13)

The resulting complete set of EKF equations now require the Jacobians G and H. The prediction step is given by the following equations:

$$\hat{x}_t^- = g(\hat{x}_{t-1}, u_t)$$
(14)

$$P_t^- = G_t P_{t-1} G_t^T + Q_t$$
(15)

The correction step is calculated according to the following equations:

$$K_t = P_t^- H_t^T (H_t P_t^- H_t^T + R_t)^{-1}$$
(16)

$$\hat{x}_t = \hat{x}_t^- + K_t(z_t - h(\hat{x}_k^-))$$
(17)

$$P_t = (I - K_t H_t) P_t^-$$
(18)

3.3 The EKF and the SLAM problem

The Extended Kalman Filter can now be used to solve the SLAM problem. The seminal idea is to have a state vector comprising both the robot pose and the landmark poses. Now the overall state can be estimated at once. Thus, the chicken-and-egg problem is solved. In case of a robot pose $(x, y, \phi)^T$ and a map of N landmarks $(x, y)^T$, we have a $(3 + 2N)$ dimensional Gaussian to represent the state of the SLAM problem.

The online SLAM problem estimates the posterior over the current pose along with the map:

$$p\left(x_t, m \mid z_{1:t}, u_{1:t}\right) \tag{19}$$

The robot pose at time t is denoted x_t, m is the map, and z_{1t} and u_{1t} are the measurements and controls, respectively. Most algorithms for the online SLAM problem are incremental that is they discard past measurements and controls once these have been processed. Maps in EKF based SLAM approaches are feature-based. A map feature is usually represented as a point landmark m_i with coordinates x_i and y_i.

The Kalman filter consists of a prediction and a correction step executed in a loop. For the SLAM problem, the prediction step is based on a motion model of the robot taking into account the control inputs. The correction step integrates landmark observations and is based on an observation model. The observation model relates the state vector to the measurement, that is, it allows to calculate the expected measurement.

4. Bearing-only SLAM

4.1 The state vector

Bearing-only SLAM requires several observations of the same landmark from different robot poses to intersect line of sights to calculate a landmark pose. Since a new landmark can be initialized only after collecting a bunch of measurements, we call this a delayed landmark initialization. The problem now is that while collecting several observations, the robot pose already gets updated by the SLAM mechanism. Since the various observation poses are not independent of each other, these need to be updated as well. Otherwise, the lateron performed intersection of line of sights from different observation poses would not be consistent anymore. The basic idea now is to include the observation poses into the SLAM state vector. Now these observation poses are consistently updated by the SLAM mechanism. Since the measurements are relative to the observation poses and since these are updated consistently, we can consistently transform angular observations into line of sights in the global frame of reference even when the measurements are from different points in time. Thus, a deferred but consistent evaluation of measurements is possible. Measurements of a possible landmark can now be accumulated over time until sufficient information is available for a reliable landmark initialization.

The extended state vector for bearing-only SLAM with a delayed landmark initialization is thus given by (Bailey, 2003):

$$x = \left[x_v^T, x_{v_m}^T, \ldots, x_{v_1}^T, x_{f_1}^T, \ldots, x_{f_n}^T\right]^T \tag{20}$$

The vehicle pose $x_v = [x_v, y_v, \phi_v]^T$ is the position and heading of the robot in the global frame of reference. The entry $x_{v_i} = [x_{v_i}, y_{v_i}, \phi_{v_i}]^T$ represents an observation pose where not yet evaluated measurements are available. Each pose x_{v_i} corresponds to the time and location where a set of measurements $\{\theta_{v_i}^1, \dots, \theta_{v_i}^k\}$ was obtained. Already initialized landmarks $x_{f_i} = [x_{f_i}, y_{f_i}]^T$ are represented by their 2D position without orientation.

4.2 The overall processing scheme

Before describing each step of the bearing-only SLAM approach in detail, we first give a short overview on the overall processing scheme. The processing order per time-step executing the loop of correction and prediction steps of the EKF is as follows (Bailey, 2003):

- Perform prediction step according to motion model.
- Get image from omnicam and get robot pose from odometry. Next, we extract features and determine angle to features.
- Measurements of existing map features are processed first in a batch update.
- If there exists a well-conditioned pair of measurements for a non-initialized landmark, the initial landmark estimate is added to the SLAM state vector.
- For each newly initialized feature, the remaining accumulated measurements are applied in a batch update.
- Observation poses x_{v_i} are removed from the SLAM state vector as soon as they do no longer possess not yet processed measurements. In that case these observation poses are not needed any longer.
- In case the current robot pose provides a measurement for a not yet initialized landmark, the observation is stored and the current robot pose is added as new observation pose to the SLAM state vector by *stochastic cloning* (Roumeliotis & Burdick, 2002):

$$\begin{bmatrix} \hat{x}_v^T \\ \hat{x}_m^T \end{bmatrix} \rightarrow \begin{bmatrix} \hat{x}_v^T \\ \hat{x}_v^T \\ \hat{x}_m^T \end{bmatrix} \; , \quad \begin{bmatrix} P_{vv} & P_{vm} \\ P_{vm}^T & P_{mm} \end{bmatrix} \rightarrow \begin{bmatrix} P_{vv} & P_{vv} & P_{vm} \\ P_{vv} & P_{vv} & P_{vm} \\ P_{vm}^T & P_{vm}^T & P_{mm} \end{bmatrix} \tag{21}$$

4.3 The motion model for the Kalman filter

The motion model describes the pose change of the robot given the current pose and control commands. Since action execution is always subject to errors, the motion model is used with stochastic variables. The motion model then propagates the pose estimate according to the motion model and given control commands and adds the additional uncertainty representing flawed execution.

As can be seen in figure 2, the pose x_{k+1} is reached from pose x_k by applying the virtual control command $(d_1, \Delta\alpha_1, \Delta\alpha_2)$. This virtual control command is calculated out of two

odometry reports x_{k+1} and x_k. Thus, we regularly sample the odometry values and approximate each motion between two poses by first turning towards the new pose x_{k+1} by $\Delta\alpha_1$, then driving the distance d and finally adjusting the heading by a final turn $\Delta\alpha_2$. The motion model can now be used to carry forward covariances of pose estimates based on two subsequent odometry reports. This model always considers the total amount of rotation even for S-shaped trajectories. This is important since rotations induce high uncertainties on the pose estimate of the robot. It now depends on the odometry sampling rate on how tight the approximation (blue line) captures the actual motion (red line).
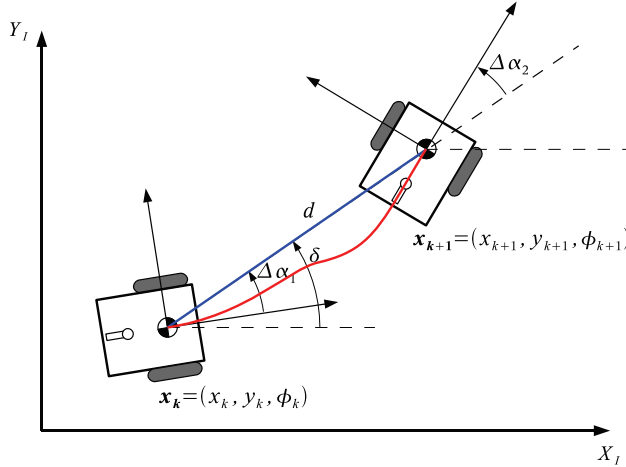
Fig. 2. The robot motion model.

The full motion model is given by the following equation:

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \\ \phi_{k+1} \end{pmatrix} = m(x_k, y_k, \phi_k, d, \Delta a_1, \Delta a_2, \Delta\beta) = \begin{pmatrix} x_k + d\cos(\phi_k + \Delta a_1) \\ y_k + d\sin(\phi_k + \Delta a_1) \\ \phi_k + \Delta a_1 + \Delta a_2 + \Delta\beta \end{pmatrix} \quad (22)$$

The parameters of the motion model are calculated as follows with $\Delta\beta$ a drift error that depends on the driving distance:

$$d = \sqrt{(y_{k+1} - y_k)^2 + (x_{k+1} - x_k)^2} \quad (23)$$

$$\delta = arctan\left(\frac{y_{k+1} - y_k}{x_{k+1} - x_k}\right) \quad (24)$$

$$\Delta a_1 = \delta - \phi_k \quad (25)$$

$$\Delta a_2 = \phi_{k+1} - \delta \tag{26}$$

$$\Delta \beta = s(d) \tag{27}$$

The Jacobian, which is needed to propagate the covariances in the linearized motion model of the EKF, is given by

$$
\frac{\partial m(\dots)}{\partial \left(x_k, y_k, \varphi_k, d, \Delta \alpha_1, \Delta \alpha_2, \Delta \beta \right)} =
$$

$$
\begin{bmatrix}
1 & 0 & -d \sin\left(\varphi_k + \Delta \alpha_1\right) & \cos\left(\varphi_k + \Delta \alpha_1\right) & -d \sin\left(\varphi_k + \Delta \alpha_1\right) & 0 & 0 \\
0 & 1 & d \cos\left(\varphi_k + \Delta \alpha_1\right) & \sin\left(\varphi_k + \Delta \alpha_1\right) & d \cos\left(\varphi_k + \Delta \alpha_1\right) & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 1 & 1
\end{bmatrix} \tag{28}
$$

The variances of the control commands are given by $\sigma_d^2 = |d| \lambda_d$ with $\lambda_d \left[mm^2 / mm \right]$, $\sigma_{\Delta\alpha}^2 = |\Delta\alpha| \lambda_{\Delta\alpha}$ with $\lambda_\alpha \left[rad^2 / rad \right]$ and $\sigma_{\Delta\beta}^2 = |d| \lambda_{\Delta\beta}$ with $\lambda_\beta \left[rad^2 / mm \right]$.

4.4 Angular measurements in omnicam images

An omnicam image maps the environment radially. The center point corresponds to the optical axis of the camera. It is important to note that we only use the observation angles of landmarks. The observation angle of a landmark in the omnicam image is equivalent to the yaw-angle γ of the landmark in the 3d-environment when using polar-coordinates (pitch β, yaw γ, distance d). As can be seen in figure 3, the image distortion of the omnicam does not affect the observation angle. Of course, all landmarks with the same yaw-angle γ but different pitch-angles β possess the same observation angle for the SLAM procedure.
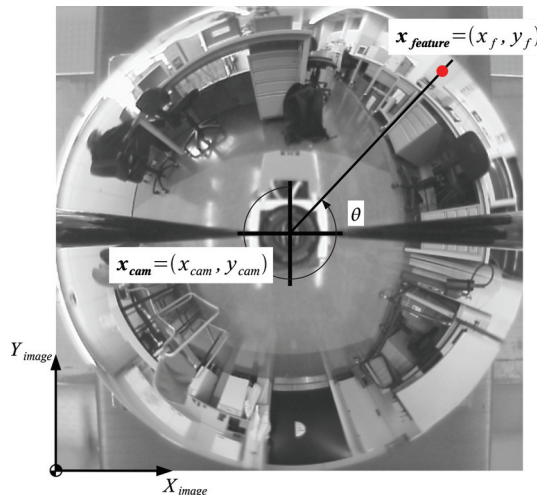
Fig. 3. Calculation of the observation angle of a feature in an omnicam image.

The calculation of the observation angle can be done in the image coordinate system of the omnicam image. The optical axis of the camera in image coordinates is denoted by x_{cam} and the feature position in image coordinates by $x_{feature}$. The observation angle relative to the robot is denoted by $z_{observation\ angle}$.

$$z_{observation\ angle} = h(x_{cam}, x_{feature}) = arctan\left(\frac{y_{feature} - y_{cam}}{x_{feature} - x_{cam}}\right) + n_i \qquad (29)$$

The measurement error n_i is modeled as zero-mean white gaussian noise with variance σ_θ^2. The variance is set to a fixed value independent of the image coordinates of the landmark. This approach assigns the same worst case uncertainty to all landmark observation angles. We assume a one-pixel jitter of a landmark position in image coordinates. The closer the landmark is to the center point, the bigger is the effect of the one-pixel jitter on the angular error. However, the minimum distance of landmarks to the center point is restricted due to the characteristics of the omnicam. The center part of the image contains the robot itself and thus is masked out for potential landmarks.

4.5 The observation model for the Kalman filter
The observation model of a landmark is used to integrate another observation of an already initialized landmark by means of the Kalman filter. The Kalman filter exploits the difference between the expected angular measurement and the current measurement. Instead of image coordinates, we now use the current robot pose estimate x_v and the current pose estimate of the considered landmark x_{f_i}. Of course, we now have to take into account the robot heading ϕ_v. The observation model can be expressed by

$$z_i = h(x_v, x_{f_i}) = arctan\left(\frac{y_{f_i} - y_v}{x_{f_i} - x_v}\right) - \phi_v + n_i \qquad (30)$$

with n_i the measurement error modeled as zero-mean white gaussian noise with variance σ_θ^2. Again, a fixed value representing the worst case is used for all landmark angles.

The EKF update equations require the Jacobian of the observation model with d_{v,f_i} the euclidean distance between vehicle and landmark. The position of H_{f_i} in H corresponds to the position of the landmark in the state vector.

$$H = \nabla_x h(\hat{x}) = \begin{bmatrix} H_v & 0\dots0 & H_{f_i} & 0\dots0 \end{bmatrix} \qquad (31)$$

$$H_v = \begin{bmatrix} \dfrac{\partial h}{\partial \hat{x}_v} & \dfrac{\partial h}{\partial \hat{y}_v} & \dfrac{\partial h}{\partial \hat{\phi}_v} \end{bmatrix} = \begin{bmatrix} \dfrac{\hat{y}_{f_i} - \hat{y}_v}{\hat{d}_{v,f_i}^2} & -\dfrac{\hat{x}_{f_i} - \hat{x}_v}{\hat{d}_{v,f_i}^2} & -1 \end{bmatrix} \qquad (32)$$

$$H_{f_i} = \begin{bmatrix} \dfrac{\partial h}{\partial \hat{x}_{f_i}} & \dfrac{\partial h}{\partial \hat{y}_{f_i}} \end{bmatrix} = \begin{bmatrix} -\dfrac{\hat{y}_{f_i} - \hat{y}_v}{\hat{d}^2_{v,f_i}} & \dfrac{\hat{x}_{f_i} - \hat{x}_v}{\hat{d}^2_{v,f_i}} \end{bmatrix} \tag{33}$$

4.6 Landmark initialization

For proper initialization of a landmark in SLAM with bearing-only information, at least two bearing measurements z_i and z_j from two different vehicle poses x_{v_i} and x_{v_j} are needed.

In case of no errors, the true location of the landmark would then be given by intersecting two lines given in point-slope form as illustrated in figure 4.
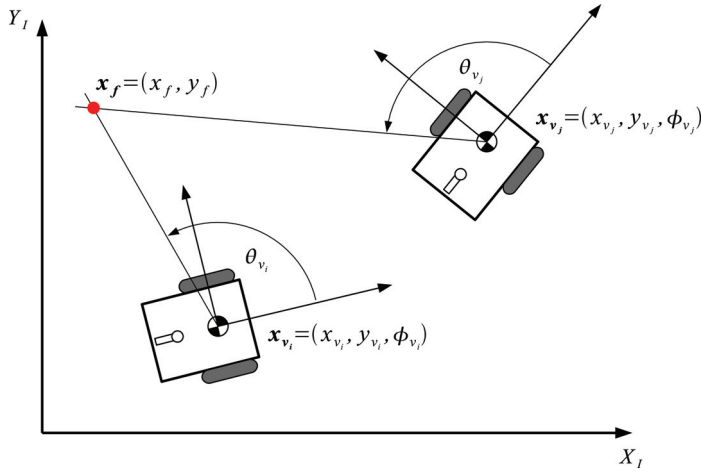
Fig. 4. Calculation of the landmark position based on two angular observations.

Given the following point-slope form

$$\begin{aligned} \left(y_f - y_{v_i} \right) &= \tan\left(\theta_i + \phi_{v_i} \right)\left(x_f - x_{v_i} \right) \\ \left(y_f - y_{v_j} \right) &= \tan\left(\theta_j + \phi_{v_j} \right)\left(x_f - x_{v_j} \right) \end{aligned} \tag{34}$$

the intersection is obtained by:

$$\begin{aligned} x_f &= \begin{bmatrix} x_f \\ y_f \end{bmatrix} = g\left(x_{v_i}, x_{v_j}, \theta_i, \theta_j \right) \\ &= \begin{bmatrix} g_x(\dots) \\ g_y(\dots) \end{bmatrix} = \begin{bmatrix} \dfrac{x_{v_i} s_i c_j - x_{v_j} s_j c_i + \left(y_{v_j} - y_{v_i} \right) c_i c_j}{s_i c_j - s_j c_i} \\[4mm] \dfrac{y_{v_j} s_i c_j - y_{v_i} s_j c_i + \left(x_{v_i} - x_{v_j} \right) s_i s_j}{s_i c_j - s_j c_i} \end{bmatrix} \end{aligned} \tag{35}$$

where we abbreviate $s_i = sin\left(\phi_{v_i} + \theta_i \right)$ and $c_i = cos\left(\phi_{v_i} + \theta_i \right)$.

In case of noise-corrupted vehicle poses and measurements, the landmark estimate is given by

$$\begin{bmatrix} \hat{x}_f \\ \hat{y}_f \end{bmatrix} = g\left(\hat{x}_{v_i}, \hat{x}_{v_j}, z_i, z_j\right) \tag{36}$$

with $z_i = \theta_i + n_i$ and $z_j = \theta_j + n_j$ where n_i and n_j denote zero mean and white gaussian noise. Using first order Taylor series expansion for $g(\ldots)$ and taking into account that the measurement noise is independent of the vehicle pose, one can approximate the 2×2 covariance matrix P_{LL} of the landmark position estimate by

$$P_{LL} = GPG^T + W \begin{pmatrix} \sigma_\theta^2 & 0 \\ 0 & \sigma_\theta^2 \end{pmatrix} W^T \tag{37}$$

and the correlation with the other entries in the SLAM state vector, P_{LX} and P_{XL}, by $P_{LX} = P_{XL}^T = HP$ assuming uncorrelatedness between system and measurement error (Hesch & Trawny, 2005) where $t = \left(s_i c_j - s_j c_i\right)^2$ and

$$G = \nabla_x g(\hat{x}) = \begin{bmatrix} G_{v_i} & 0\ldots 0 & G_{v_j} & 0\ldots 0 \end{bmatrix} \tag{38}$$

$$G_{v_i} = \nabla_{x_{v_i}} g(\hat{x}) = \begin{pmatrix} \dfrac{\partial g_x}{\partial x_{v_i}} & \dfrac{\partial g_x}{\partial y_{v_i}} & \dfrac{\partial g_x}{\partial \varphi_{v_i}} \\[2mm] \dfrac{\partial g_y}{\partial x_{v_i}} & \dfrac{\partial g_y}{\partial y_{v_i}} & \dfrac{\partial g_y}{\partial \varphi_{v_i}} \end{pmatrix}$$

$$= \frac{1}{t}\begin{pmatrix} s_i c_j\left(s_i c_j - s_j c_i\right); & -c_i c_j\left(s_i c_j - s_j c_i\right); & \left(\hat{x}_{v_j} - \hat{x}_{v_i}\right)s_j c_j - \left(\hat{y}_{v_j} - \hat{y}_{v_i}\right)c_j^2 \\ s_i s_j\left(s_i c_j - s_j c_i\right); & -s_j c_i\left(s_i c_j - s_j c_i\right); & \left(\hat{x}_{v_j} - \hat{x}_{v_i}\right)s_j^2 - \left(\hat{y}_{v_j} - \hat{y}_{v_i}\right)s_j c_j \end{pmatrix} \tag{39}$$

$$G_{v_j} = \nabla_{x_{v_j}} g(\hat{x})$$

$$= \frac{1}{t}\begin{pmatrix} -s_i c_j\left(s_i c_j - s_j c_i\right); & -c_i c_j\left(s_i c_j - s_j c_i\right); & \left(\hat{x}_{v_i} - \hat{x}_{v_j}\right)s_i c_i + \left(\hat{y}_{v_j} - \hat{y}_{v_i}\right)c_i^2 \\ -s_i s_j\left(s_i c_j - s_j c_i\right); & -s_j c_i\left(s_i c_j - s_j c_i\right); & \left(\hat{x}_{v_i} - \hat{x}_{v_j}\right)s_i^2 + \left(\hat{y}_{v_j} - \hat{y}_{v_i}\right)s_i c_i \end{pmatrix} \tag{40}$$

$$W = \nabla_n g(\hat{x})$$

$$= \frac{1}{t}\begin{pmatrix} \left(\hat{x}_{v_j} - \hat{x}_{v_i}\right)s_j c_j - \left(\hat{y}_{v_j} - \hat{y}_{v_i}\right)c_j^2; & \left(\hat{x}_{v_i} - \hat{x}_{v_j}\right)s_i c_i + \left(\hat{y}_{v_j} - \hat{y}_{v_i}\right)c_i^2 \\ \left(\hat{x}_{v_j} - \hat{x}_{v_i}\right)s_j^2 - \left(\hat{y}_{v_j} - \hat{y}_{v_i}\right)s_j c_j; & \left(\hat{x}_{v_i} - \hat{x}_{v_j}\right)s_i^2 + \left(\hat{y}_{v_j} - \hat{y}_{v_i}\right)s_i c_i \end{pmatrix} \tag{41}$$

4.7 Deciding on landmark initialization

The above described landmark initialization can only be applied if suitable observations are available. An initialization is performed only if there exists a well-conditioned pair of measurements for a non-initialized landmark. As described in detail in (Bailey, 2003), "a new feature is considered well-conditioned if the true probability density function of its location closely resembles the Gaussian approximation obtained from a Jacobian-based linearized transform".

The initial landmark pose estimate is calculated by intersecting line of sights. This calculation includes non-linear operations. Thus, the calculation of the covariance of the initial landmark estimate is not straightforward. The standard approach is to use the above derived linearizations of the calculation of the intersection point. This allows to transform the covariances of the observation poses and of the angular measurements into a covariance of the initial landmark estimate. However, the result might deviate substantially from the true distribution due to the effects of linearization. Since the Kalman filter is extremely brittle with respect to such inconsistencies, we have to introduce some kind of "quality measure" that only selects well-conditioned pairs of measurements.

For this, the difference between the linearized and the non-linearized calculation is considered. In short, the algebraic density transformation of the 8-D Gaussian distribution $p(x_{v_i}, x_{v_j}, \theta_{v_i}, \theta_{v_j})$ into the 8-D probability density function $p(x_{v_i}, x_{v_j}, x_f)$ is emulated by sampling. Each sample is transformed to a landmark pose according to the equation for calculating the intersection point. As result, a set of samples representing the non-Gaussian distribution of the landmark estimate is available. Each sample is scaled by the normalization factor of the algebraic density transformation evaluated at its sample value. This is the weight w_k. Furthermore, each sample gets another weight v_k based on the Gaussian of the linearized transformation. The sample relative entropy can now be calculated based on both weights by $\frac{1}{n} \sum_{k=1}^{n} \left(\ln w_k - \ln v_k \right)$. This is an approximation of the *Kullback-Leibler-distance* (Cover & Thomas, 1991). The smaller the relative entropy value is, the more similar are both probability density functions. Of course, this is only a heuristic to give a hint on whether a pair of measurements results in an initial landmark estimate that does not suffer too much from linearization effects.

5. Bearing-only SLAM with artificial landmarks

5.1 Feature extraction with artificial landmarks

The color segmentation is performed in HSV color space. The segmentation is based on static intervals per predefined landmark color for the hue and saturation channels. A color blob is identified as landmark if its size and its compactness values fall into a predefined interval. The angle is defined by the polar line starting in the center of the image and going through the color blob such that the lateral error of the color blob boundary with respect to the line is minimal. Of course, the data association problem cannot be solved in case of nearby landmarks of the same color. However, for evaluation purposes of the overall bearing-only SLAM approach, we manually placed landmarks such that different landmarks of the same color never appear close to each other in the omnicam image.
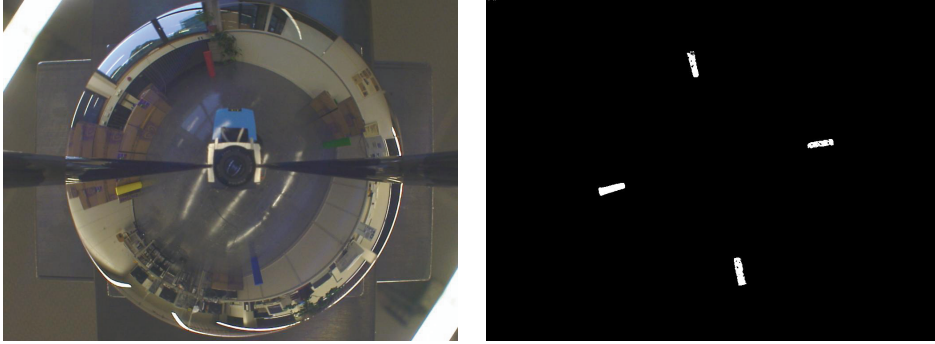
Fig. 5. The left image shows the omnicam image with colored cylinders as artificial landmarks. The right image shows the result after the segmentation step.

5.2 Experimental setup

To perform the experiments on a real world platform, we implemented all necessary steps such that a closed loop system including a Pioneer-3DX robot, an omnicam and the bearing-only SLAM algorithm is available. The omnicam is a Sony DFW-X710 camera (1024x768, 1/3 inch, progressive scan, firewire, YUV color, 15 images/second) with a hyperbolic glass mirror (H3G, Neovision). The robot is equipped with an on-board Pentium M class small sized PC. The software is Linux based and is a combination of C++ code with calls to a Matlab server for the SLAM algorithm. The image processing is based on the *Open Source Computer Vision Lib* (Intel, 2008).

The parameters of the motion model are $\lambda_d = (0,05m)^2 / 1m$ (distance error), $\lambda_\alpha = (5\deg)^2 / 360\deg$ (rotational error) and no drift error. The sensor model uses $\sigma_\theta^2 = (1.5\deg)^2$ as angular error of the landmark detection independent of the image coordinates of the landmark. This value is set larger than the theoretical one-pixel error since the segmentation process introduces further jitter on the object boundary with effects on the landmark center point. The threshold of the distance metric to decide on the initial integration of a landmark is set to 12. This threshold has been determined empirically. We set the threshold such that the angle enclosed by two line of sights of a well-conditioned pair of measurements is not below a minimum angle. This typically avoids ill-formed Gaussians for the landmark pose estimate. The robot typically moved about 1m between measurements.

5.3 Experimental results

The environment of the loop closure experiment is shown in figure 6. Figure 7 shows the sensing steps 1, 4, 5, 14, 23, 28 and 31 of a 36 step run. The ellipses show the 2-sigma contour. All units are meter. The map rotates since there is no absolute initial reference. The green lines indicate which landmark was seen by the robot in that step. The bottom right figure shows the standard error over all relative landmark distances. Since the landmarks are static, we can easily determine the ground truth relative distance between each pair of landmarks. The relative distances can also be calculated based on the landmark poses estimated by the SLAM approach. The squared error between the actual and the estimated

distances is then summed up and plotted over time steps. Of course, the overall error grows by including new landmarks since more relations between landmarks are covered. However, as long as the number of landmarks is not extended, the correction steps reduce the overall error in the landmark relationships and thus improve the overall consistency of the landmark poses.

The experiments on a real platform prove that EKF-based bearing-only SLAM methods can be applied to features extracted from an uncalibrated omnicam. The 2-sigma value of the robot pose error is typically 15cm. However, artificial landmarks are not suitable for most applications and we thus have to get rid of them.

Fig. 6. The everyday indoor lab environment where this experiment has been performed.
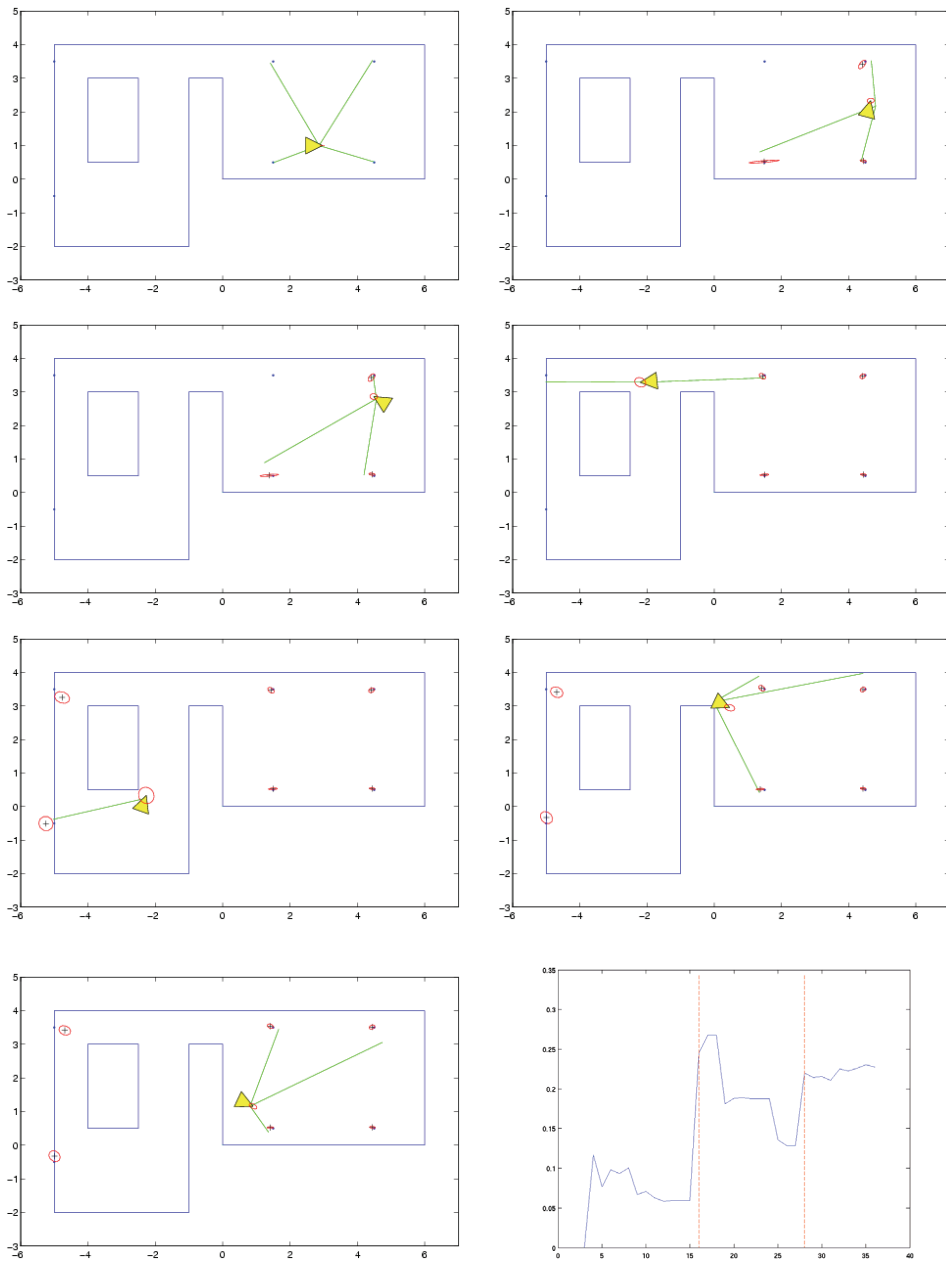
Fig. 7. The sensing steps 1, 4, 5, 14, 23, 28 and 31 of a 36 step run with 2-sigma contours. This experiment is based on artificial landmarks.

6. Bearing-only SLAM with SIFT features

Instead of artificial landmarks, we now consider SIFT features as natural landmarks for bearing-only SLAM. The SIFT approach (scale invariant feature transforms) takes an image and transforms it into a "large collection of local feature vectors" (Lowe, 2004). Up to a certain degree, each feature vector is invariant to scaling, rotation or translation of an image. SIFT features are also very resilient to the effects of noise in an image. For instance, we do not have to rely on specific shape or color models.

Depending on the parameter settings, a single omnicam image contains up to several hundred SIFT features. However, the Kalman filter based approach shows two characteristics that need to be addressed. It does not scale well with increasing numbers of landmarks and it is very brittle with respect to false observation assignments. Thus, one needs a very robust mechanism to select a small but stable number of SIFT features in an image. Potential landmarks have to be distributed sparsely over the image and should also possess characteristic descriptor values to avoid false assignments.

We still represent landmarks by 2-D poses as already explained in the previous section.

6.1 Calculation of SIFT features

SIFT features of an image are calculated by a four-step procedure. We apply the plain calculation scheme described in detail in (Lowe, 2004).

The first step is named *scale-space extrema detection*. The input images of the omnicam are of size 480x480 pixels. The first octave consists of five images, that is the original image and another four images. The latter are obtained by repeatedly convolving the original image with Gaussians. We use a σ-value of 2.4. This parameter is very robust and can thus be determined empirically. A larger value increases the computational load without improving the re-recognition of SIFT features. It is set such that the output of the overall processing chain is a stable set of roughly 90 SIFT features. In the next step, the four DOG (difference of Gaussians) images are calculated. Afterwards, extrema are detected in the two inner DOG images by comparing a pixel to its 26-neighbors in 3x3 regions. We use a down-sampling factor of 2 where downsampling ends at an image of 4x4 pixels. Therefore, we consider 7 octaves. The different octaves are illustrated in figure 8.
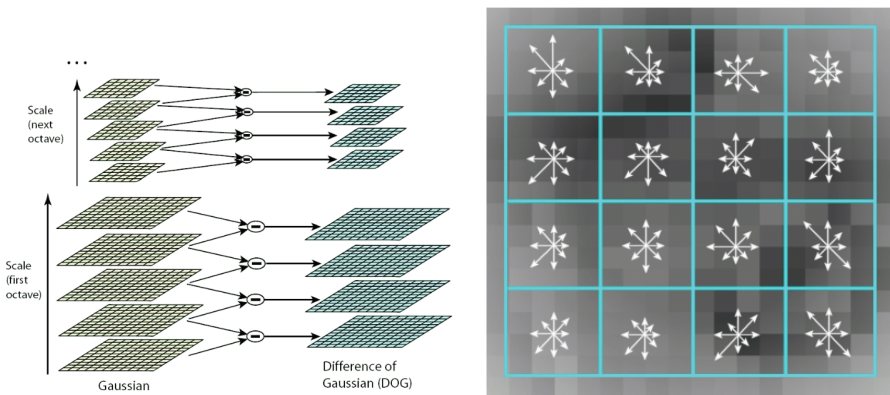
Fig. 8. The left image shows the structure of the scale space (Lowe, 2004). The right image shows the structure of a keypoint descriptor (Rihan, 2005).

The second step is named *keypoint localization*. The contrast threshold is set to 0.10. Again, the value is determined empirically. We first set it such that we obtain stable landmarks. Then we modify this threshold to reduce the number of obtained landmarks. Finally, we modify it to reduce the computational load without further reducing the number of landmarks. The curvature threshold is set to 10 (same as Lowe).

The third step is named orientation assignment. Since we do not further exploit the orientation value, we omit this step.

The fourth step is named keypoint descriptor. As described in (Lowe, 2004), we use 4x4 sample regions with 8 gradients and perform a Gaussian weighting with σ=1.5. The result are SIFT feature vectors each of dimension 128 with 8 bit entries.

6.2 The overall sequence of processing steps using SIFT features

The overall sequence of processing steps is shown in figure 9. It is important to note that we still only use the observation angles of landmarks. In case of distinct SIFT feature vectors, we just have different landmarks at the same observation angle independently of the pitch-angle value. Identical SIFT feature vectors at the same yaw-angle but at different pitch-angles need not to be discriminated since we only exploit the yaw-angle.
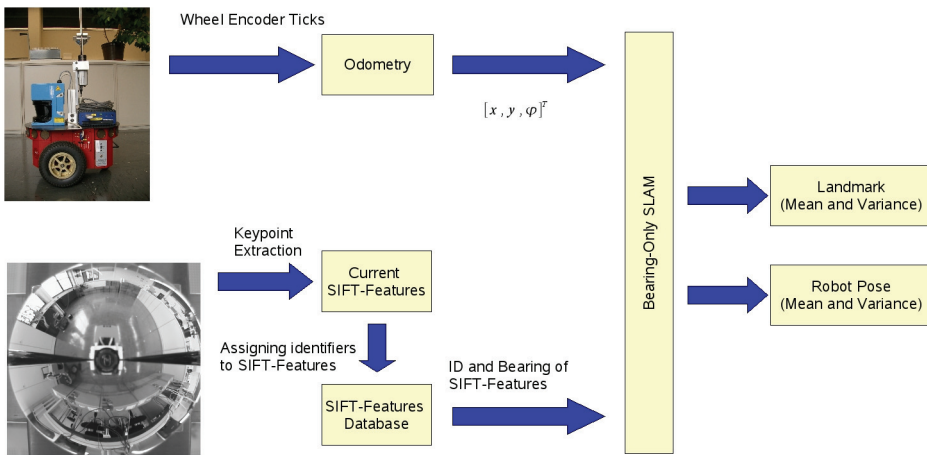
Fig. 9. The overall bearing-only SLAM system based on SIFT features.

6.3 Processing of an image

Each omnicam image is reduced to a 480x480 resolution. SIFT features are extracted based on the standard attributes (gaussian filter, contrast, curvature ratio). Since the omnicam image also comprises the robot and mountings of the camera, we again remove all landmarks in those areas by a simple masking operation.

6.4 Assigning identifiers to SIFT-features

The decision tree behind the identifier assignment procedure is illustrated in figure 10. The SIFT feature descriptors of the current image are compared with all the SIFT feature descriptors of the previous images. However, we only consider those images where the

euclidean distance to the image acquisition pose is less than two times the maximum viewing distance of the omnicam (in our case 15m). This preselection significantly reduces the computational load of the comparisons of the descriptors. The viewing range is motivated by the typical size of free space in indoor environments.
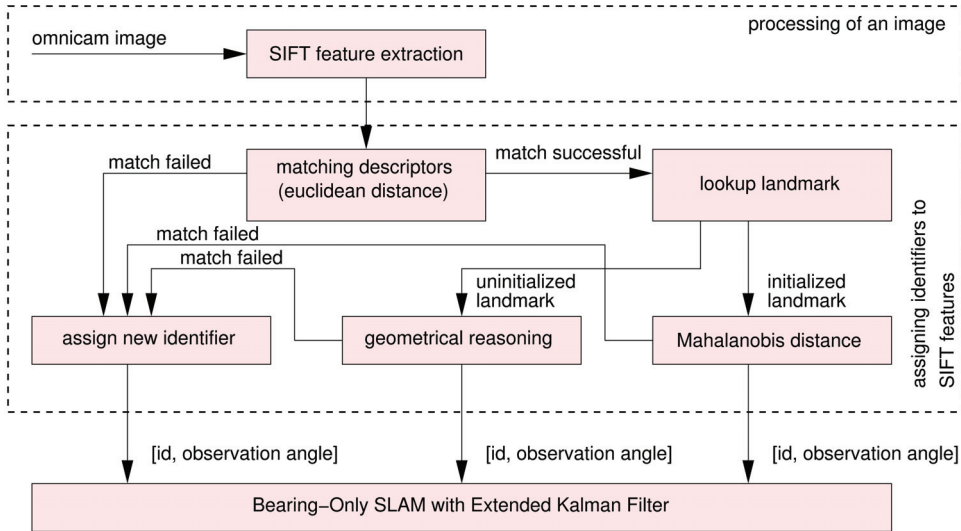
Fig. 10. The decision tree behind the identifier assignment procedure.

Next the euclidean distance between the current SIFT feature vectors and the remaining ones of the previous steps are calculated. A SIFT feature of the current image is considered as not matching an already known landmark (either initialized or not initialized landmark) if the ratio of the smallest and second smallest distance value is above a given threshold (value 0.6, see (Lowe, 2004)). In that case, this SIFT feature gets a new and unique identifier. This SIFT feature is the first observation of a potentially new landmark (first measurement of an uninitialized landmark).

Otherwise, the SIFT feature is considered as matching an already known landmark. In this case, we have to distinguish whether the SIFT feature matched an initialized or an uninitialized landmark.

In the first case, the considered SIFT feature is just a reobservation of an already known landmark which is validated by a test based on the Mahalanobis distance (Hesch & Trawny, 2005). In case of passing this test, the measurement is forwarded to the EKF as reobservation of the initialized landmark. Otherwise, the current measurement and its SIFT feature is the first observation of a potentially new landmark (first measurement of an uninitialized landmark).

In the second case, we solely have several observations (bearing-only measurements) of the same SIFT feature (uninitialized landmark) from different observation poses. Since in that case we cannot apply the Mahalanobis distance, we use geometrical reasoning for validating the reobservation. The new observation can belong to the uninitialized landmark only if its viewing direction intersects the visual cone given by the previous measurements of this uninitialized landmark. In that case, this SIFT feature is considered as a new observation of this not yet initialized landmark. Otherwise, this SIFT feature is the first observation of a potentially new landmark (first measurement of an uninitialized landmark).

6.5 Geometrical reasoning

In case of an uninitialized landmark, covariances are not yet available. Thus, we cannot apply the Mahalanobis distance to validate the assignment. Therefore, we apply a simple geometrical validation scheme that reliably sorts out impossible matches. In figure 11, *P2* denotes the current robot pose with *a* being the vector towards the previous robot pose *P1* and *c2* limiting the viewing range. At *P1* a landmark *L* has been seen with heading *b* and a maximum distance as indicated by *c1*. Thus, *L* can only be seen from *P2* in case its observation angle is in the viewing angle *r*. However, the closer the landmark is to the half-line *P1P2*, the less selective is the viewing angle *r*. In worst case, the full range of 180 degree remains. The closer the landmark is to the half-line *P2P1*, the more selective is this approach. In best case, a viewing angle close to zero remains.
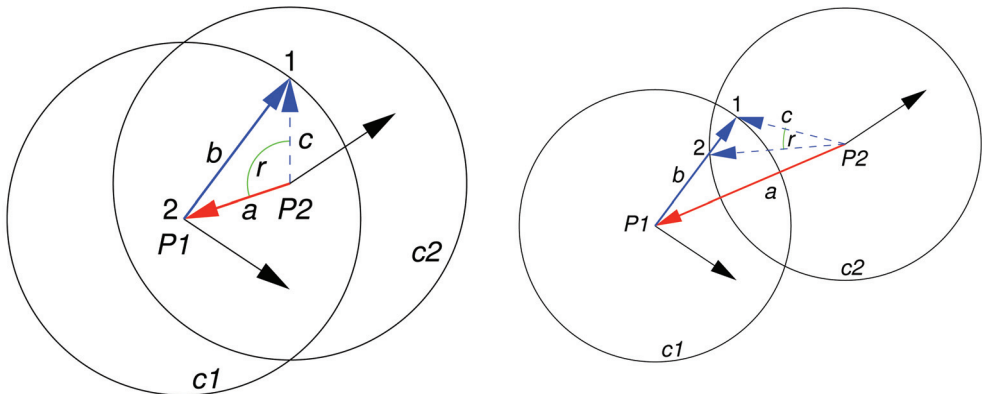
Fig. 11. Geometrical validation of matches.

6.6 Experimental setup

Due to extended experiments with our Pioneer-3DX platforms, we could meanwhile further improve the parameters of our motion model. The updated values are $\lambda_d = (0,03m)^2 / 1m$ (distance error), $\lambda_\alpha = (4\deg)^2 / 360\deg$ (rotational error) and still no drift error. The sensor model uses $\sigma_\theta^2 = (0.5\deg)^2$ as angular error of the landmark detection independent of the image coordinates of the landmark. The improved value results from the sub-pixel resolution of the SIFT feature keypoint location. The threshold of the distance metric to decide on the initial integration of a landmark is now reduced to 3. The reduced value is stricter with respect to landmark initializations. This adjustment is possible due to the higher accuracy of the angular measurements.

6.7 Experimental results

The experiment is performed in the same environment as the previous experiment but now without any artificial landmarks. The only difference is another wall that separated the free space into two sections and restricted the view. Thus, the scenario required another loop closure.
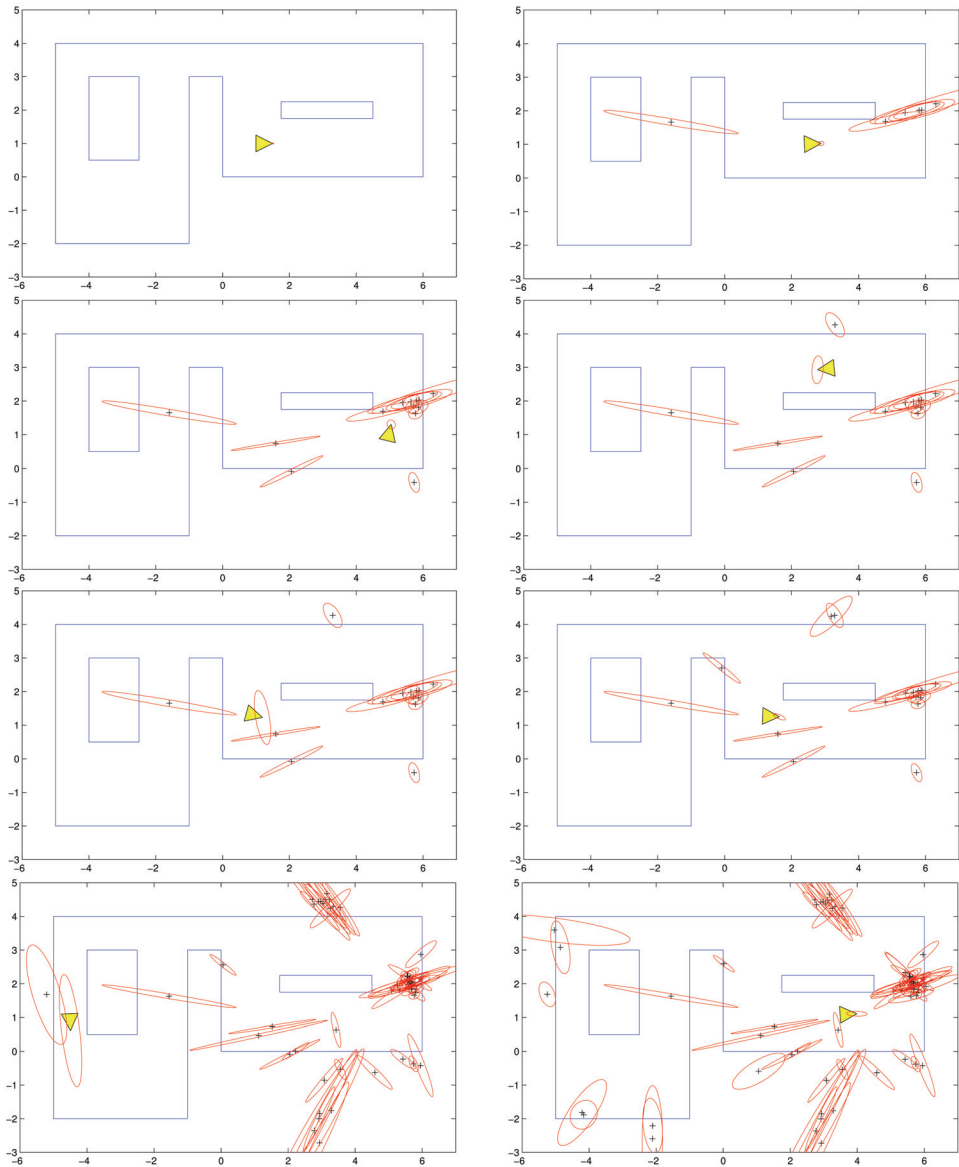
Fig. 12. The sensing steps 2, 5, 10, 18, 25, 26, 62 and 91 of a 95 step run with closing two loops using the geometrical reasoning approach with SIFT features.

Figure 12 shows the sensing steps 2, 5, 10, 18, 25, 26, 62 and 91 of a 95 step run with closing two loops. The first five images 2, 5, 10, 18 and 25 show landmark initializations and a growing robot pose uncertainty. The subsequent images 25 and 26 show the loop closure. Image 62 shows further explorations with growing uncertainty and image 95 shows the final

map after another closure with reduced uncertainties. At the end of this experiment, the 2-sigma values of the robot pose are $(0,28m \quad 0,1m \quad 0,06rad)$. Of course, the landmark variances are still much higher and require further observations.

The experiments prove that SIFT features can be used as natural landmarks in an indoor SLAM setting. The distinctive feature of this approach is that we only use 2D landmark poses instead of 3D poses. Thus, no methods to correct image distortion or perspective are needed. The various parameters are robust and can be set in wide ranges. Thus, they can be determined with low effort.

7. Bearing-only SLAM with SIFT feature patterns

The geometrical approach is not able to restrict the viewing angle in all cases. The selectiveness depends on the positioning of the landmark relative to its observation poses. In some situations the reduced selectiveness provably led to false landmark initializations. Thus, a different approach is introduced that does not anymore depend on the geometrical configuration of the landmark and observation poses.

7.1 SIFT feature patterns
This approach improves the robustness of the re-recognition of a SIFT feature by exploiting further SIFT features in its local neighbourhood. The first extension affects the feature extraction. For each SIFT feature, the *n* nearest SIFT features (in terms of Manhattan distance in image coordinates) are determined. Now, each SIFT feature is enriched by its *n* nearest neighbours.

The second modification is the replacement of the *geometrical reasoning* box (see figure 10). The task of this box is to validate the re-recognition of an uninitialized landmark. The re-recognition hypothesis is provided by the *matching descriptor* box. The validation now compares two SIFT features by including their neighbours. For each member of the set of neighbours of the first SIFT feature, a matching entry in the set of neighbours of the second SIFT feature is searched. For efficiency reasons, this is done by calculating the correlation coefficient. The match between both SIFT features is successful as soon as at least *m* neighbouring features show a correlation coefficient value greater than a threshold t_{cc}.

This approach improves the robustness of the re-recognition since further characteristics of the local neighbourhood of a SIFT feature are considered. However, we do not exploit any geometric relationship between the neighbouring SIFT features. Thus, the SIFT feature pattern does not form a certain texture and is thus largely independent of the observation distance and angle. This is of particular importance since we do not un-distort the omnicam images.

7.2 Experimental setup
The experimental setup is the same as in the previous section. We set *n = 5*, *m = 2* and $t_{cc} = 0.8$. However, the experiments have been performed in different rooms of the same building. Figure 13 shows the hallway with artificial light and a lab room with a large window front. Thus, the lighting conditions vary extremely over a run.

Fig. 13. Another part of our buildings with very different lighting conditions.

7.3 Experimental results

Figure 14 shows the sensing steps 5, 15, 25, 34, 45, 60, 75 and 92 of a 96 step run with closing a loop. The first seven images show landmark initializations and a growing robot pose uncertainty. Image 92 shows the robot and landmark uncertainties after loop closure. At the end of this experiment, the 2-sigma values of the robot pose are $\begin{pmatrix} 0,14m & 0,14m & 0,04rad \end{pmatrix}$. Again, the landmark variances are still much higher than the robot pose uncertainties and thus require further observations.

Using neighbouring SIFT features proved to be a suitable approach to get rid of geometrical assumptions while achieving the same overall performance. Thus, this more general approach should be preferred.
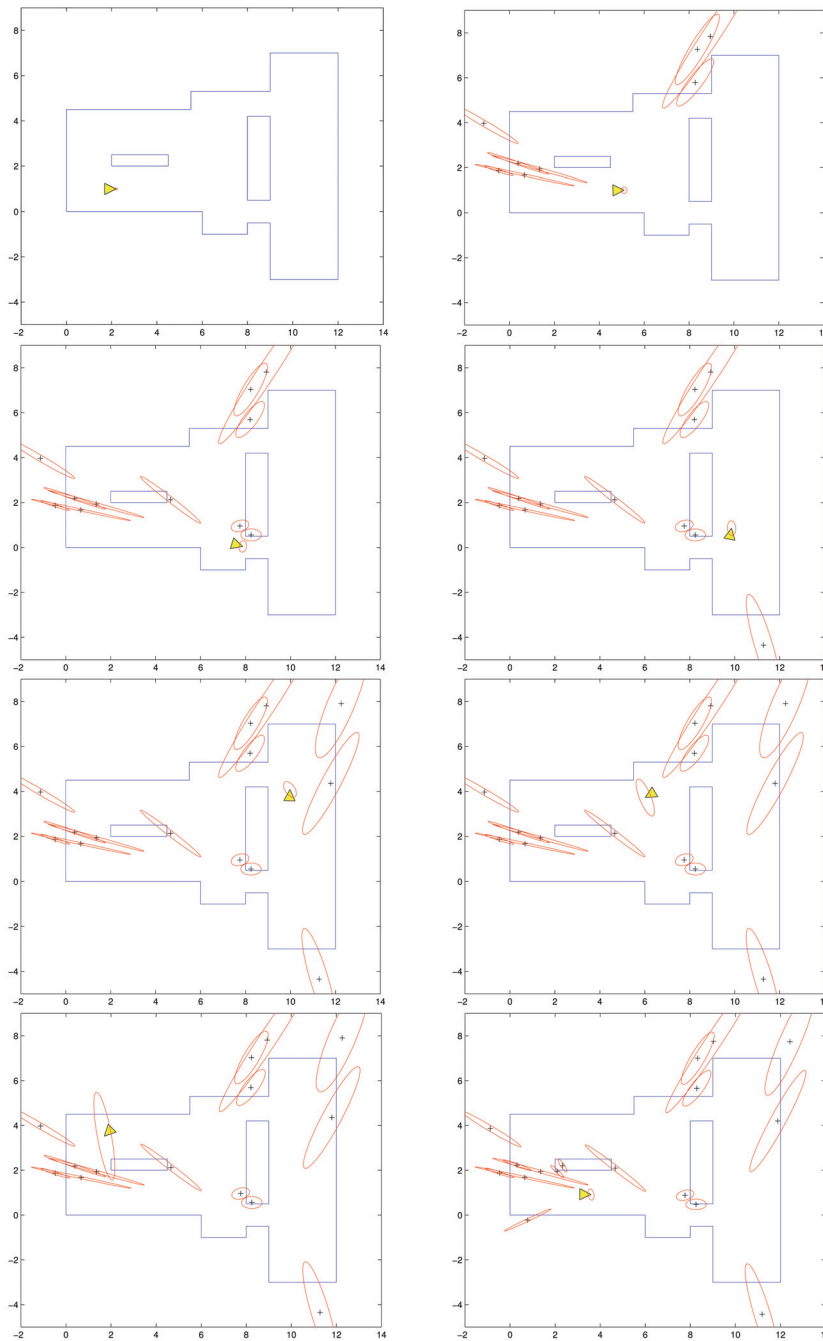
Fig. 14. The sensing steps 5, 15, 25, 34, 45, 60, 75 and 92 of a 96 step run with closing a loop. These results are based on the SIFT feature patterns.

8. Conclusion

The conducted experiments on a real platform prove that EKF-based bearing-only SLAM methods can be applied to features extracted from an omnicam image. In a first step, we successfully used artificial landmarks for the principal investigation of the performance of EKF-based bearing-only SLAM. However, service robotics applications ask for approaches that do not require any modifications of the environment. The next step was to introduce SIFT features into a bearing-only SLAM framework. We kept the idea of only estimating 2D poses of landmarks. This significantly reduces the overall complexity in terms of processing power since the state space of the Kalman filter is smaller and since the observation model is much simpler compared to 3D landmark poses. In particular the latest improvement exploiting the local neighbourhood of a SIFT feature shows stable performance in everyday indoor environments without requiring any modifications of the environment. The approach performed even under largely varying lighting conditions. Thus, the proposed approach successfully addresses the aspect of suitability for daily use as mandatory in service robotics.

9. References

Bailey, T. (2003). Constrainted Initialisation for Bearing-Only SLAM, *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1966-1971, Taipei, Taiwan

Bekris, K. E. et al (2006). Evaluation of algorithms for bearing-only SLAM. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1937-1943, Orlando, Florida

Cover, T. M. & Thomas, J. A. (1991). *Elements of Information Theory*, Wiley & Sons, Inc., ISBN 0-471-06259-6, USA

Davison, A. J. et al (2007). MonoSLAM: Real-Time Single Camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 29, 6, June 2007, pp. 1052-1067, ISSN 0162-8828

Fitzgibbons, T. & Nebot, E. (2002). Bearing-Only SLAM using Colour-based Feature Tracking. *Proceedings of the Australasian Conference on Robotics and Automation (ACRA)*, Auckland, November, 2002

Gil, A. et al (2006). Simultaneous Localization and Mapping in Indoor Environments using SIFT Features. *Proceedings of the IASTED Conference on Visualization, Imaging and Image Processing (VIIP)*, Palma de Mallorca, Spain, August, 2006, ACTA Press, Calgary

Hesch, J. & Trawny, N. (2005). Simultaneous Localization and Mapping using an Omni-Directional Camera. *Unpublished*

Hochdorfer, S. & Schlegel, C. (2007). Bearing-Only SLAM with an Omnicam – Robust Selection of SIFT Features for Service Robots. *Proceedings the 20th Fachgespräch Autonome Mobile Systeme (AMS)*, pp. 8-14, Springer

Intel (2008). Open Source Computer Vision Library.
http://www.intel.com/technology/computing/opencv/

Kwok, N. M. et al (2005). Bearing-Only SLAM Using a SPRT Based Gaussian Sum Filter. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1109-1114, Barcelona, Spain

Lemaire, T. et al (2005). A practical 3D Bearing-Only SLAM algorithm. *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pp. 2449-2454, Edmonton, Canada

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60, 2, pp. 91-110, ISSN 0920-5691

Ortega et al (2005). Delayed vs Undelayed Landmark Initialization for Bearing-Only SLAM. *ICRA 2005 Workshop W-M08: Simultaneous Localization and Mapping*, Barcelona, Spain

Rihan, J. (2005). An exploration of the SIFT operator. http://cms.brooks.ac.uk/research/ visiongroup/publications/msc/an_explaration_of_the_sift_operator_text.pdf

Roumeliotis, S. & Burdick, J. (2002). Stochastic cloning: A generalized framework for processing relative state measurements. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1788-1795, Washington, DC, USA

Schlegel, C. & Hochdorfer, S. (2005). Bearing-Only SLAM with an Omnicam – An Experimental Evaluation for Service Robotics Applications. *Proceedings of the 19th Fachgespräch Autonome Mobile Systeme (AMS)*, pp. 99-106, Springer

Sola, J. et al (2005). Undelayed Initialization in Bearing-Only SLAM. *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pp. 2499-2504, Edmonton, Canada

Tamimi, H. et al (2006). Loalization of mobile robots with omnidirectional vision using Particle Filter and iterative SIFT. *Robotics and Autonomous Systems*, 54, 9, pp. 758-765, ISSN 0921-8890

Thrun, S. et al (2005). *Probabilistic Robotics*. The MIT Press, ISBN 0-262-20162-3, Cambridge, Massachusetts

Wang, X. & Zhang, H. (2006). Good Image Features for Bearing-Only SLAM. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2576-2581, Bejing, China

Developing a Framework for Semi-Autonomous Control

Kai Wei Ong, Gerald Seet and Siang Kok Sim
Nanyang Technological University, School of Mechanical & Aerospace Engineering,
Robotics Research Centre
Singapore 639798

1. Introduction

Researchers and practitioners from the field of robotics and artificial intelligence (AI) have dedicated great effort to develop autonomous robotic system. The aim is to operate without any human assistance or intervention. In an unstructured and dynamic environment this is not readily achievable due to the high degree of complexity of perception and motion of the robots. For real-world applications, it is still desirable to have a human in the control loop for monitoring, detection of abnormalities and to intervene as necessary. In many critical operations full autonomy can be undesirable.

Such tasks require human attributes of perception (e.g. judgment), reasoning and control to ensure reliable operations. Although, robots do not possess the these human attributes, it is possible for the current-state-of robots to perform useful tasks and to provide appropriate assistance to the human to correct his control input errors by supporting perception and cooperative task execution. Systems which facilitate cooperation between robots and human are becoming a reality and are attracting increasing attention from researchers. In the context of human-robot cooperation (HRC), one of the research concerns is the design and development of flexible system architecture for incorporating their strengths based on their complementary capabilities and limitations. A well-known paradigm to facilitate such cooperation is that via the concept of semi-autonomy.

Although the concept of semi-autonomy is a widely adopted metaphor for developing human-robot system (HRS), there is no clear definition or agreement of how it should be represented. First, a formal representation of semi-autonomy is needed to identify and synthesise the key elements involved in the process of HRC. The purpose is to facilitate the development of a semi-autonomous control framework to seamless blend degree/level human control and robot autonomy at system-level. Second, there is a need to have a representation to address the role of semi-autonomy in decomposing and allocating tasks between humans and robots in a structured and systematic manner. This is essential in the initial design stage of HRS for providing a holistic basis of determining which system-level task should be performed by a human, by a robot or by a combination of both in accordance to their capabilities and limitations during task execution. A formalisation of semi-autonomy to address how task can be allocated between humans and robots is lacking in the current literature of robotics. This is because applications of semi-autonomy are normally

applied on an ad hoc basis, without a comprehensive formalism to address the problems of task allocation. Finally, without a formal representation, it is difficult to address, or discuss the research issues associated with the concept of semi-autonomy in a holistic manner.

Generally, the primary research issues of semi-autonomy can be summarised as follows:

i.   When human-robot share control, there are apparent dependencies between the actions taken by them and the actions available to another as they can be operating competitively or cooperatively. In this case, their actions can reinforce or interfere with each other. Here, the main concern is how to resolve their conflicting actions dynamically during task execution.

ii. When human-robot exchange or trade control, either the human or robot has full control at any one time, and over time this control responsibility is switched between them in accordance to the task at hand. The next issue involved is: who decides when the control is to be transferred, and how to ensure the transfer of control is exchanged smoothly.

iii. The above issues consider parallel and serial control separately. This is useful as it simplifies the types of human-robot cooperation strategies, explicitly. If both parallel and serial controls are to be applied in conjunction, it can be unclear how these control strategies can assist in the design and development of a cooperative HRS. In particular, issues relating to the consequences, requirements and form of semi-autonomous control arise. A proposed HRS architecture must not only facilitate the combination of humans and robots actions, it must also allow for the arbitration of their actions.

The aim of this paper is to address the concerns above and the system design issues raised by the concept of semi-autonomy. This work emphasises the importance of modelling a framework of semi-autonomy as a foundation for the development of cooperative HRS. This includes a discussion of how the formulated framework can be applied for implementation of an HRS. The key idea of the development of the semi-autonomous control architecture is based on the different human-robot roles, namely Master-Slave, Supervisor-Subordinate, Partner-Partner, Teacher-Learner and Fully Autonomous mode by the robot. Finally, using the implemented HRS, proof-of-concept experiments are conducted to assess how semi-autonomous control is achieved at system-level. This has been implemented on a wide range of mobile robots including the iRobot ATRV-Jr and the Argo an amphibious all terrain, off-road vehicle.

2. Related literature

The research on semi-autonomy relates to many topics in the literature of robotics. This paper focuses on the exploitation of semi-autonomous control strategy to facilitate effective human-robot interaction (HRI) to increase task performance and reduce errors. Generally, research in this domain is widely known as supervisory control (Sheridan, 1992), collaborative control (Fong, 2001), mixed-initiative control (Bruemmer et al., 2003), adjustable autonomy (Kortenkamp et al., 2000), sliding scale autonomy (Desai & Yanco, 2005) and dynamic autonomy (Goodrich, 2007). In the context of semi-autonomous control, HRI practitioners and researchers normally adopt certain interaction paradigms for human delegation of control to the robotics system, where the control can be taken back or shared dynamically (i.e. sharing and trading of control) during operation (Ong, 2006). Their interaction paradigm can be characterized by the interaction roles and relationships between humans and the robots in an HRS. This section provides a background on the interaction

roles that human and robot may play in an HRS. This is important in the design and development of semi-autonomous control architecture for HRI, because the human relationship with the robot can dictate the boundaries and constraints on the interactions between them. Subsequently, the idea of human-robot team (HRT) is introduced to differentiate the work here with other research work that also considers the use of different human-robot roles and relationships. This includes a discussion on the considerations of task allocation in developing a framework for semi-autonomous control, which is lacking in the literature of HRI.

2.1 Evolution of human-robot roles and relationships in robotics
2.1.1 Master-slave
According to Norman (2002), how human interacts with any technological system directly depends upon the human view of his relationship with that system. Historically, human normally recognizes himself as the *master* of the robot (Hancock, 1992). On the other hand, the robot is normally viewed as a *slave* of the human to service the needs and demands of the human. The history of modern robotics application based on this human-robot role and relationship began in the late 1940's, when the first *master-slave* telemanipulator system was developed in the Argonne National Laboratory for chemical and nuclear material handling (Vertut & Coiffet, 1985). With this system, the "slave" robot manipulator at the remote site reproduced exactly the motions imposed on the "master" handle by a human operator.

2.1.2 Supervisor-subordinate
With technological advancement comes robotic system of increasing capability, expanding the potential to facilitate and augment human work activities. It becomes critical to refine the roles and relationships that both human and robot can interact instead of just simply the "master-slave" relationship. In the late 1960's, many researchers and practitioners recognized this potential and started to consider how to improve the human relationship with the robot. Sheridan (1992) was one of the first to extend beyond the master-slave paradigm and formalized a new human-robot role and relationship called *supervisor-subordinate* relationship. This human-robot role and relationship is derived from the analogy between the human supervisor's interactions with human subordinates in a organization. A human supervisor gives directives that are understood and translated into detailed actions by human subordinates. In turn, human subordinates gather detailed information about results and present it in summary to the human supervisor, who must then infer and make decision for further actions. Sheridan stated that the human and the robot can also engage in such relationship but how "involved" the human supervisor becomes in the interaction process is determined by the autonomy of the subordinate robot. To date, the majority of research in robotics using this human-robot role and relationship has focused on telemanipulation for process control (Vertut & Coiffet, 1985) and also the teleoperation of mobile robots for space exploration (Pedersen et al., 2003), search and rescue (Casper & Murphy, 2003), military operation (Gage, 1985), automated security (Carroll et al., 2002).

2.1.3 Partner-partner
The master-slave and supervisor-subordinate relationship in HRS is hierarchical, with the human always acting as superior and the robot always subservient. In the early 1990s, researchers began to look into other human-robot role and relationship that is non-

hierarchical, where the nature of interaction between the human and the robot is liken to a *partner-partner* relationship. One of the first to design an HRS (i.e. a telemanipulation system) based on this perspective is from Lee (1993). According to Lee, the robot should not be viewed as a slave or subordinate of the human, but rather as an active partner of the human. In particular, taking the full advantage of the robot capabilities to let the robot supports the human perception, action and intention. This was purported by Fong (2001) that to develop a cooperative HRS, the human and the robot should work as partners to exchange ideas, to ask questions, and to resolve differences just as in human-human interaction. He stated that: "*instead of the human always being completely in charge, the robot should be more equal and can treat the human as a limited source of planning and information*". To date, the *partner-partner* human-robot role and relationship is widely adopted in the area of rehabilitation to let the robot work as a partner of the human so as to provide appropriate assistance to him (Martens et al., 2001; Wasson & Gunderson, 2001). One example in rehabilitation is from Bourhis & Agostini (1998) that uses the *supervisor-subordinate* paradigm in which the human works cooperatively with a robotics wheelchair. As compared to partner-partner paradigm, the interaction between the human and the robot in supervisor-subordinate relationship is mutually exclusive where either human or robot can take control at any one time. In Bourhis & Agostini work, the cooperation between the human and the robot is based on the idea that both the human and the robot can be supervisor of each other for overriding each other actions.

2.1.4 Teacher-learner

Teaching a robot through a human teacher has been widely studied since 1970s (Shimon, 1999). For example, humans have performed the role of a teacher in the domain of robot manipulators. In this domain, the robot as a learner normally learns its trajectory either through a teach-pendant or direct guidance through a sequence of operations given by a human. With recent advances in the theory and practice of robotics, this approach has been extended to allow the robot learner to learn from the interaction at the human teacher's high level of abstraction (e.g. by demonstration (Nicolescu & Matarić, 2001)). Through this interaction the robot learns up to the point at which the robot is able to carry out complex task and request appropriate help when required. Currently, this human-robot role and relationship is widely used in HRI to *enhance* the interaction between the human and the robot. This is because researchers in HRI recognize that effective HRI not only requires technological intelligence of the robot but also a "knowledge" transfer between the human and the robot during operation, so as to let the robot learns more difficult or poorly defined tasks (Haegele et al., 2001).

2.1.5 Fully autonomous

Since the days of the Stanford cart and the SRI's Shakey in the early 1970's, the goal of building fully autonomous system has been what researchers in robotics have aspired to achieve (Arkin, 1998). To date, cleaning robots (e.g. intelligent vacuum cleaner) are among the first members of the autonomous robot family to reach the marketplace with practical and economical solutions (Fiorini & Prassler, 2000). In such HRS configuration, once the human has specified a goal for the robot to achieve (e.g. "Clean Area A"), the robot operates independently. As the robot performs the task, the primary role of the human is to monitor the robot's execution.

2.2 Human-robot team

Each of the five human-robot roles and relationships discussed in Section 2.1 is important, since each stresses a different aspect of the interactions between the human and the robot. It will be beneficial if the advantages of these five human-robot roles and relationships are considered in the design and development of a semi-autonomous robotics system. This implies that instead of using only one fixed role and relationship, multiple interaction roles and relationships are envisaged to let human and robot to work as a team. From the HRS design perspective, it is an advantage as it decomposes the HRI problem into smaller sub-problems. System designers can now concentrate on each specified HRI role and design the appropriate functions. As a result, this provides an interactive HRS development that is based on what the human and the robot are best suited under different task situations and different levels of system autonomy. This idea is analogous to a human-human team where each team member usually does not engage in a single role when they work together. Within a human-human team, they normally engage different roles based on their task skills and their changes in interaction roles to meet new and unexpected challenges (Chang, 1995). The idea of getting a human and a robot to work as a team is not novel but the concept and implementation of multiple interaction roles and relationships and roles changing during operation is. The concept of Human-Robot Team (HRT) in the literature basically refers to human and robot adopting a partner-partner role and relationship described in Section 2.1.3. One notable exception is the work from Bruemmer et al. 2003. They explored the concept of HRT where each team member has the ability to assume initiative within a task. They state that to achieve this, both the human and the robot must have equal responsibility for performance of the task, but responsibility and authority for particular task elements shifts to the most appropriate member, be it the robot or the human. To facilitate, they suggested four roles for each member of a human-robot team, where either human or robot can take on the role of *supervisor* to direct the other team member to perform a high level task; *subordinate* role to perform high level task with less direct supervision by a supervisor; *equality of role* (i.e. partner), where each team member is wholly responsible for some aspect of the task; and as a subservient *tool* (i.e. slave), which performs a task with direct supervision by a supervisor.

The work by Bruemmer et al. (2003) is similar to the idea of HRT envisaged here, because both use multiple interaction roles and the need of role transitions. However, there are two differences. Firstly, the *type* of human-robot roles and relationships envisaged here not only considered human as supervisor and partner, it also considered human as master and teacher of a robot. Secondly, the work presented in this paper does not claim that the robot has the responsibility and authority to *direct* human in performing a task. In this work, human retains as the overall responsibilities of the outcome of the tasks undertaken by the robot and retains the authority corresponding with that responsibility even though the robot may be in the authority to guide certain aspect of the tasks (e.g. correct human control actions). This issue is discussed in Section 3.2.6.

In short, to achieve effective semi-autonomous control, the idea of HRT envisaged here requires both the human and the robot to engage in multiple interaction roles and role transitions during operation. The purpose is to let them perform different type of tasks and to meet new and unexpected challenges with the human maintained as the final authority over the robot. However, the HRS design considerations are no longer just on robotic development but rather on the complex interactive development in which both the human

and the robot work as a cohesive team. To facilitate, one important consideration is the allocation of tasks between humans and robots in the initial design stage of an HRS. This is further discussed in the following section.

2.3 Task allocation

Issues pertaining to the task allocation between human and robot do not gain much attention in the domain of HRI. To date, research effort in HRI mostly concentrates on the development of HRS architectures and the incorporation of human-robot interfaces as means for human to control the robot (Burke et al .2004). The consideration of task allocation in HRI is normally done in an informal manner. HRI designers normally make allocation decisions implicitly based on the unique advantages possessed by both the human and the robot (i.e. based on the "who does what" mandatory allocation decisions). For example, prior knowledge of a task, "common sense" in reasoning and perception are attributes that are possessed by humans but not by robots. On the other hand, rapid computation, mechanical power, diverse sensory modalities and the ability to work in hazardous environment are great advantages of robots that humans do not possess (Sheridan, 1997). Although the informal allocation of task can be used to make allocation decisions reasonably well, it may not be able to provide a judicious provisional allocation decisions; i.e., looking into: *when* problem arises during task execution and *how* might human and robot cooperate to resolve the problem. Successful resolution of task allocations often requires not only an understanding of fundamental issues concerning the capabilities and limitations of humans and robots, but also of a number of subtle considerations when both human and robot interact in performing an assigned task. This view is based upon the literature from human factors engineering for Human-Machine Interaction (HMI) and Human-Computer Interaction (HCI) in automated system (Sheridan, 1997; Hancock, 1992); such as flying an airplane (Inagaki, 2003; Billings, 1997), supervising a flexible manufacturing system (Tahboub, 2001; Hwang, 1984) or monitoring a nuclear power plant (Sheridan, 1992; Hamel, 1984).

To illustrate, consider the following situations. In performing an HRS task, a human may get tired or bored after long hours of operations, or a robot may fail to perform an allocated task due to a lack of prior task knowledge or sensors malfunction. If any of these situations happen and the HRS is designed solely based on mandatory allocated decisions that do not anticipate any interaction strategies that allow human to exchange control with the robot, the overall HRS performance may degrade or the HRS may breakdown physically. This implies that such decisions are not efficient and efficacious under certain situations. A successful task allocation scheme for semi-autonomous control must also include considerations of *timeliness* and *pragmatism* of the situation for making provisional task allocation decisions (e.g. when a robot fails to perform its allocated task during operation, how does human assists the robot; or when the human has problem performing a task, how does the robot provides appropriate assistance to the human).

2.3.1 Concept of task sharing and trading

In the context of a HRT (Section 2.2), when reallocating tasks adaptively between human and robot, it is vital to know that dynamic HRI role adjustment comes at a cost. This is because it may interrupt the ongoing dynamic task process of human and/or robot. A major challenge is to ensure that the task interaction between human and robot is continuous and

transparent so as to achieve seamless semi-autonomous control during task execution. As the task interaction between human and robot in an HRS may not be predictable and may occur in an arbitrary manner depending on the ongoing task performance and situation, it is not feasible to employ pre-programmed decision rules (e.g. as in adaptive task allocation for automated system (Hancock, 1992)) to trigger task reallocations dynamically based on pre-defined conditions. For successful accomplishment of a particular task in an HRS, both human and robot should cooperate through varying degree of human control, robot autonomy and appropriate human-robot communication when problem arises (Ong, 2006). This implies that task reallocation not only require to reallocate task responsibilities among human and robot (i.e. via role changing) but also to coordinate the interaction process between them. Examples include, resolve their conflicts, actions and intentions, arbitrate human/robot request for assistance, etc.

The decision to perform a task reallocation discussed above is invoked either by a human or a robot during task execution. By specifying task reallocation in this manner, the original definition of task reallocation based solely on the overall system task performance used in automated system may not be suitable (Sheridan, 1997; Hancock, 1992). Here, *task reallocation* is defined as the reallocation of a current desired input task that is allocated to the human and the robot with a completely new task specification. The conditions for task reallocation can be based on the ongoing task performance of the human and the robot, changes in task environmental or simply changes in the task plan that causes the current desired input task to be discarded. An approach useful for addressing this issue, i.e. making timeliness and pragmatic task allocation decisions is the concept of *task sharing and trading* proposed by Ong (2006). A human-robot cooperation concept that allows human and robot to work as a team by letting them contributes according to their degree/level of expertise in different task situations and demands. This concept not only considers how a robot might assist human but also how the human might assist the robot. Through this, a spectrum of cooperation strategies (Table 1) ranging from "no assistance provided to the human by the robot" to "no assistance provided to the robot by the human" can be envisaged to address contingencies that emerge when the human and the robot work together during task execution. Table 2 provides an abstract description of the prior task allocation in an HRS based on the considerations of capability of the performer but also on the timeliness and pragmatism of the situation. Consequently, this concept is adopted here for the formalization of the semi-autonomous control framework.

3. A framework for semi-autonomous control

Given Section 2.3.1, the interaction between a human and a robot in a semi-autonomous control system is in the context of a task. By *task* implies the required human's and robot's functions and the goals they are attempting to accomplish. This means that the "things" that the human and the robot can share and trade is placed within the context of a task. As posited by Ong (2006), the "things" that a human and a robot shared and traded is in the context of *human control*, *robot autonomy* and *information*, which constitute the key elements involved in the process of interactions between them. However, to consider how these elements constitute to the semi-autonomous control of a robot, there is a need to look into the basic activities within an HRS. This is further discussed in Section 3.1.

Human-Robot Cooperation Strategies	Characteristics
No assistance provided to human by robot.	This strategy is useful when human wants to perform a task by him/herself manually.
Robot assists human by extending his/her capability.	This strategy is useful to let the robot extends the human capability so that he/she can perform a task that is beyond his/her ability.
Robot assists human by dealing with different aspects of a task.	This strategy is useful to let the human and the robot cooperate to deal with mutually complementary parts of a task.
Robot assists human by providing appropriate support to the human.	This strategy is useful to let the robot provide active (i.e. constant or continuous) assistance to the human so as to reduce his/her burden or task demands.
Robot assists human by taking over the task from the human.	This strategy is useful to let the robot take over a task from the human when the human fails to perform a task or it can be the human who want the robot to perform the task by itself when he/she find that the robot has the ability to perform the task.
Human assists robot by providing appropriate support to the robot.	This strategy is useful to let human provide the require assistance to the robot when the human perceived that the task performance of the robot is not satisfactory or the robot request for human assistance.
Human assists robot by taking over the task from the robot.	This strategy is useful to let the human take over a task from the robot when the robot fails to perform the task.
No assistance provided to robot by human.	This strategy is useful to let the robot perform a task by itself with minimal or no human intervention.

Table 1. Different types of cooperation strategies between a human and a robot based on how the human and the robot might assist each other

Task Allocation	Determine By
Tasks that are best performed by the human.	"Who does what"
Tasks that require human-robot cooperation but may require the robot to assist the human.	Timeliness and pragmatism of the situation
Tasks that require human-robot cooperation but may require the human to assist the robot.	Timeliness and pragmatism of the situation
Tasks that are best performed by the robot.	"Who does what"

Table 2. A flexible prior task allocation based on "who does what" mandatory allocation and "when and how" provisional allocation decisions

3.1 Defining semi-autonomous control

According to Ong (2006), the basic task activities within an HRS may consist of:

- Desired task as input task, T_I
- Task allocated to the human, T_H
- Task allocated to the robot, T_R
- Task sharing and trading between the human and the robot, $T_{S\&T}$
- Task reallocation, T_{RE}

These basic activities may be related as shown in Fig. 1.
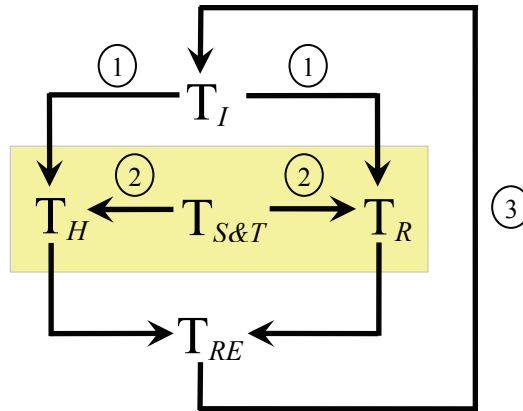
Figure 1: Activities within a Human-Robot System

In Fig. 1, it is suggested that there are three main paths to describe the activities within an HRS. The first path shown in Fig. 1 defines the input task (T_I) allocated to the human (T_H) and/or the robot (T_R). The second path defines task sharing and trading ($T_{S\&T}$) between the human and the robot. The third path represents task reallocation (T_{RE}) of the T_H and/or the T_R with a completely new T_I specification. Each of these five activities (i.e. T_I, T_H, T_R, $T_{S\&T}$ and T_{RE}) in Fig. 1 is further discussed in Section 3.1.1 to 3.1.5 respectively

3.1.1 Input task (T_I)

Given the task definition of a particular application task, such as large area surveillance, reconnaissance, objects transportation, objects manipulation, exploration of unknown environment, hazardous waste cleanup, to name a few. The next stage is to determine whether human, robot, or some combination of both should perform the T_I (i.e. prior task allocation) as follows. First, identify which tasks can *only* be allocated to either human (T_H, Section 3.1.2) or robot (T_R, Section 3.1.3) based on "who does what" mandatory allocation decisions. Subsequently, provisionally allocate tasks based on timeliness and pragmatic decisions, so as to take advantage of the symbiosis of the human and the robot capabilities to achieve task goals during task execution.

Generally, the considerations of making provisional task allocation based on the human and the robot capabilities can be characterised along several dimensions as follows:

- *Reasoning*: This includes attributes such as decision-making, task planning, situation understanding, and error detection and correction, to name a few. Consider an example

of a robot performing a navigation task of traversing from one location to another location and get "trap" in the dead end environment causing the robot unable to reach the specified location. To let the robot perform this navigation task successfully requires human assistance such as decision-making and situation understanding to guide the robot out of this situation.

- *Perception*: This includes attributes such as multi-modalities sensing, object recognition/discrimination/classification, to name a few. Consider a situation in which a robot attempting to move into a room and encounters door curtains directly in its path. Depending on the robot sensors suite, the robot's perception system may have difficulty determining if the curtains are obstacles or whether its path is truly blocked. Thus, the robot may not be able to traverse into the room. However, if human perceives through the video feedback, from the robot's camera, that the obstacles are only curtains, he/she can assist the robot by overriding the robot's perception system and command the robot to drive through the door.

- *Mobility*: This includes attributes such as traverse distance, mission duration, repetitive/unique mission, consequence of failure, moving with minimal disturbance to environment, and complexity of working environment (e.g. distribution of targets/obstacles, accessibility (e.g. small spaces), slope variability, soil/surface consistency, degree of uncertainty, etc.), to name a few. For example, these attributes are important considerations when a human is delegating a navigation task to a robot or providing appropriate assistance to the robot when its encounter problems (as discussed above).

- *Manipulability*: This includes attributes such as object shapes (standard/unique), repetitive/unique motion, precision/dexterity motion, consequence of failure, moving with minimal disturbance, and complexity of motion, to name a few. For instance, these attributes are important considerations when a human is delegating a manipulation task to a robot or providing appropriate assistance to the robot when its encounter problems while performing the task.

The discussion above has provided an abstract view of T_I and attributes for making provisional task allocation decisions during task execution. This is essential because it provides a basis for describing T_H and T_R in Section 3.1.2 and 3.1.3 respectively.

3.1.2 Task of human (T_H)

The primary T_H in an HRS is to *control* a robot to perform particular application tasks. In general, this encompasses the following functions that the human might require to perform:

- *Decision-making* - to decide whether the robot has the ability to perform the desired task. The considerations for making this allocation decision can be based on task attributes discussed in Section 3.1.1. For example, to control a robot to perform a navigation/manipulation task, the human must first determine whether the robot has the "physical" functions or *operating autonomy (i.e.* the basic physical operational capability*)*, such as mobility/manipulability to execute the desired task. Next, if the robot has the required operating autonomy, the human must decide whether the robot has the required *decisional autonomy* (i.e. level of competence/ intelligence imbued in a robot) to carry out the task by itself. If the human decides that the robot has the required autonomy, then the human will proceed to task planning (discussed below). However, if the human determines that the robot does not have the required

knowledge, he/she would need to imbue the robot with the necessary capabilities to perform the task

- *Task planning* - to schedule the task process and how it is carried out. For instance, setting goals, which the robot can comprehend.
- *Teaching* - to transfer task knowledge to the robot if the robot does not have the prior knowledge to perform the desired task.
- *Monitoring* - to ensure proper robot task execution and performance.
- *Intervention* - to provide appropriate assistance to the robot if any problems arise during task execution. Problems can include hardware failures, software failures, and human manual configuration requests for unscheduled support, to name a few.

The above are the conceivable tasks that can only be allocated to human based on the human roles in an HRS, e.g. as supervisor, partner or teacher of the robot as discussed in Section 2.1. The human roles in an HRS in turn determine how a robot might perform the HRS task. This is discussed below.

3.1.3 Task of robot (T_R)

The primary T_R in an HRS is to response to human control and in turn adapts its *autonomy* to perform the application tasks. Basically, this encompasses two basic functions that the robot requires to perform:

- *Physical task execution*: In general, how a robot might execute an HRS task depends on how human control the robot; i.e. based on the human-robot roles and relationships in an HRS as established in Section 2.1. For example, if the human adopts the master-slave paradigm to control the robot, then the robot will just mimic the human control actions exactly in performing the HRS task. On the other hand, if the human adopts the supervisor-subordinate paradigm to control the robot, then the robot will perform the HRS task planned by the human with minimum human intervention.
- *Feedback information*: To facilitate human monitoring and intervention of the robot task execution, the robot must feedback information to the human. This includes task information, environment information and the robot state information.

Section 3.1.2 and this section have provided an overview of T_H and T_R. This is essential because it provides a basis for describing the $T_{S\&T}$ between the human and the robot in the following section. Consequently, this will define the mode of operation for semi-autonomous control.

3.1.4 Task sharing and trading ($T_{S\&T}$) between human and robot

The concept of $T_{S\&T}$ (Ong 2006) introduced in Section 2.3.1 is based on how robot assists human – human assists robot (RAH-HAR). Within this paradigm, both the human and the robot may work as a team by engaging in different roles and relationships (Section 2.2) so as to exploit each other capabilities and/or compensate for the unique kinds of limitations of each other during task execution. Although the concept of $T_{S\&T}$ is able to describe the different types of cooperation strategies between a human and a robot based on how they might assist each other (as depicted in Table 1), it does not provide much insight into the design and development of a semi-autonomous control architecture given the interaction roles they might adopt during task execution. To facilitate, it is important to consider the dynamics of the $T_{S\&T}$ process so as to address the contingencies that arise when the human

and the robot work together during task execution. To address, there is a need to characterise the underlying basic elements that constitute the $T_{S\&T}$ between the human and the robot.

<u>A. Basic Elements of $T_{S\&T}$</u>

Given the key elements namely *human control*, *robot autonomy* and *information* involved in the process of interaction between a human and a robot (Ong 2006); it is defined here that for semi-autonomous control of a robot, the human must select the right control mode to share and trade control with the robot. On the other hand, the robot must adapt the right degree of autonomy so as to respond to the selected control mode (i.e. sharing and trading its autonomy with the human). This implies that "human control" and "robot autonomy" are placed within the context of a task collaboration for the human and the robot to accomplish their respective goals. By *task collaboration* means that both T_H and T_R are performed via appropriate human control, and varying level/degree of robot autonomy respectively. Thus, both "human control" and "robot autonomy" are the basic elements that a human and a robot can share and trade with each other respectively to achieve $T_{S\&T}$ (i.e. semi-autonomous control). In both cases, to perform the appropriate actions (i.e. changes in human control and robot autonomy), it invariably involves sharing of information. If the human and the robot have different perceptions regarding the shared information, they must trade information to clarify any doubt before actual actions can be performed. In short, information sharing and trading is to find out what the other party is doing, what the intention of the other party might be and to resolve any conflict if it arises during task execution. Hence, $T_{S\&T}$ is classified into *human control*, *robot autonomy* and *information* sharing and trading respectively to depict what can be shared and traded between a human and a robot during task execution.

The basic elements discussed above are important because they provide the basic constructs towards the characterisation of $T_{S\&T}$ in different HRI roles and relationships established in Section 2.1. The intention is for describing how semi-autonomous control can be achieved based on the concept of $T_{S\&T}$. This is discussed below.

<u>B. Characterisation of $T_{S\&T}$ in Different HRI Roles and Relationships</u>

The main corollary of the concept of HRT discussed in Section 2.2 is it requires the flexibility in HRI roles transition in order to let both human and robot work as a team. Given the HRI roles discussed in Section 2.1, the concern here is: how are these roles related to the process of $T_{S\&T}$ between human and robot. Here, it is posited that different kinds of HRI roles and relationships will inherently induce different phenomenon of $T_{S\&T}$, ranging from pure task decomposition to more complex task or sub task interactions. This is depicted in Fig. 2, in accordance to the basic elements, i.e. human control, robot autonomy and information.

As depicted in Fig. 2, each of the human-robot roles and relationships concentrates on different aspects of $T_{S\&T}$. Therefore, it will be advantageous if they can be integrated under the same framework to provide effective semi-autonomous control. This is achieved through the concept of the different roles and relationships of the human and the robot within an HRS is to provide multiple levels of human control and robot autonomy. In this context, each level of human control and robot autonomy will map in accordance to roles and relationships, such as those classified in Fig. 2. Issues pertaining to this topic are further discussed in Section 3.2.

3.1.5 Task reallocation (T_{RE})

As discussed in Section 2.3.1, T_{RE} is defined as the reallocation of a current desired input task that is allocated to the human and the robot with a completely new task specification. The consideration of T_{RE} as one of the activity within an HRS leads to the differentiation of two types of $T_{S\&T}$. To distinguish, the terms *local* and *global* are introduced. *Local* $T_{S\&T}$ is defined as the ongoing HRI in performing a desired input task with the aim of improving the current HRS task performance. If interaction roles transition occurs within the same task, it is considered as local $T_{S\&T}$. On the other hand, *global* $T_{S\&T}$ is defined as the reallocation of the desired input task that may involve HRI roles and relationships changes; where the change of role has completely different types of task specifications (e.g. change of role from supervisor-subordinate to master-slave, Fig. 2). This implies that a representation of semi-autonomy must take into the consideration of both local and global $T_{S\&T}$. so as to facilitate seamless human control changes and robot autonomy adjustment.

$$T_H \leftrightarrow T_{S\&T} \leftrightarrow T_R$$	Task sharing and trading ($T_{S\&T}$) between human and robot, extracted from Fig. 1. It is further break down into the basic elements (i.e. human control, robot autonomy and information) to describe the five roles and relationships below.
Master-Slave $$T_H \leftarrow I_S \leftarrow T_R$$	Trading is not involved. Sharing is in the form of information sharing (I_S) to extend human's sensing to a remote location. For example (Roman, 1993), a submersible robot operated via teleoperation extends the human's sensing (e.g. through video images relay from the robot on-board cameras) to a remote location so that he/she can work safely in the hazardous environment.
Supervisor-Subordinate	Control trading (C_T) from human to robot is to let human delegate control to robot. On the other hand, the robot needs to adjust its autonomy (i.e. autonomy trading from the robot to the human, A_T) to respond to the human control. Information sharing (I_S) from the robot to the human is for the human to monitor the robot task execution. However, if problem arises, both the human and the robot need to trade information (I_T) to clarify any doubts. An example to illustrate this is the Mars Rover developed to perform planet exploration, scientific measurement and data collection (Pedersen et al., 2003). The subordinate robot can achieve safe and effective navigation to the designated location and perform the desired tasks by augmenting its actions based on its local intelligence and the commands from the human supervisor. In this context, once the control is delegated to the robot (C_T) by the human, the robot will response to the delegation by adjusting its autonomy (A_T) to perform the task. Human typically assumes a monitoring role based on the information feedback by the robot (i.e. I_S). However, if the human has doubt regarding the task or robot status, he/she may exchange information (i.e. I_T) with the robot via dialog.

Figure 2. Phenomenon of sharing and trading induce by different human-robot roles and relationships described in Section 2.1

Teacher-Learner C_S/C_T I_S $T_H \quad T_R$ I_T A_S/A_T	Human teacher and robot learner shared and traded to transfer task knowledge (Klingspor et al. 1997) in the form of information sharing (I_S) and also information trading (I_T) to certain extent to seek understanding and clarification through inquiry by the learner (e.g. in a dialog manner). Depending on the teaching method, human may either teaches the robot via shared control (C_S) (Suomela & Halme, 2002) or traded control (C_T) (Papanikolopoulos & Khosla, 1992). In this context, the robot needs to share its autonomy (A_S) or traded its autonomy (A_T) to respond to the human control while learning the task. Shared control is normally used if the robot is required to perform the task along with the human and to provide appropriate assistance to the human.
Partner-Partner C_S I_S $T_H \quad T_R$ I_T A_S	Human control sharing (C_S), from human to robot, and robot autonomy sharing (A_S), from robot to human, are to let the robot provides "active" assistance to the human by supporting perception and cooperative task execution. Information sharing (I_S) between robot and human is for the human to monitor the robot task execution, and also for the robot to monitor the human control actions and intention so as to know when to provide appropriate assistance to the human. Similar to that of supervisor-subordinate, if problem arises, both human and robot need to trade information (I_T) to resolve conflicts and coordinate activities. An example to illustrate this comes from the rehabilitation application (e.g. Martens et al., 2001; Wasson & Gunderson 2001; Bourhis & Agostini, 1998). To provide assistance to the handicapped human user, the robot is normally equipped with high-level of artificial sensing and intelligent for navigation and reasoning. In this application, shared control is widely used to envisage a tight cooperation between the human and robot. Generally, both human and robot might cooperate as follows: the human brings his/her perception and interpretation of the close environment abilities to control the robot via control sharing (C_S). The robot, on the other hand, brings its highly autonomous mobility ability and its knowledge concerning a global model of the environment, moves in to response to the human control by adapting its autonomy (i.e. A_S) to prevent collision with any obstacles.
Fully Autonomous C_T $T_H \leftarrow I_S \quad T_R$ A_T	Sharing is in the form of information sharing (I_S) for human to monitor the robot task execution. Trading is in the form of human control trading (C_T) and robot autonomy trading (A_T) for starting and stopping the robot operation. Starting is normally performed by the human, but stopping can either be initiated by the human or the robot. A typical example is an intelligent vacuum cleaner (Fiorini & Prassler, 2000) that can operate without any human guidance and control once its start to operate.

Figure 2. Continue.

3.2 Discussion on framework formulation

The definition given in Section 3.1 indicates that semi-autonomous control must be represented with respect to a task, and that humans and robots must actively use its capabilities to pursue this underlying task via $T_{S\&T}$. In the context of $T_{S\&T}$, the aim of this section is to discuss how a framework formulated for semi-autonomy can be used to assist in the design and development of a cooperative HRS. To facilitate, a list of basic questions are considered as follows. Each of these questions is further discussed in Section 3.2.1 to 3.2.6 respectively.

- Why should human and robot share and trade?
- When should human and robot share and trade?
- How does human and robot know when to share and trade?
- How does human and robot share and trade?
- What triggers the change from sharing to trading (or vice versa)?
- Who is in charge of the sharing and trading process?

3.2.1 Why should human and robot share and trade?

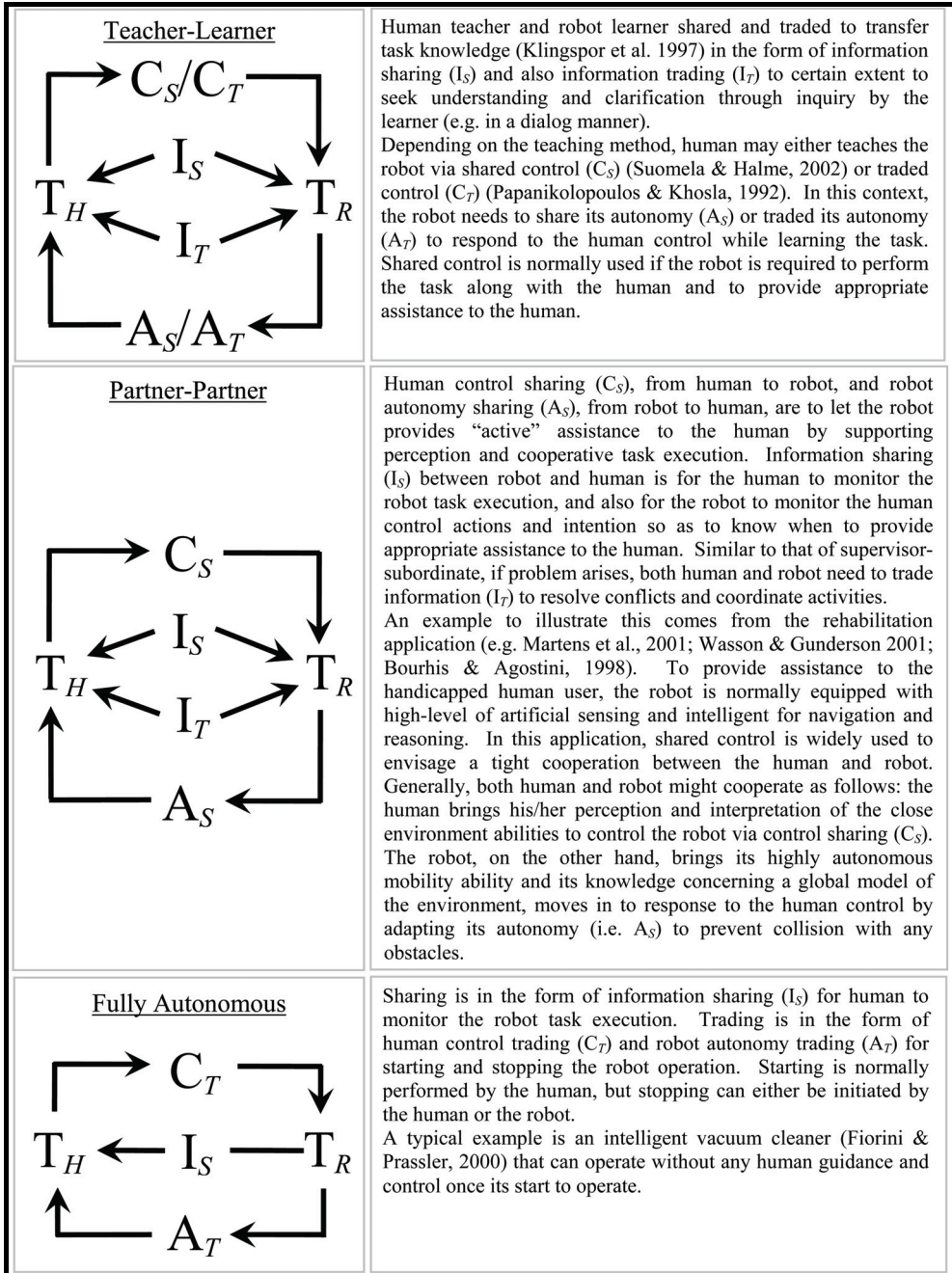In the context of performing an HRS task, $T_{S\&T}$ between a human and a robot is essential to let the human and the robot work together in different task situations and to ensure the overall system performance is achieved during task execution. By specifying in this manner, it does not mean the human and the robot share and trade only to deal with errors or contingency situations. They may even share and trade to provide appropriate assistance to each other during "normal operation", e.g., to let human assists a robot in object recognition, decision-making, etc. or to let a robot assists human in remote sensing such as obstacle avoidance and guidance. This implies that they may simply share and trade to strive for better system performance or to ensure that the system performance does not degrade when the other team mate is performing the HRS task. As such $T_{S\&T}$ process between the human and the robot may occur in an arbitrary manner, it is not feasible to pre-programme such $T_{S\&T}$ process. The "conditions" to invoke $T_{S\&T}$ must be based on the human and the robot current awareness and perception of the ongoing task execution. This topic is discussed below.

3.2.2 When should human and robot share and trade?

An intuitive view of looking into this question is based on the invocation of specific task events. It is possible to envisage a range of invocation events in accordance to the application tasks and invoke them based on the available information in the HRS. An advantage of this is that it directly addresses the possible sharing and trading strategies. From the extreme of initial task delegation to task completion, a spectrum of events can occur during task execution. Within this spectrum, three types of events to invoke or initiate a $T_{S\&T}$ process are distinguished. The first is termed *goal deviations* where the $T_{S\&T}$ process would be invoked by human intervention. This highlights how human assists' robot. The notion of *goal* here does not necessarily refer only to the goal of achieving a specific task, but also to the goal of attaining the overall task of the HRS. The word *deviation* refers to the departure from normal interactions between the robot and its task environment resulting in the robot being unable to achieve the goal. This also includes abnormalities arising during task execution. This may be due to either unforeseen changes in the working environment

that cannot be managed by the robot; where an undesirable functional mapping from perception to action causes the robot to "misbehave" (e.g. due to sensing failures).

The second event is *evolving situation* in which the $T_{S\&T}$ process would be invoked by the robot to veto human commands. This highlights how robot assists' human. The types of robot's veto actions can be loosely classified into *prevention* and *automatic correction*. Prevention implies that the robot will only impede the human actions but make no changes to it. The human is responsible for correcting his own actions. An example is when the robot simply stops its operation in a dangerous situation and provides the necessary feedback to the human to rectify his commands. On the other hand, automatic correction encompasses prevention and rectification of human commands simultaneously. Depending on the task situation, the robot may or may not inform the human how to correct his actions. For example, to prevent the human from driving into the side wall when teleoperating through a narrow corridor, the mobile robot maintains its orientation and constantly corrects the side distance with respect to the wall to align with it. In this case, the human may not be aware of this corrective action and he/she is able to drive the robot seamlessly through the corridor. According to Sheridan's (1997) ten-level formulation of system autonomy, both prevention and automatic correction are positioned at level seven or higher, i.e. the "system performs the task and necessarily informs the human what it did". This is because it is the robot that judges whether the situation is safe or unsafe, as the human is unable to judge.

Finally, the third event is when both the human and the robot *explicitly request assistance* from each other. In such an event, the $T_{S\&T}$ process between the two is mixed initiated, where each one strives to facilitate the individual activities in accordance to the task situation.

3.2.3 How does human and robot know when to share and trade?

Given the characterisation of $T_{S\&T}$ in different HRI roles and relationships in Fig. 2, a basic concern towards the achievement of seamless HRI is the need for each team-mate to be able to determine and be aware of and recognise the current capabilities/limitations of each other's during the process of $T_{S\&T}$. The ability for the human and the robot to recognise and identify when to share and trade control/autonomy/information so as to provide appropriate assistance to each other is essential in developing an effective HRT. To enable the robot to assist human, the robot needs to develop a model of the interaction process based upon readily available interaction cues from the human. This is to prevent any confusion during control mode transition. Just as robots need to build a model of the interaction process (and the operating environment) to ensure effective $T_{S\&T}$, it is also important for human to develop a mental model regarding the overall operation of an HRS (e.g. the operation procedures/process, robot capabilities, limitations, etc.), to operate the system smoothly.

A good guide in ensuring that the human is in effective command within a scope of responsibility is the principles from Billings (1997, pp. 39-48). For the human to be involved in the interaction process, he/she must be informed of the ongoing events (to provide as much information as the human needs from the robot to operate the system optimally). He/she must be able to monitor the robot or alternatively, other automated processes (i.e. information concerning the status and activities of the whole system) and be able to track/know the intent of the robot in the system. A good way to let human know the intention of the robot is to ensure that, the feedback from the robot to the human indicates

the "reason" for the invocation or initiation action during HRI. This implies that if the robot wants to override the human commands, the robot must provide clear indication for the human to know its intention to prevent any ambiguities. For example, during manual teleoperation, when the robot senses that it is in danger (e.g. colliding into an obstacle), the robot may stop the operation and send a feedback to warn the human in the form of a simple dialog.

3.2.4 How does human and robot share and trade?

As discussed in Section 2.3.1, the considerations of how does a human and a robot share and trade in response to changes in task situation or human/robot performance is based on the paradigm of RAH-HAR. Given the different types of cooperation strategies invoked by this paradigm (Table 1), the challenge is how $T_{S\&T}$ based on RAH-HAR capabilities can be envisaged. To address, consider the characterisation of $T_{S\&T}$ in different human-robot roles and relationships in Fig. 2. Based on this characterisation, Fig. 3 is presented to depict how these human-robot roles and relationships can be employed in designing a range of task interaction modes from "no assistance provided to the human by the robot" to "no assistance provided to the robot by the human" for the human and the robot to share and trade control. Consequently, this depicts how semi-autonomous control modes can be designed.
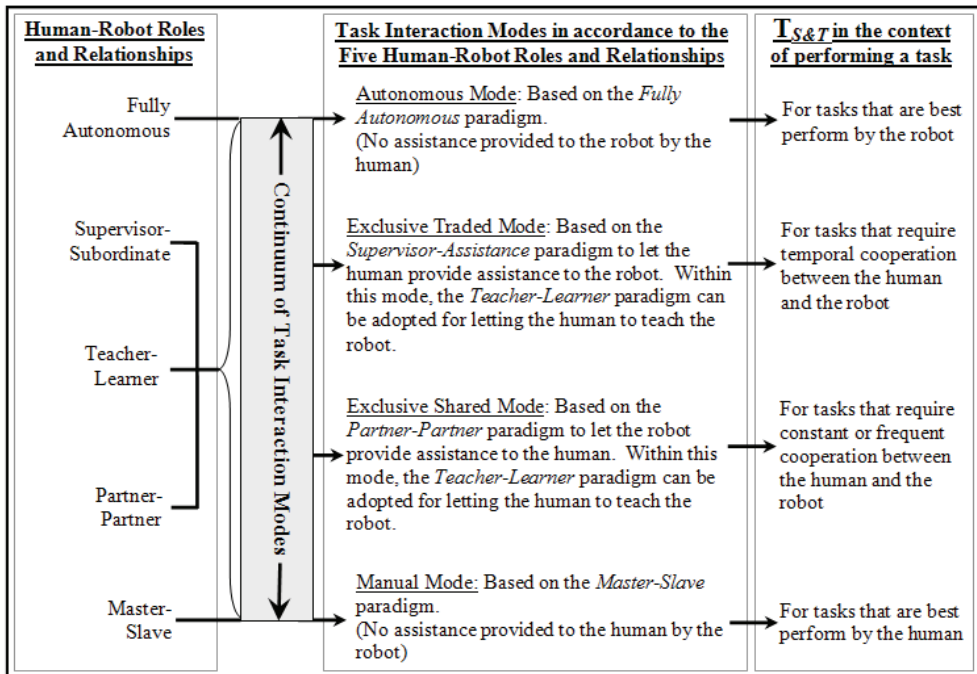

Figure 3. Range of task interaction modes in accordance to the characterisation of $T_{S\&T}$ in different human-robot roles and relationships depicted in Fig. 2

As shown in Fig. 3, to characterise the five human-robot roles and relationships, four discrete levels of interaction modes namely, *manual mode*, *exclusive shared mode*, *exclusive traded mode* and *autonomous mode* are defined. By defining sharing and trading in this manner does not mean that trading does not occur in sharing, or vice versa. Here, the term "exclusive" is used to highlight that the shared mode is exclusively envisaged to let the *robot assists human*, while the traded mode is exclusively envisaged to let the *human assists robot*. The reason of placing the shared mode below the traded mode is based on the degree of human control involvement. This implies that in exclusive shared mode, human is required to work together with the robot by providing continuous or intermittent control input during task execution. On the other hand, in exclusive traded mode, once the task is delegated to the robot, the human role is more of monitoring rather than of controlling or requires "close" human-robot cooperation, as compared to the exclusive shared mode. Therefore, the interactions between the human and the robot in this mode resemble the supervisor-subordinate paradigm instead of a partner-partner like interaction as in the exclusive shared mode.

A. Level and Degree of Task Interaction Modes

Fig. 3 depicts different *level* of human control and robot autonomy. This is for global $T_{S\&T}$ (Section 3.1.5), where each level represents different type of task specifications. To ensure that human maintains as the final authority over the robot (discussed in Section 3.2.6), level of task interaction mode transitions can only be performed by the human. Within each level, a degree of mixed-initiative invocation strategies (Section 3.2.2 and further discussed in Section 3.2.5) between human and robot can be enviaged to facilitate local $T_{S\&T}$ (Section 3.1.5). An approach to design invocation strategies is to establish a set of policies or rules to act as built–in contingencies with respect to a desired application. Based of these policies, the robot can adjust its degree of autonomy appropriately to response to the degree of human control changes or unforeseen circumstances during operation. To illustrate, Fig. 4 provides a basic idea of how this can be envisaged, by setting up a scale of operation modes, which enables the human to interact with the robot with different degree of human control involvement and degree of robot autonomy. The horizontal axis represents the degree of robot autonomy, while the vertical axis corresponds to the degree of human control involvement.

As shown in Fig. 4, the robot autonomy axis is inversely proportional to the human control involvement axis. Within these two axes, the manual control mode is situated at the bottom-left extreme, while the autonomous control mode is located at the top-right extreme. Between these two extremes is the continuum of semi-autonomous control. Within this continuum, varying degrees of sharing and trading control can be achieved based on varying nested ranges of action as proposed by Bradshaw et al. (2002). They are: *possible actions*, *independently achievable actions*, *achievable actions*, *permitted actions* and *obligated actions*. Based on these five actions, constraints can be imposed so as to govern the degree of robot autonomy (e.g. defined using a set of perception-action units) within each level of task interaction modes (Fig. 3). In this manner, human can establish preferences for the autonomy strategy the robot should take by changing or creating new rules. Consequently, rules can be designed to establish conditions where a robot must ask for permission to perform an action or seek advice from the human about a decision that must be made during task execution in accordance to the capabilities of the robot.

Given the range of task interaction modes defined in Fig. 3 and Fig. 4, to facilitate semi-autonomous control, concern pertaining to what triggers the change from sharing to trading (or trading to sharing) must be addressed. This is discussed in the following section.
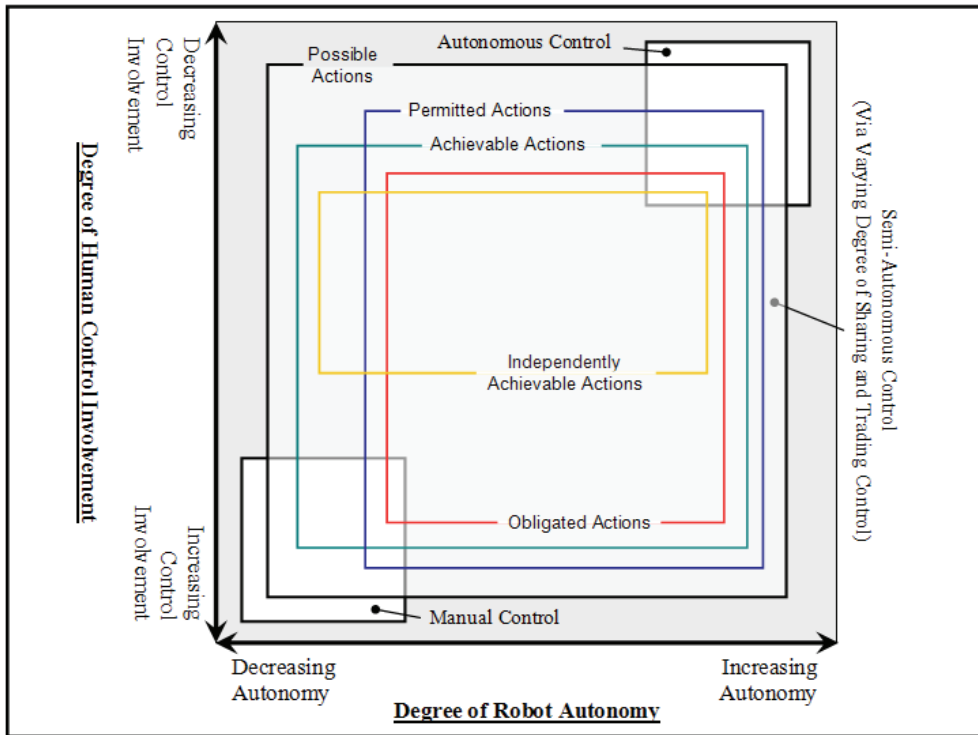
Figure 4: Control modes based on robot autonomy and human control involvement in accordance with varying nested ranges of action of robot

3.2.5 What triggers the change from sharing to trading (or trading to sharing)?

In accordance to the range of task interaction modes defined in Fig. 3, a transition from sharing to trading (or vice versa) may involve in a total new task specifications (i.e. global $T_{S\&T}$) or within the context of a same task specifications (i.e. local $T_{S\&T}$). Both global and local $T_{S\&T}$ are defined in Section 3.1.5 respectively. To discuss what triggers the change from sharing to trading (or vice versa) in both types of $T_{S\&T}$ process, two types of trigger, namely *mandatory* and *provisional* for global $T_{S\&T}$ and local $T_{S\&T}$ respectively are distinguished.

- *Mandatory Triggers* are invoked when there is a change of task plan by the human due to environmental constraints that may require different control strategy (e.g. from shared control to traded control), when the robot has completed performing a task leading to the specification of a new task that may required different control strategy or when task performance of the robot is perceived to be unsatisfactory resulting in human to use other control strategy, to name a few.
- *Provisional Triggers* are invoked when the human or the robot wants to assist each other to strive for better task performance. In this context, a change from sharing to trading can be viewed from a change from the robot assisting human to human assisting the robot, or vice versa in the case of from trading to sharing.

3.2.6 Who is in charge of the sharing and trading process?

The paradigm of RAH-HAR requires that either the human or the robot be exclusively in charge of the operations during $T_{S\&T}$. This means that the robot may be in authority to lead certain aspect of the tasks. This may conflict with the principle of human-centred automation, which emphasises that the human must be maintained as the final authority over the robot. As this issue of authority is situation dependent, one way to overcome this is to place the responsibility of that $T_{S\&T}$ process to the human. This means that the human retains as the overall responsibilities of the outcome of the tasks undertaken by the robot and retains the final authority corresponding with that responsibility. To facilitate, apart from giving the flexibility to delegate tasks to the robot, the human need to receive feedback (i.e. what the human should be told by the robot) on the robot intention and performance (e.g. in term of the time to achieve the goal, number of mistakes its make, etc.) before the authority is handed over to the robot. To delegate tasks flexibly, the human must be able to vary the level of interaction with specific tasks to ensure that the overall HRS performance does not degrade. Ideally, the task delegation and the feedback provided should be at various levels of detail, and with various constraints, stipulations, contingencies, and alternatives.

4. Application of semi-autonomous control in telerobotics

In Section 3, a framework for semi-autonomous control had been established to provide a basis for the design and development of a cooperative HRS. The type of HRS addressed here is a telerobotics system, where the robot is not directly teleoperated throughout the complete work cycles, but can operate in continuous manual, semi-autonomous or autonomous modes depending on the situation context (Ong et al., 2008). The aim of this section is to show how this framework can be applied in the modelling and implementation of such system.

4.1 Modelling of a telerobotics system

In the formulated semi-autonomous control framework, the first phase towards the development of a telerobotics framework is the *application requirements and analysis phase*. The emphasis of this phase is to identify and characterise the desired application tasks for task allocation between humans and robots. Given the desired inputs tasks for allocation, the second phase towards the telerobotics framework development is the *human and robot integration phase*. The primary approach of integrating human and robot is via the concept of $T_{S\&T}$, in accordance to how human and robot assist each other. This section discusses these two phases; presented in Section 4.1.1 and 4.1.2 respectively. The final phase which is the implementation of the telerobotics system is discussed in Section 4.2. Subsequently, proof-of-concept experiments are presented in Section 4.3 to illustrate the concept of semi-autonomous control.

To provide an overview of how the first phase and second phase described above are involved in the development of the sharing and trading telerobotics framework, a conceptual structure of an HRS is depicted in Fig. 5.

4.1.1 Application requirements and analysis phase

The first component in Fig. 5 is the task definition of a particular application goals and requirements which involves the translation of a target application goals and requirements

into a "task model" that defines how a telerobotics system will meet those goals and requirements. This includes conducting studies to assess the general constraints of the potential technology available (e.g. different types of sensing devices) and environment constraints (e.g. accessibility, type of terrain) that may be useful for the telerobotics system under design. For this research, the type of applications considered are those based on mobile telerobotics concept, such as planetary exploration, search and rescue, military operation, automated security, to name a few. This implies that the characteristic of the desired input task (i.e. T_I, Fig. 1) of such applications is to command a mobile robot (by a human) to move from one location to another location while performing tasks such as surveillance, reconnaissance, objects transportation, etc.
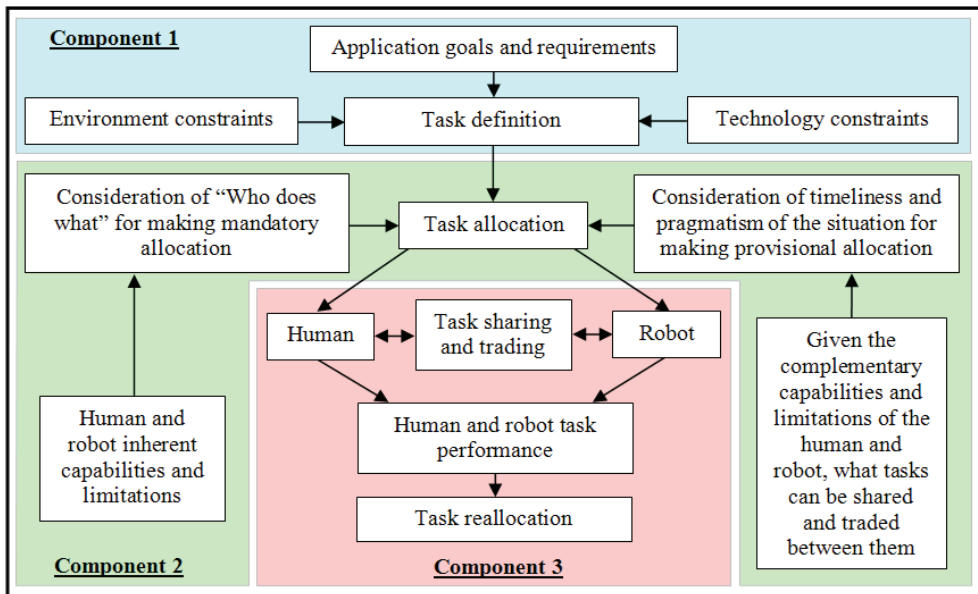

Figure 5. A conceptual structure of an HRS

4.1.2 Human and robot integration phase

The second component in Fig. 5 is the allocation of the desired input tasks to human (i.e. T_H, Fig. 1) and robot (i.e. T_R, Fig. 1). Possible analyses to the type of tasks that can only be allocated to human and robot (i.e. "who does what") are discussed in Section 3.1.2 and 3.1.3 respectively. The difficult part is to consider tasks that can be performed by both human and robot. For example, the T_I discussed in Section 4.1.1 (i.e. *moves from one location to another location*) implies three fundamental functions: *path planning*, *navigation* and *localisation*. Both human through control of robot by teleoperation and robot have the capabilities to perform these functions. The main consideration is who should perform these functions or is it possible for human and robot to cooperate to perform these functions. In accordance to the paradigm of RAH-HAR (Section 3.1.4), this is not a problem because this paradigm takes into the consideration of timeliness and pragmatic allocation decisions for resolving conflicts/problems arising between human and robot. Therefore, it allows human and robot

to perform the same function. The advantage of allowing human and robot perform the same function is that they can assist each other by taking over each other task completely when the other team member has problem performing the task. To achieve such a complementary and redundancy strategy, the approach by RAH-HAR is to develop a range of task interaction modes for human and robot to assist each other in different situations as described in Section 3.2.4. The four main task interaction modes are *manual mode*, *exclusive shared mode*, *exclusive traded mode* and *autonomous mode* as shown in Fig. 3. The two extreme modes (i.e., manual and autonomous) are independent to each other. They are also independent to the exclusive shared and traded modes. On the other hand, the dependency of the exclusive shared and traded modes depends on how human use the interaction modes to perform the application task. For example, to command a robot to a desired location based on environment "landmarks", the robot must first learn to recognise the landmarks so as to perform this navigation task. In this situation, the exclusive traded mode is dependent on the exclusive shared mode. This is because this mode facilitates robot learning of the environmental features to the desired location via teleoperation by the human.

A. Robot Capabilities

It is reasonable to argue that a human is currently the most valuable agent for linking information and action. Therefore, in an HRS, the intelligence, knowledge, skill and imagination of the human must be fully utilised. On the other hand, robot itself is a "passive component", its' level/degree of autonomy depends on the respective robot designer or developer. As highlighted in Section 2.2, for a human-robot team, the considerations are no longer just on robotic development but rather more complex interactive development in which both the human and the robot exist as a cohesive team. Therefore, for the robot to assume appropriate roles to work with the human counterpart, the robot must have the necessary capabilities. In accordance with the research in robotics (Arkin, 1998) and AI (Russell & Norvig, 2002), the capabilities required by a robot are numerous but may classify along four dimensions. They are *reasoning*, *perception, action* and *behaviours* (i.e. basic surviving abilities and task-oriented functions) as depicted in Fig. 6.
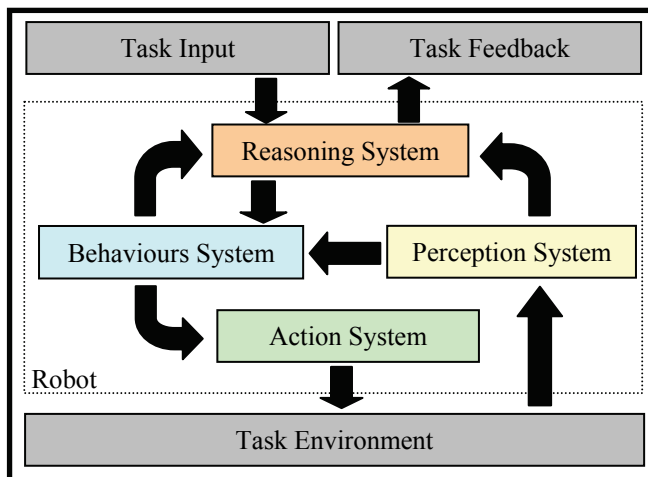

Figure 6. The Relationship between different capabilities of a robot

Reasoning: A robot must have the ability to reason so as to perform tasks delegated by human. In AI, the term "reasoning" is generally used to cover any process by which conclusions are reached (Russell & Norvig, 2002, pp. 163). By specifying in this manner, reasoning can be used for a variety of purposes, e.g. to plan, to learn, to make decision, etc. In robotics, *planning* and *learning* have been identified as the two most fundamental intelligent capabilities a robot must be imbued with so as to build a "fully autonomous robot" that can act without external human intervention (Arkin 1998). However, due to the fact that a robot must work in a real-world environment that is continuous, dynamic, unpredictable (at least from the robot's point of view), and so forth, the goal of building such a fully autonomous robot has not yet been achieved. In the context of planning and learning in an HRS, human may assist the robot in performing these functions. For example, the human can assist in solving "nontrivial" problems by decomposing the problem that must be solved into smaller pieces and let the robot solve those pieces separately. In the case of learning, human may teach the robot to perform a particular task via demonstration (Nicolescu & Matarić, 2001). However, apart from learning from the human, the robot must also learn from its task environment (e.g. through assimilation of experience) so as to sustain itself over extended periods of time in a continuous, dynamic and unpredictable environment when performing a task. A good discussion of different robot learning approaches and considerations can be found in (Sim et al. 2003). However, to plan or learn, the robot must have a perception system to capture incoming data and an action system to act in its task environment. This is discussed below.

Perception: A perception system of a robot is in the front line against the dynamic external world, having the function as the only input channel delivering new data captured from outside world (i.e. task environment) to an internal system (e.g. the software agents) of the robot. The perceptual system can play a role of monitoring actions by identifying the divergence between observed state and expected state. This action monitoring to ensure correct response against a current situation by examining an action in-situ should be an indispensable capability particularly in dynamic uncertain environments, in which the external world may change. Therefore, competent perceptual system would significantly contribute to improve the autonomy of a robot. Specifically, this is essential for the robot to monitor human control behaviours so as to provide appropriate assistance to the human. Through this, task performance can be improved (Ong et al. 2008).

Action: An action system of a robot is the only output channel to influence the external world with the results of deliberation (i.e. reasoning) from the internal system. Taking appropriate action in a given moment is one of the fundamental capabilities for an intelligent robot. How long a robot deliberates to find a sequence of actions to attain a goal is a critical issue particularly in real-time task domain. Sometimes it is rational to take actions reactively without planning if the impacts of those actions are minor to the whole accomplishment for the goal and easy to invoke, in emergent situations that require immediate actions.

Behaviours: Finally, a robot must have a set of behaviours to perform particular application tasks. This can range from basic behaviours such as point-to-point movement, collision prevention, obstacle avoidance to more complex task behaviours such as motion detection, object recognition, object manipulation, localisation (i.e. determining robot own location), map building, to name a few.

4.2 Implementation

Based on the system framework established in Section 4.1, this section outlines the design approach and described the system architecture of the telerobotics system. Implementations have been made on the Real World Interface (RWI) model ATRV-Mini™ and the ATRV-Jr™, the Segway-RMP™ (200) and the gasoline engine ARGO™-Vanguard 2, as depicted in Fig. 7.
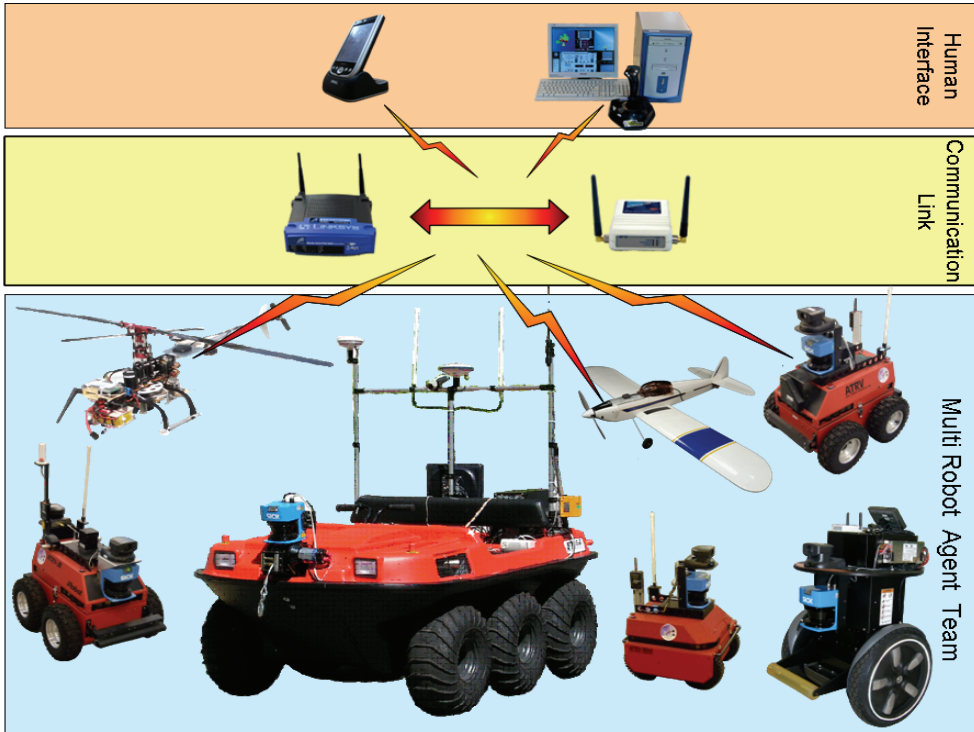

Figure 7. An overview of the telerobotics system setup

The technological considerations are that a telerobotics vehicle requires the following basic components to facilitate human control. Firstly, it must have adequate sensors to perform the desired tasks, e.g. navigation. For example, range sensors for obstacles avoidance, detection and location sensors to determine its location. Secondly, it must have some means of communication transceivers to communicate with the human control interface. Finally, the robot must have embedded computation and program storage for local control systems. That is, a "computer ready" mechatronic system for automated control of the drive, engine and brake system. This is to facilitate the interpretation of command from the human control interface and translate it into signals for actuation. Fig. 8 provides an overview of one of our implemented robotic vehicle, a COTS off-road all-terrain utility vehicle, the ARGO™ that is equipped with the components described above. The main characteristic of this vehicle is its ability to travel on both land and water. This amphibious vehicle is powered by a 4-cycle overhead valve V-Twin gasoline engine with electronic ignition. It can travel at a cruising speed of 35 km/h on land, and 3 km/h on water.
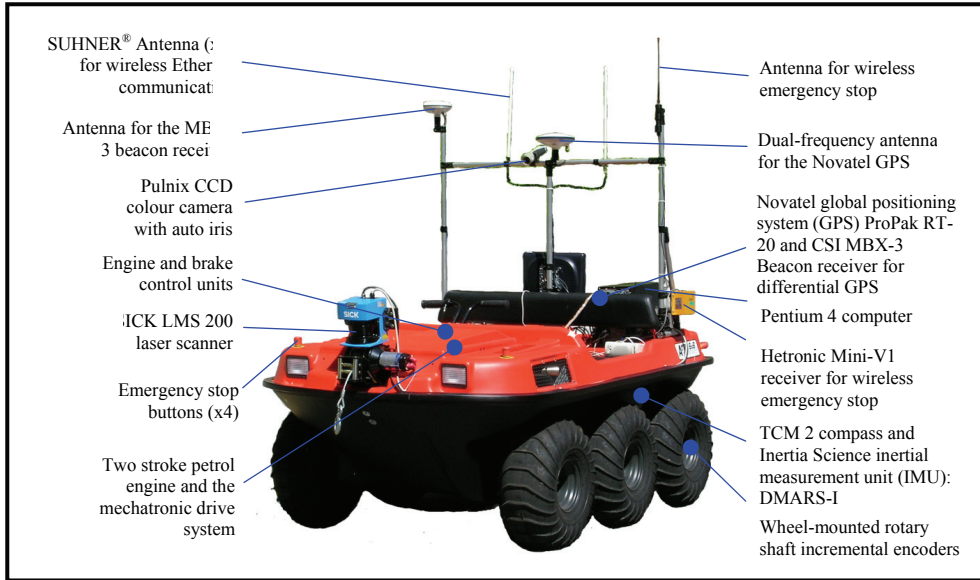
SUHNER® Antenna (s
for wireless Ether
communicat

Antenna for the ME
3 beacon recei

Pulnix CCD
colour camera
with auto iris

Engine and brake
control units

ICK LMS 200
laser scanner

Emergency stop
buttons (x4)

Two stroke petrol
engine and the
mechatronic drive
system

Antenna for wireless
emergency stop

Dual-frequency antenna
for the Novatel GPS

Novatel global positioning
system (GPS) ProPak RT-
20 and CSI MBX-3
Beacon receiver for
differential GPS

Pentium 4 computer

Hetronic Mini-V1
receiver for wireless
emergency stop

TCM 2 compass and
Inertia Science inertial
measurement unit (IMU):
DMARS-I

Wheel-mounted rotary
shaft incremental encoders

Figure 8. The customised ARGO™ - Vanguard 2 (6x6) amphibious robot

4.2.1 Telerobotics system architecture

The principal part of a robot is the control architecture; it is in charge of all the possible movements, coordination and actions of the robot in order to achieve its goal. To facilitate, different control strategies for robotic control have been proposed over the years. These strategies can be characterised from deliberative control, which is highly influenced by the AI research in the 1970s to reactive control, revolutionised by Brooks (1989) in the 1980s. Deliberative control is purely symbolic and representation-dependent but with high-level of intelligence that often requires strong assumption about the world model the robot is in. This control approach is suitable for structured and highly predictable environment. However, in complex, non-structured and changing environments, serious problems (such as computation, real-time processing, symbol grounding, etc.) emerge while trying to maintain an accurate world model. On the other hand, reactive control, which is highly reflexive and representation-free, couples the perception and action tightly, typically in the context of motor behaviours, to produce timely robotic response in dynamic and unstructured worlds (Arkin, 1998). However, this approach has its own difficulty, i.e. the problem of designing a set of perception-action processes or behaviours in order to plan and to specify high-level goals. Consequently, a hybrid control approach based on the advantages of both the traditional deliberative and reactive approaches has since been the de facto standard in robotics research since 1990s (Arkin, 1998). This approach is widely known as three-layer architecture (Gat, 1998), as it requires a sequencing layer (i.e. the middle layer) to coordinate the processes between the deliberative and the reactive layer. Although this system allows the three layers to run independently, it is considered centralised because it requires all the three layers to work together. The telerobotics system architecture depicts in Fig. 9 adopts this approach. The control architecture is hierarchical. It consists of a low-lever controller (i.e. a mechatronic actuation system) to provide functions

for automated control of throttle, steering and brakes, and a high-lever controller that determines the desired velocity and steering direction based on a given goal (e.g. go to a location, follow another vehicle, etc.).

The telerobotics system presented in Fig. 8 is complex and composed of a number of subsystems. Each of these subsystems presents its own challenges. In order to provide a roadmap of how the telerobotics system is developed, the subsystems are classified into:

a) *Robot Hardware* – that consists of all the actuators, sensors and communication devices.
b) *Navigation* – that concerns the movement of the mobile robot.
c) *Localisation* – that estimates the position and orientation of the mobile robot.
d) *Planning* – that describes the planning of actions to be performed by the mobile robot.
e) *Interfaces* – that provides the necessary mechanisms for the human to monitor and control the mobile robot.

Schematically, the components of the telerobotics system architecture in terms of these five subsystems are organised into five levels as depicted in Fig. 9. They are classified based on the robot capabilities discussed in Section 4.1.2-A. In accordance to the three layer architecture, the robot hardware and the navigation behavioural system belong to the reactive layer. The localisation and the navigation behavioural/task coordination system belong to the sequencing layer. The planning and the interfaces level belongs to the deliberative layer. Current implementation does not employ a deliberative planner; instead the responsibility of performing its task is given to the human. This implies that the human is not only responsible for specifying the overall system goal but is also responsible for planning, global world modelling, etc. On the other hand, the mobile robot is concerned with lower level goal, for example to find the "best" path via the path planner (e.g. the shortest available path) and responds to external stimuli reactively while moving towards the goals set by the human. Hence, the system architecture can be categorised under hybrid intelligence.

The decomposition of the system architecture into its subsystems encourages modularity in system development. The modular approach allows easy replacement of components in the event of component failures. It also allows experimentation of components with similar functions, an important consideration in the system development.

4.2.2 Design and development of task interaction modes components

As shown in Section 4.1.2, to facilitate semi-autonomous control, multiple/different methods for sensing, navigation, localisation and planning are implemented in a modular manner. This is essential because without these methods, there can be no rendering of assistance between robot and human. Section 4.2.1 only presents the overall implementation and configuration of the telerobotics system; this section presents the achievement of semi-autonomous control via $T_{S\&T}$ between human and robot at system-level.

Given the task interaction modes characterised in Section 3.2.4 (Fig. 3), to understand how these task interaction modes or their sub-modes transit seamlessly at system-level, there is a need to look into the interaction modes components and how they are coordinated. Modes transition which is situation dependent can be initiated by human, by robot or by both the human and the robot (i.e. mixed initiation). The strategies to effect mode transitions (which also provide pre-conditions for modes coordination) are addressed in Section 3.2.2. Two important attributes involved in modes transition is monitoring and intervention. To define the different levels of human and robot intervention, the classic Rasmussen's (1983) Skill-

Rule-Knowledge (SRK) control behaviour model is adopted. The overall concern here is to discuss the components for facilitating modes transition. The components are classified into: *human control*, *robot autonomy*, and *feedback*. The design of these components is discussed below. This includes the discussion of the coordination of these task interaction mode components.
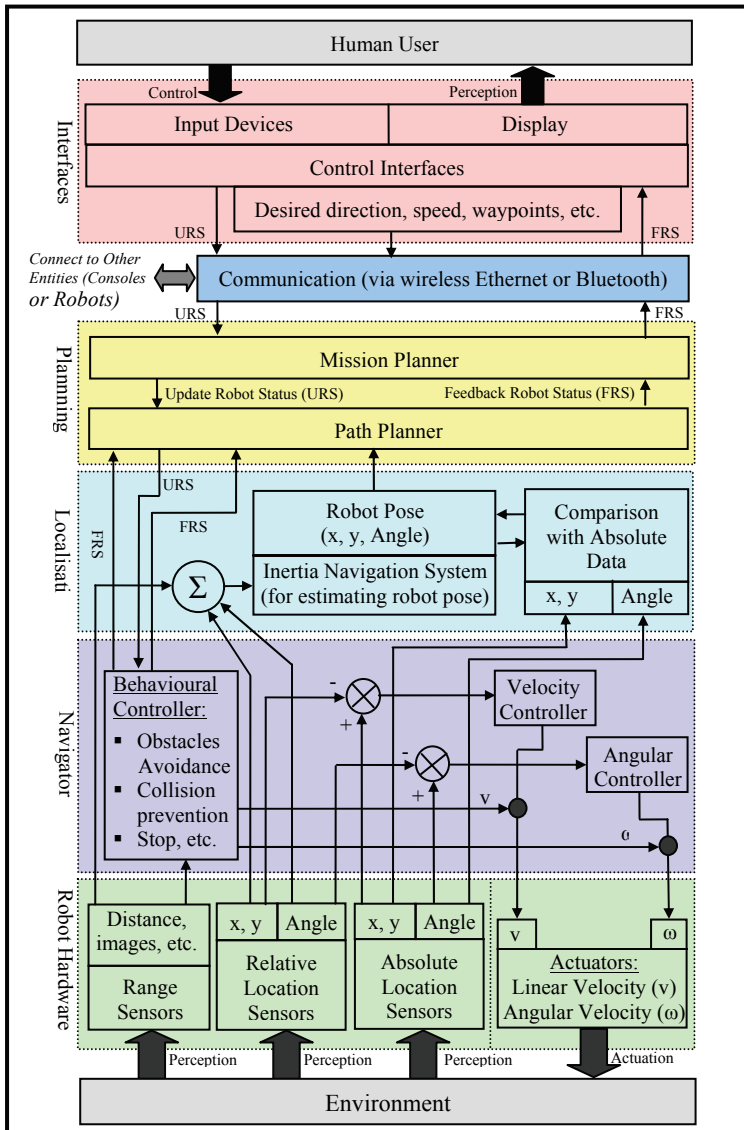

Figure 9. An overview of the telerobotics system architecture

A. Human Control Components

These components are responsible for integrating the human control input into the robotics system. The primary function is to allow human intervention of robotics operations at different levels. In accordance to the Rasmussen's SRK model, the implementation of the human control intervention is classified into three levels as follows:

Level 1. *Skill-based control behaviour*: Piloting or co-piloting of the robot which uses the human perception-motor coordination. This is applied in both the manual mode and exclusive shared mode via continuous human input. The human inputs at this level are rate control of the robot translation and rotation speed.

Level 2. *Rule-based control behaviour*: Command the robot which uses the human perception-decision action to define intermediate navigation goal during task execution. This is applied in exclusive shared mode via human intermittent input. The human input at this level is directional control of the robot heading.

Level 3. *Knowledge-based control behaviour*: Task-level supervision that requires human perception, decision making and planning. This is applied in exclusive traded mode and autonomous mode. The human inputs at this level are as follows: (a) single-point (both relative and absolute) and multi-points (absolute) commands for waypoint navigation; (b) landmarks specification for location-based navigation; and (c) shapes and colours model specification for visual-based task (e.g. object following). Finally, the human inputs can also be in the form of system configurations, such as sensing distance and navigation speed, activation and deactivation of particular sensors, robot autonomy and feedback components specified before operation. This applies at all three levels.

Apart from performing the above functions, the human control components also tries to derive the human control action and intention in real-time. This is to facilitate robot monitoring of the human's behaviours through the exclusive shared mode. The purpose is for the robot to provide appropriate assistance to the human.

B. Components for Enabling Robot Autonomy

Components for enabling robot autonomy control how the robot operates and cooperates with the human. The components are further classified into components for operating autonomy and decisional autonomy. The operating autonomy components provide means for sensing and hardware operation. The decisional autonomy components provide means for the navigation, localisation and planning subsystem described in Section 4.2.1. To understand how the robot varies its degree of autonomy and cooperates with the human within each level of interaction modes, there is a need to discuss the navigation subsystem, where all the behavioural units are situated. The basic behaviours are designed based on the required abilities of a mobile robot to navigate safely in a task environment. The implemented task-level behaviours are customized based on the tasks for an automated security system. Behaviour such as emergency stop uses a fixed autonomy (0 or 1); other behaviours, such as collision prevention, obstacle avoidance, goal seeking, etc. operates with a degree of autonomy varying from 0 to 1. The condition to select which behaviours is based on the task input from a human (or from other robots). The current implemented behavioural units includes behaviour such as emergency stop that uses a fixed autonomy (0 or 1); while other behaviours, such as collision prevention, obstacle avoidance, goal seeking, etc. operates with a degree of autonomy varies from 0 to 1. The conditions to vary the behaviours autonomy are dependent on:

- the strength of the control signal from the human;
- the strength of the corresponding sensors information about the environment, and

- the autonomy of other behaviours.

The condition to select which behaviours should be active is based on the interaction mode the human selected. As this topic is related to the coordination of behaviours, it is further discussed below in Part D.

C. Feedback Components

Feedback is important to ensure that the human is in effective command during task execution (Section 3.2.3 and 3.2.6). The current implementations of the interfaces are being explored on a PC-based station and also on portable devices such as PDA to facilitate different types of HRI (e.g. proximity and remote interactions). Although the interfaces are implemented on different systems, all have similar functions, such as displaying different type of the sensors information from the robot and robot control (i.e. from direct manual to autonomous control). The implementation details of these interfaces are described in Ong et al. (2004).

D. Coordination of Task Interaction Modes

A task interaction modes coordinator is required to coordinate human and robot activities to facilitate $T_{S\&T}$. Consequently, this facilitates the achievement of semi-autonomous control. In accordance to Section 3.2.4-5, a coordinator is required to coordinate system activities both "globally" and "locally". In this implementation, global coordination is achieved directly via interaction mode selection by human. This implies that the human will determine which level of task interaction mode is suitable for performing a particular task either through monitoring or during HRI. Through this, the coordinator will determine the appropriate robot behavioural units to use. This greatly reduces the need of coordinating all the behavioural units locally during task execution and hence requires lower computation. Those behaviours that are not in used are set to a lower priority (or off) according to their process ID. This form of coordination, via priority-based arbitration is used together with superposition-based command fusion to efficiently coordinate the robot behaviours locally. An illustration of this coordination scheme is depicted Fig. 10. The notation representation is as follows: V – Actuator vectors, A – Degree of autonomy, Σ – Summation (superposition-based command fusion) and S – Suppression (priority-based arbitration)
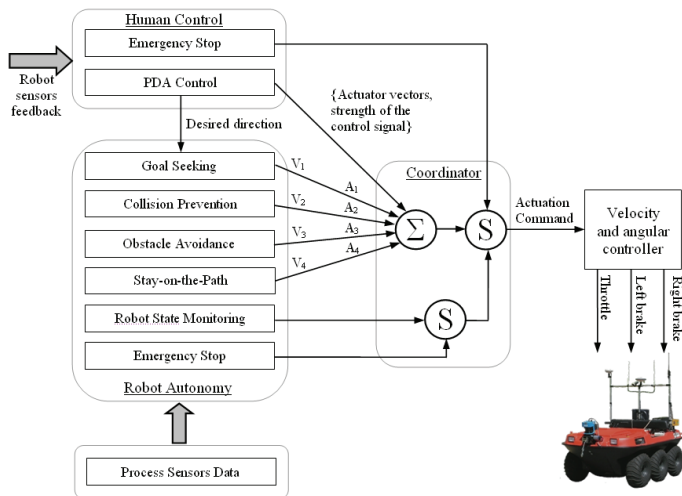

Figure 10. An example of hybrid coordination in exclusive shared mode

4.3 Experimental studies

It is essential that research into robotics be proven by experimental study of concepts and ideas, and validation of algorithms. Many problems in robotics evolve from phenomena that occur during its interaction with a dynamically changing environment which do not lend themselves to a formal treatment. Thus experimental studies of robotics system would preferably be based on real-world experimentation using physical robots, as opposed to the use of simulation (Brooks, 1986). In this context, a widely subscribed approach for the evaluation of robotics system is the use of *proof-of-concept*, which is to show that the system is capable of accomplishing the particular task it is designed for. Consequently, this approach is adopted to investigate the concept of semi-autonomous control invoked by $T_{S\&T}$ (i.e. task sharing and trading) established in Section 3. Different experiments have been conducted to assess the concept of semi-autonomy. This includes:

- The assessment of the cooperation between human and robot from the perspective of how assistance can be provided to the human by the robot. The aim is to establish a basis to show that there is a difference in task performance between robot assistance provided to the human and when no robot assistance is provided. The purpose is to provide a basis for assessing human-robot roles transitions from no assistance to maximum assistance provided to the human by the robot. The results obtained shows that the system performance is improved significantly when the robot provided appropriate assistance to the human as compared to no assistance was provided (Ong et al., 2008).

- The assessment of seamless semi-autonomous control due to local $T_{S\&T}$. As defined in Section 3.1.5, local $T_{S\&T}$ is the ongoing interaction role changes between human and robot in performing a desired input task with the aim of improving the current HRS task performance. The purpose is to show the need of using different human-robot roles and flexibility in roles changing as discussed in Section 2.2. The results obtained shows that a better performance can be achieved if human and robot can change their interaction roles dynamically during task execution as compared to the use of fixed interaction role (Ong et al., 2008).

- The assessment of the cooperation between human and robot from the perspective of how assistance can be provided to the robot by the human. This includes the evaluation of semi-autonomous control due to change of human-robot roles with completely different type of task specification (i.e. global $T_{S\&T}$). The assessment is based on how human assistance can be provided to robot seamlessly in different task situations. One of the main experiments conducted is *waypoints navigation*: The task is for the human to command the robot(s) from one location to another location via waypoint(s) control as depicted in Fig. 11. Assistance provided to the robot by the human is in the context of planning a safe path for the robot to navigate. This experiment is presented in (Ong et al., 2007).

Finally, to illustrate the functionality and working principles of the concept of semi-autonomy, a simple teleoperation demonstration, using manual and exclusive shared control (Fig. 3) is presented. Here, the task is to allow for the human to teleoperate the ARGO (Fig. 8) from location "A to B". That is, if the ARGO is at location A, the problem is to

control the robot's actions so that it moves from A to B. As basic as this task may seem, executing it successfully is critical for many mobile robotics applications.
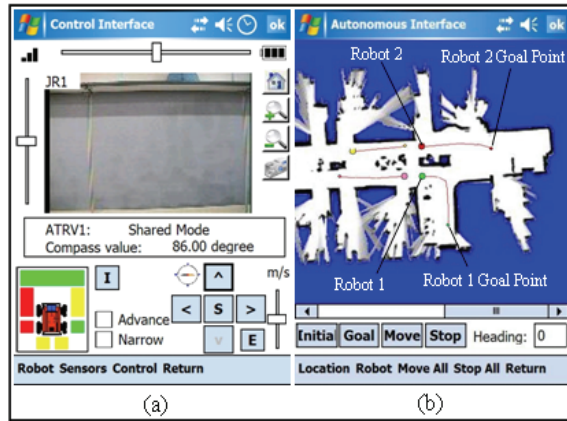

Figure 11. Snapshot of the PDA interface: (a) Control Menu and (b) Waypoint Menu

In the *manual* mode, the human has full control of the robot actions. This includes starting and stopping the engine. The robot takes no initiative except to stop when the wireless communications breakdown. However, the human can configure the robot to take basic initiative to protect itself. Under this safety feature, when the robot approaches an obstacle, the collision prevention behaviour will be activated to decelerate and stop near the obstacle within a safe sensing distance configured by the human. In the context of teleoperation using this mode, the human needs to make adjustments to compensate for drift to ensure the robot motion is straight (e.g. due to travel over uneven terrain). In addition, the human must also make fine adjustments to achieve smooth steering during turning.

In the *shared* mode, both the human and the robot can control same/different aspects of the system concurrently. In contrast to the manual mode, the robot has the capability to automatically accomplish the navigation task by assessing its own status and surroundings, and to decide whether the commands by the human are safe. This includes automatically adjusts for drift, guidance (i.e. stay-on-path) and obstacles avoidance. This relieves the human of detailed control and lets him concentrate on the overall goal of the task. However, as compared to the manual mode, the shared mode could not reach the maximum speed (i.e. 1.8 m/s for this experiment setup) because the robot was constantly reacting to the obstacles in the test environment to ensure safe teleoperation. This is depicted in Fig. 12, which illustrates the semi-autonomous control of the ARGO using a sequence of behaviours (i.e. collision prevention, obstacle avoidance, stay-on-path and goal seeking).

As shown in Fig. 12a-d, when the ARGO approaches an object (i.e. the orange cone), the collision prevention behaviour was first invoked for safe deceleration. Subsequently, the robot assisted the human by performing autonomous obstacle avoidance manoeuvre (Fig. 12e-h) and continues to move forward in accordance to the desired human control direction (Fig. 12i) indicated by the goal seeking behaviour, directed via the path planner. The coordination architecture to achieve this is depicted in Fig. 10.
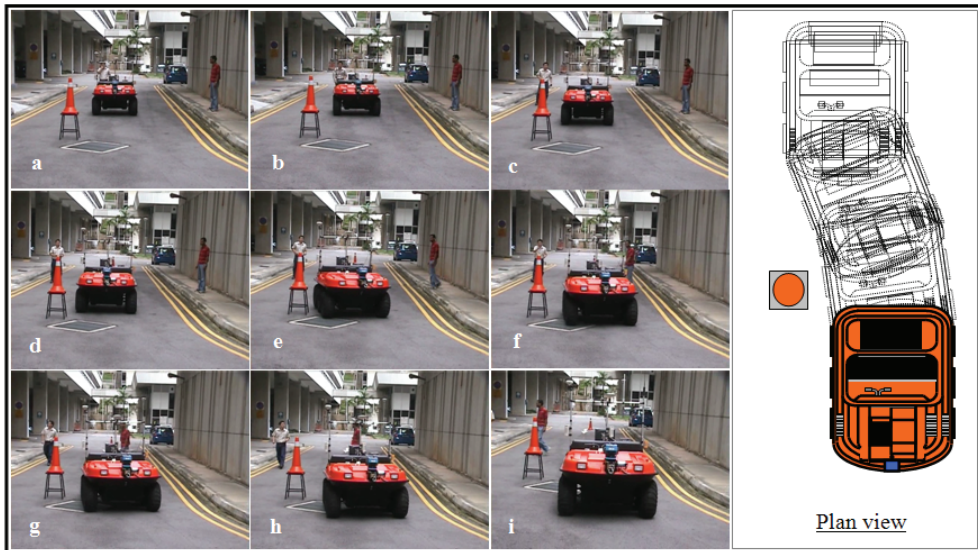
Figure 12. Montages of the ARGO vehicle during teleoperation under shared mode

5. Discussion and conclusion

In this paper, the development of a framework for semi-autonomous control using the concept of sharing and trading is discussed. This includes the application of this framework in the implementation of a telerobotics system. The purpose of the semi-autonomous control scheme is to facilitate the human and the robot to cope with different aspects of their interactions. By enabling the human to interact with the robot semi-autonomously, the human can decide the most appropriate scheme for performing a task. The main advantage of designing the robot control architecture using the concept of semi-autonomy is the ability for the control system to tolerate shifts in autonomy from manual to full autonomy (Fig. 3). In another words, at each level of interaction mode, the required robot competence is considered from the start of the design stage. The main theme of the semi-autonomous scheme is to design the robotics system as a cooperative endeavour, where both the human and the robot can assist each other.

In general, the implemented control modes based on the concept of semi-autonomy have two important features: *complement* and *redundancy*. The control modes are *complementary* in order to allow for the human and the robot to deal with different aspects of a task. In addition, the control modes are also *redundant* to provide more options for the human to develop strategies to perform the task. This implies that if any control mode fails, the human can use another mode to perform the desired task. In accordance Section 4.3.1, to navigate from location A to B, the human can have the option to either teleoperate the robot to location B using the manual mode or the shared mode. In the context of system implementation, to give the human the abovementioned control flexibility, the robot must imbue with the capabilities to adjust its level of autonomy during operation. To realise, the solution offers in this paper is the integration of the human control and the robot autonomy

via an implementation of modular semi-autonomous software architecture for them to interact dynamically.

Although this paper established a comprehensive study on the formulation of a framework for semi-autonomous control, there are other related issues that are not addressed. Firstly, to address the human-robot authority issues it requires that neither human nor robot to be exclusively in charge of HRS task, but rather it requires human to retain as the overall responsibilities of the outcome of the tasks undertaken by the robot and retains the final authority corresponding with that responsibility. This is achieved by giving human both the flexibility in delegating control to the robot at different levels and varying his/her control involvement with the robot at varying degree of task details. However, to develop a truly cooperative HRS, it may not be the best approach, particularly in situation when human does not have the capability to delegate control to the robot or vary his/her control involvement with the robot due to insufficient/poor task feedback. If this happen, the system may be disabled and, thus affect the achievement of seamless HRI. A possible solution is to allow both human and robot to have "equal" authority to correspond to the overall responsibilities of the outcome of tasks together. This implies not just giving the robot the authority to leads certain aspect of a task, but also responsible for the task success or failure of the task. If this can be modelled, a more "synergistic" approach can be provided for addressing the human-robot authority issues. Thus, an extension of this research is to investigate this issue and how it can be incorporated into the current framework.

Second, to envision a "tighter" cooperation between human and robot just as in human-human teamwork requires not only understanding (or modelling) of each other actions and intentions, but may also depend on human's "trust" in interacting with robot. As operations in an HRS can be complex, humans may fail to understand the mechanism of the operations. For instance, robot may operate abnormally under certain conditions. Therefore, the level of trust human has in an HRS is important. If the human trusts the system too much, he/she may become complacent about the behaviour of the robot and may not be vigilant about understanding its effects (this may lead to serious accidents). On the other hand, if the human distrusts the system, the system can be disabled. According to Barber (1983), differing degree of trust impacts all interactions involving people and technology. Trust has been studied in a number of domains. For example in automation, Abe et al. (1999) showed that human's trust in automation is one of the major factors in the usage of automation. Following this, to make seamless semi-autonomous control effective,, the issue of human trusting the robot is important. Logically, trust is not acquired instantaneously; it must be built up gradually. For example, it is based on the experience of controlling the robot to know how it acts. If the action of the robot is predictable, the level of trust of the human will be high. However, if the robot acts abnormally, the level of trust of the human will be low. In the context of HRI, it will be useful to look into this aspect and how it can also be incorporated into the current framework.

6. References

Abe, G.; Itoh, M. & Tanaka, K. (1999). Trust in and Use of Automation: Their Dependence on Occurrence Patterns of Malfunctions Systems, *Proceedings of IEEE International Conference on System, Man, and Cybernetics*, Vol. 3, pp. 715 -720, ISSN 0884-3619.

Arkin, R. C. (1998). *Behavior-Based Robotics*. MIT Press, ISBN 0262011654, Cambridge.

Barber, B. (1983). *The Logic and Limits of Trust*, Rutgers University Press, ISBN 0813509580, New Brunswick, N. J.

Billings, C. E. (1997). *Aviation Automation: The Search for a Human-Centered Approach*, Lawrence Erlbaum Associates, ISBN 0805821260, Mahwah.

Bourhis, G. & Agostini, Y. (1998). The Vahm Robotised Wheelchair: System Architecture and Human-Machine Interaction, *Journal of Intelligent and Robotic systems*, Vol. 22, No. 1, May, 1998, pp. 39-50, ISSN 0921-0296.

Bradshaw, J. M.; Boy, G.; Durfee, E.; Gruninger, M.; Hexmoor, H.; Suri, N.; Tambe, M.; Uschold, M. & Vitek, J. (Eds.). (2002). *Software Agents for the Warfighter,* ITAC Consortium Report, AAAI Press/MIT Press, Cambridge.

Brooks, R. A (1986). A Robust Layered Control System for a Mobile Robot, *IEEE Journal of Robotics and Automation,* Vol. RA-2, April, 1986, pp. 14-23, ISSN 0882-4967.

Bruemmer, D. J.; Marble, J. L.; Dudenhoeffer, D. D.; Anderson, M. O. & McKay, M. D. (2003). Mixed-Initiative Control for Remote Characterisation of Hazardous Environments, *Proceedings of the 36th IEEE Annual Hawaii International Conference on System Sciences, pp. 127-135,* ISBN 0769518745, Waikoloa Village, Hawaii, January 06-09, 2003.

Burke, J. L.; Murphy, R. R.; Rogers, E.; Lumelsky, V. L.; & Scholtz, J. (2004). *Final Report for the DARPA/NSF Interdisciplinary Study on Human-Robot Interaction, IEEE Transactions on Systems, Man, and Cybernetics – Part C, Vol. 34, No. 2, May, 2004, pp. 103-112, ISSN 1094-6977.*

Carroll, D.; Gilbreath, G. A. & Everett H. R. (2002). Extending Mobile Security Robots to Force Protection Missions. *AUVSI Unmanned Systems,* Lake Buena Vista, FL, July 9-11, 2002.

Chang, R. Y. (1995). *Success through Teamwork: A Practical Guide to Interpersonal Team Dynamics*, Kogan Page, ISBN 0749416610, UK.

Desai, M. & Yanco, H. A. (2005). Blending Human and Robot Inputs for Sliding Scale Autonomy. *Proceedings of the 14th IEEE International Workshop on Robot and Human Interactive Communication (Ro-MAN)*, pp. 537-542, ISBN 0780392752, Nashville, TN, August, 2005.

Gat, E. (1998). On Three-Layer Architectures, *Artificial Intelligence and Mobile Robots: Case Studies of Successful Robotics System*, D. Kortenkamp, P. R Bonnasso & R. Murphy (Eds.), ISBN 0262611376, AAAI Press.

Fiorini, P. & Prassler, E. (2000). Cleaning and Household Robots: A Technology Survey, *Autonomous Robots*, Vol. 9, pp. 227-235, December, 2000, ISSN 09295593, Kluwer Academic Publishers, The Netherlands.

Fong, T. W. (2001). *Collaborative Control: A Robot-Centric Model for Vehicle Teleoperation*, Ph.D. Dissertation, Robotics Institute, Carnegie Mellon University, US.

Gage, W. D. (1995). UGV History 101: A Brief History of Unmanned Ground Vehicle (UGV) development efforts, *Unmanned Systems Magazine* Vol. 13, No. 3, Summer, 1995, pp 9-16.

Goodrich, M. A., McLain, T. W., Anderson, J. D., Sun, J. & Crandall, J. W.. Managing Autonomy in Robot Teams: Observations from Four Experiments. *Proceedings of the 2nd ACM SIGCHI/SIGART IEEE RAS Conference on Human Robot Interaction*, Arlington, Virgina, USA, March 8-11, 2007.

Haegele, M.; Neugebauer J. & Schraft, R. D. (2001). From Robots to Robot Assistants, *Proceedings of the 32nd International Symposium on Robotics*, pp. 404-409, ISBN 8988366042, Seoul, Korea, April 19-21, 2001.

Hamel W. R. & Feldman M. J. (1984). The Advancement of Remote Technology: Past Perspectives and Future plans, *Proceedings of the National Topical Meeting on Robotics and Remote Handling in Hostile Environments*, American Nuclear Society, Gatlinburg, TN.

Hancock, P. A. (1992). On the Future of Hybrid Human-Machine Systems. In: *Verification and Validation of Complex Systems: Human Factors Issues*, J. A. Wise, V. D. Hopkin & P. Stager (Eds.), NATO ASI Series F, Vol. 110, pp. 61-85, Springer-Verlag, ISBN 3540565744, Berlin.

Hwang, S-L.; Barfield, W.; Chang, T-C. & Salvendy, G. (1984). Integration of Humans and Computers in the Operation and Control of Flexible Manufacturing Systems, *International Journal of Production Research*, Vol. 22, 1984, pp. 841-851, ISSN 0020-7543, Taylor & Francis.

Inagaki, T (2003). Adaptive Automation: Sharing and Trading of Control, In: *Handbook of Cognitive Task Design*, E. Hollnagel (Ed.), Lawrence Erlbaum Associates, ISBN 0805840036, Hillsdala, NJ.

Klingspor, V.; Demiris, J. & Kaiser, M. (1997). Human-Robot Communication and Machine Learning, *Applied Artificial Intelligence*, Vol. 11, pp.719-746, ISSN 1087-6545.

Kortenkamp, D., Kerin-Schreckenghost, D. & Bonasso, R. P. (2000). Adjustable Control Autonomy for Manned Space Flight Systems, *Proceedings of the IEEE Aerospace Conference*, pp. 629-640, ISBN 0780395468, March 18-25, 2000, Big Sky, Montana.

Lee, S. (1993). Intelligent Sensing and Control for Advanced Teleoperation, *IEEE Control System Magazine*, Vol. 13 (3), June, 1993, pp. 19-28, ISSN 0272-1708.

Martens, C.; Ruchel, N.; Lang, O.; Ivlev, O. & Gräser, A. (2001). A Friend for Assisting Handicapped People, *IEEE Robotics & Automation Magazine*, March, 2001, pp. 57-65, ISSN 1070-9932.

Nicolescu, M. N. & Matarić, M. J. (2001). Learning and Interacting in Human-Robot Domain, *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, Vol. 31, No. 5, September, 2001, pp.419-430, ISSN 1083-4427.

Norman, D. A. (2002). *The Design of Everyday Things*, Basic Book, ISBN 0465067107, New York.

Ong, K. W.; Seet, G.; Sim, S. K.; Teoh, W.; Lim, K. H.; Yow, A. N. & Low, S. C. (2004). A Testbed for Human-Robot Interactions, *Proceedings of ASME DETC'04 28th Biennial Mechanisms and Robotics Conference*, ISBN 0791837421, Salt Lake City, Utah, September 28 to October 2.

Ong, K. W. (2006). *Sharing and Trading in a Telerobotics System*, Ph.D. Dissertation, Robotics Research Centre, School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore.

Ong, K. W.; Sim S. K.; Seet G.; Goh B. K. & Huynh V. A. (2007). A Handheld Interface for mixed Multi-Agent Interaction, *Proceedings of the 4th International Conference on Ubiquitous Robots and Ambient Intelligence*, Pohang, Korea, Nov. 22-24, 2007.

Ong, K. W., Seet G. & Sim S. K. (2008), *An Implementation of Seamless Human-Robot Interaction* for Telerobotics, *International Journal of Advanced Robotic Systems,* to be appeared on the June 2008 issue.

Papanikolopoulos, N. P. & Khosla, P. K. (1992). Shared and Traded Telerobotics Visual Control, *Proceedings of the IEEE Conference on Robotics and Automation*, pp. 878-885, ISSN 1050-4729, 1992.

Pedersen, L.; Kortenkamp, D.; Wettergreen, D. & Nourbakhsh, I. (2003). A Survey of Space Robotics, *Proceedings of the 7th International Symposium on Artificial Intelligence, Robotics and Automation in Space*, Nara, Japan, May 19-23, 2003.

Rasmussen, J. (1983). Skills, Rules and Knowledge; Signals, Signs and Symbols, and other Distinctions in Human Performance Models, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 13, No. 3, pp.257-266, ISSN 1083-4427.

Roman, H. T. (1993). Robotic Applications in PSE&G's Nuclear and Fossil Power Plants, *IEEE Transactions on Energy Conversion,* Vol. 8, No. 3, September, 1993, pp. 584-592, ISSN 0885-8969.

Russell, S. & Norvig, P. (2002). Artificial Intelligence: A Modern Approach (2nd Edition). Prentice Hall, ISBN 0137903952, New Jersey.

Sheridan, T. B. (1987). Supervisory Control, In: *Handbook of Human Factors/Ergonomics*, G. Salvevely (Ed.), Wiley, ISBN 0471116, New York.

Sheridan, T. B. (1992). *Telerobotics, Automation, and Human Supervisory Control*, MIT Press, ISBN 0262193167, Cambridge.

Sheridan, T. B. (1997). Task Analysis, Task Allocation and Supervisory Control, In: *Handbook of Human-Computer Interaction (2nd, Completely Revised Edition)*, M. Helander, T. K. Landauer & P. Prabhu (Eds.), pp. 87-105, Elsevier, ISBN 0444818262, Amsterdam.

Shimon, Y. N. (editor) (1999). *Handbook of Industrial Robotics*, 2nd Edition, ISBN 0471177830, John Wiley & Sons.

Sim, S. K.; Ong K. W. & Seet, G. (2003). A Foundation of Robot Learning, *Proceedings of the 4th IEEE International Conference on Control and Automation*, pp.649-653, ISBN 078037777X, Montreal, Canada, June 10-12, 2003.

Suomela, J. & Halme, A. (2002). Novel Interactive Control Interface for Centaur-Like Service Robot, *Proceedings of the 15th IFAC World Congress on Automatic Control*, Barcelona, Spain, July 21 – 26, 2002.

Tahboub, K. A. (2001). Natural and Manmade Shared-Control Systems: An Overview, *Proceedings of the IEEE International Conference on Robotics & Automation*, pp. 2655 – 2660, ISSN 1050-4729, May 21-26, 2001.

Vertut, J. & Coiffet, P. (1985). *Teleoperation and Robotics - Evolution and Development*, Robot Technology Vol. 3A, Rogan Page, ISBN 0850385881, London.

Wasson, G. S. & Gunderson, J. P. (2001). Variable Autonomy in a Shared Control Pedestrain Mobility Aid for Elderly, *Proceedings of the IJCAI Workshop on Autonomy, Delegation and Control*, 2001.

Deployment of Wireless Sensor Network using Mobile Robots to Construct an Intelligent Environment in a Multi-Robot Sensor Network

Tsuyoshi Suzuki[1], Kuniaki Kawabata[2], Yasushi Hada[3], Yoshito Tobe[1]
[1]Tokyo Denki University
[2]RIKEN
[3]National Institute of Information and Communications Technology (NICT)
Japan

1. Introduction

A Wireless Sensor Network (WSN), consisting of a huge number of small devices called sensor nodes (each with wireless communication functionality, various sensors, a processor, and a power source), is a network system that can communicate and use sensing data gathered mutually by each spatially distributed sensor node. An ad-hoc network connecting each sensor node one by one can be constructed only by deploying an enormous number of sensor nodes, and such a network can be enhanced very easily compared with wired and fixed networks. WSNs can provide various services by collecting and processing the information acquired by the sensor node. WSN technology is expected to be applicable within many fields, such as in the cooperative monitoring of the condition of an environment with a large area, factory equipment, or buildings, as well as in disaster relief support.

Additionally, in the area of robotics research, there have been studies on such topics as environmental information structuring and intelligent environments that examine the creation of intelligence not just in robots, but also in ambient environments (e.g. Sato *et al.*, 1996). WSN technology is now the object of attention among researchers attempting to create such intelligent environments.

We thought that it would be possible to construct a mobile ad-hoc network with autonomous mobile sensor nodes, environmentally adaptive WSN deployment, and applications within various intelligent systems. Autonomous mobile robot can be used and act as high-performance mobile sensor node in WSN because robots equipped with various sensors and communication capabilities. If a robot can manipulate sensor nodes, that robot can change the range and topology of its WSN according to the communication conditions, sensing and adapting to the environmental situation. Through a fusion of WSNs and robotics, it may be possible to extend communication and sensing areas, replace failure nodes by robots, and reconstruct WSN by relocating its sensor nodes.

Moreover, WSNs provide advantages for robots by enabling them to gather and communicate wide-ranging environmental information to one another without relying on

existing network infrastructures. Robots grasp the environmental circumstances using various sensor data obtained from the WSN, and use these data in several different tasks.

We expect such systems to be superior with respect to the respective costs of deployment, movement, communication, energy, etc. Additionally, we expect these systems to exhibit robustness superior to that of conventional WSNs composed of fixed sensor nodes or homogeneous mobile sensor nodes when sensing environments with remarkably adverse wireless communication conditions and wide ranges. Such systems, which we refer to as MRSNs (Multi-Robot Sensor Networks), are the current focus of our research efforts.

This chapter introduces applications of MRSN that the authors are currently investigating. The rest of the chapter is organized as follows: Section 2 describes disaster area information gathering supported by the application of MRSN. Section 3 explains the autonomous construction and management of WSNs by autonomous mobile robots, and Section 4 details the design and development of MRSN systems for the support of disaster area information gathering. Concluding remarks are given in Section 5.

2. Support for disaster area information gathering by MRSN

Large-scale earthquakes occur in various regions of the world, and the disaster awareness of most populations has risen in recent years. Thus, in addition to disaster prevention, researchers have focused on "disaster mitigation," which reduces the damage after the occurrence of a disaster, as a measure to counteract difficulty of predicting certain types of incident. In disaster mitigation, it is necessary to understand the disaster situation as promptly as possible. For instance, analytical research conducted following the Great Hanshin-Awaji Earthquake (Kobe, Japan) of 1995 highlighted the importance of the prevention of secondary disasters such as fires occurring after the main incident to the effectiveness of the rescue operation. Therefore, it is crucial that information gathering and surveillance be promptly and continuously implemented in large disaster areas if the symptoms of such secondary disasters are to be detected sufficiently early. However, it is difficult to collect disaster area information over adequately large areas because the human resources, including fire-fighters and rescue team members, in these various locations are generally limited.

Study of the manual deployment of sensor nodes on building ceilings, streetlights, etc., for preemptive disaster area information gathering following a disaster (Kurabayashi *et al.*, 2001) is in rescue research projects (Tadokoro *et al.*, 2003) commissioned by the Ministry of Education, Culture, Sports, Science and Technology in Japan. Preemptively deployed fixed sensor nodes can collect information at any time continuously before a disaster occurs. However, their functionality following a disaster cannot necessarily be guaranteed. If people are relied upon to deploy sensor nodes, human resources are often insufficient, as previously stated. Moreover, human sufferings might be caused by secondary disaster during information gathering activity.

Based on the above mentioned information, we thought that we would be able to preserve the sensing functionality and the network through the prompt deployment of a WSN by using a MRSN autonomously and adaptively, and that we would be able to collect disaster area information adaptively. Such a system is potentially applicable as an information support system for information gathering and situation awareness in large disaster areas

and the reduction of suffering from secondary disasters through the deployment WSNs by MRSNs, the collection and processing of their sensor information, and the provision of information services based on the information thus collected.

The following describes some elemental technologies developed for MRNS-based environmental information gathering support systems in disaster areas.
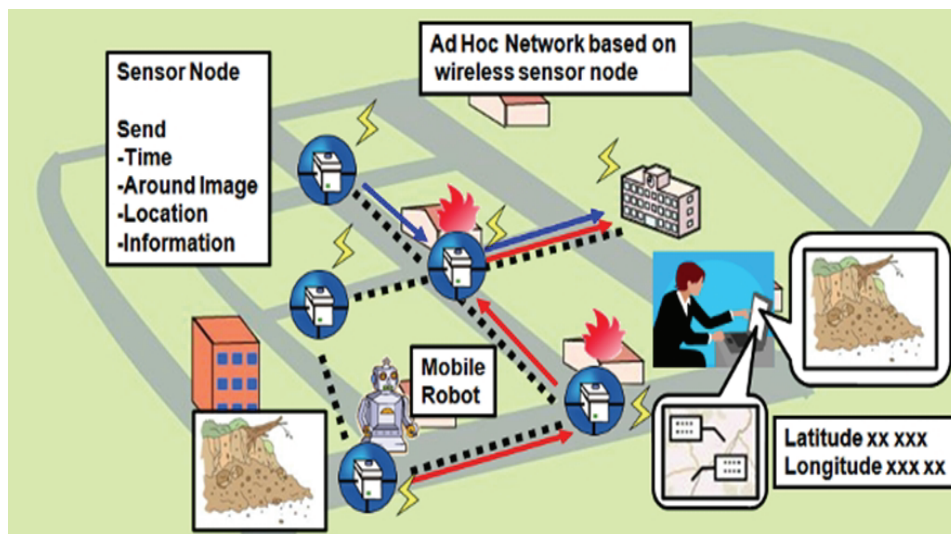

Fig.1. Conceptual sketch of MRSN-based environmental information gathering support system

3. Autonomous construction and management of WSNs by autonomous mobile robots based on electric field strength measurements

3.1 Autonomous construction and management of WSNs by mobile robots

The autonomous construction of WSNs has been previously discussed in conventional studies on sensor node deployment. The methods proposed in these studies consist mainly of randomly scattering many low-cost sensor nodes, constructing WSNs with mobile sensor nodes, etc. (McMickell *et al.*, 2003, Dantu *et al.*, 2005, Parker *et al.*, 2003). However, in the scattering deployment method, there is a possibility that the sensor nodes may not necessarily be deployed to the desired locations. Although it may be possible to evade such a problem by scattering a massive number of sensor nodes, the costs and increased communication traffic associated with the deployment of a large number of sensor nodes may prove problematic. In WSNs composed of mobile sensor nodes, the sensor nodes can be deployed to any position desired. However, sensor nodes with mobile capabilities are expensive. Moreover, the sensor nodes need not necessarily all move, depending on the environment. In general, system cost is a significant problem because a large number of sensor nodes are needed to gather information over a wide disaster area. Therefore, it is necessary to consider the costs of the sensor nodes and thereby reduce the energy costs for the entire system.

Other methods for sensor node deployment have been proposed; these were based on maximum communication or sensing range using mobile sensor nodes or mobile robots (Batalin *et al.*, 2002, Miyama *et al.*, 2003, Sugano *et al.*, 2006) and deployment by virtual interaction between sensor nodes based on physical models (Howard *et al.*, 2002, Pac *et al.*, 2006). However, these studies assumed that it would be difficult to guarantee communication between sensor nodes due to obstacles and interference waves that might block communication channels. Moreover, it is possible for that the communication between sensor nodes in WSNs to be interrupted due to decreases in sensor node battery levels or device breakage. Therefore, it is necessary to understand the status of WSNs, to ensure communication between sensor nodes, and to maintain their functionality as adaptive information communication network.

In our approach, the sensor node cost problem is solved by using low-cost sensor nodes that can perform the minimum functions necessary for environmental information gathering. Moreover, the energy cost problem is solved by enabling mobile robots to construct WSNs. In order to ensure communication between sensor nodes, the electric field strengths between nodes are monitored while the robot-deployed nodes are in transit to their designated locations. The robots confirm that the sensor nodes can communicate with one another, and deploy sensor nodes while guaranteeing communication channels between them. This proposed method is expected to enable construction of WSNs adaptable to changes in field strength caused by environmental interference.

After such a WSN is constructed, the circumstances under which it would be unable to continue functioning are estimated according to the battery level decrease that would result in the failure of a sensor node. Its robots can then specify the necessary details for a replacement sensor node by using positional information recorded when the original sensor node was deployed (odometric information, for instance). When a signal can be received from the sensor node, the accuracy of the detection of its location is improved using the field strength. Finally, the mobile robot moves to the location of the target sensor node, the alternative sensor node is deployed, and the function of the WSN is maintained.

3.2 Prototype system for verification of proposed method

A prototype platform that assumed the construction of a WSN in an indoor environment was developed and tested to verify the proposed method. The omni-directional mobile robot ZEN (Asama *et al.*, 1995) was used as the mobile robot platform (Fig. 2(b)). Mica2 MOTEs (Crossbow Technology, Inc.) were used as the sensor nodes (Fig. 2(d)). Each sensor node had a unique ID. Because the sensor nodes sent the transmission signals to on another that included the values of their own battery voltages, the condition of each of sensor node could be monitored over the WSN. A sensor node transportation and deployment device was developed for WSN construction. The device consisted of a sensor node tray into which sensor nodes were placed (Fig. 2(c)) and a sensor node manipulation mechanism able to carry and place the sensor node tray on the ground. This sensor node tray moved in a vertical direction due to the screw turned by Motor 2. Motor 1 moved the entire unit including screw and Moter 2 up and down vertically. The sensor node tray could be grounded by turning Motors 1 and 2. A single robot was able to transport and install 5 – 10 nodes at once using this device.
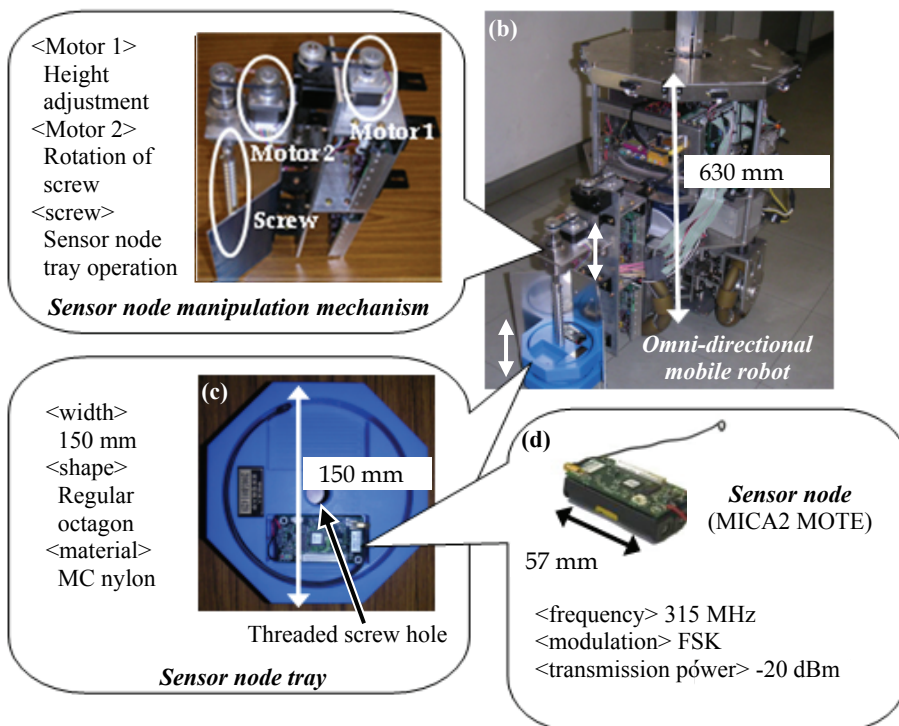
Deployment of Wireless Sensor Network using Mobile Robots to Construct
an Intelligent Environment in a Multi-Robot Sensor Network

319

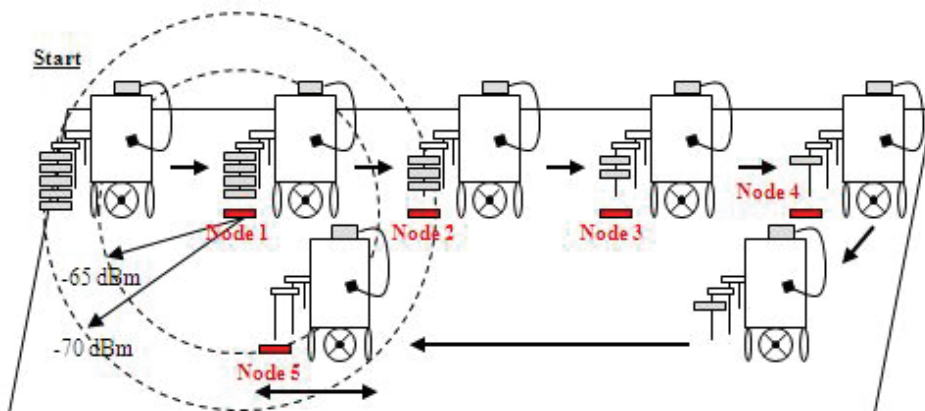Fig. 2. Overview of prototype system

Fig. 3. Outline of the experiment based on the scenario

3.3 Experimental set-up and results

A preliminary experiment was conducted in order to confirm the characteristics of the electromagnetic waves propagated between sensor nodes in order to enable stable communication between these nodes.

The results confirmed that the an electric field strength threshold of -70 dBm would be needed to ensure stable communication between sensor nodes. This experiment measured the electric field strength of a robot moving after installing a sensor node. The robot installed sensor node one by one measuring the electric field strength threshold of -70[dBm]. The experiment was executed in an indoor passage (height: about 2.24 m, width: about 1.77 m, total length: about 40 m) of a ferroconcrete building.
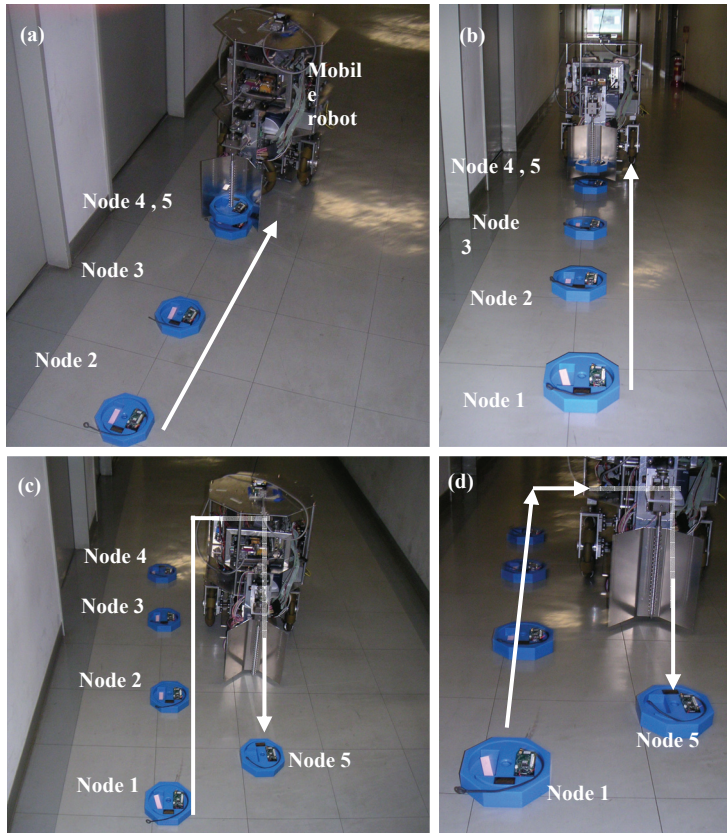

Fig. 4. Actual experiment on the autonomous construction and management of a WSN using a MOTE and omni-directional mobile robot

The scenario modeled in the experiment was as follows. The robot was given the task of installing sensor nodes. The robot initially placed a sensor node on the ground after receiving a command to carry out the autonomous construction of a WSN. The robot moved while simultaneously measuring the electric field strength between sensor nodes until it reached a preset value. The second sensor node was deployed at the point where this occurred. The robot constructed a WSN by repeating this operation, deploying sensor nodes one by one while measuring the field strength between sensor nodes. In this experiment, the battery levels of the sensor nodes were also assumed to be randomly decreasing. The robot was programmed to detect sensor nodes with low batteries, move to vicinities of such nodes, and deploy replacement sensor nodes near by in order to maintain the WSN.

Figure 3 shows an outline of the experiment based on this scenario. Photographs of the actual experiment are shown in Fig. 4. In this experiment, the robot deployed five nodes. The robot constructed a WSN using Nodes 1 to 4 by measuring electric field strength and deploying the sensor nodes with respect to its values. A simulated low battery signal was then sent from Node 1. The robot registered the status of Node 1 via the WSN and moved to a nearby location. A replacement sensor node, Node 5, was then deployed within the communication range of Node 2. Figure 4(a-b) shows the construction of the WSN. Figure 4(c-d) shows that Node 5 was deployed near Node 1 after the simulated low battery signal was detected. Figure 5 shows the distances between the successive sensor nodes. In this experimental environment, there was a distance difference of up to 10 cm between deployed sensor nodes. This is because the location of each sensor node was decided according to fluctuations in the electric field strength due to the particular characteristics of the sensor node and the status of communication within the environment. Therefore, we confirmed that it was possible to construct a WSN that ensured communication channels between sensor nodes and to manage a WSN using proposed method to restore interrupted communication paths.
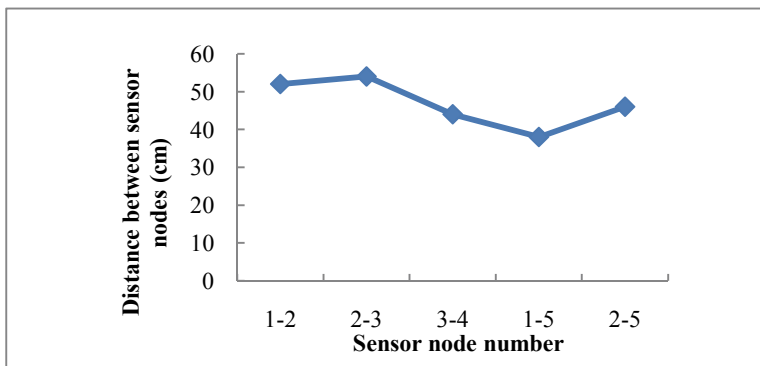

Fig. 5. Distance between deployed sensor nodes

4. Design and development of MRSN for disaster area information gathering support

4.1 Sensor node for disaster area information gathering support

In disaster areas, rubble is often scaterred due to the collapse of houses and other facilities, and lifelines, infrastructure, etc., often rupture or break down. A new sensor node device with infrastructure non-dependence, easy deployment, and the ability to construct a WSN is needed to gather information under such circumstances, because most conventional sensor nodes are difficult to use in disaster areas. We discussed the required specifications of such a new sensor node, and thought that the following functions would be necessary:

1. Power supply equipment for independent maneuvering
2. Wireless communication
3. Ability to construction of an ad hoc network
4. Information processing

5. Ability to acquire image of the surrounding environment and thereby recognize the environmental circumstances
6. Localization for effective use of sensor data
7. Low-cost direction control without depending on deployment method

Though other devices that capture transmits images of it in the hazardous areas have been developed, such as the Search Ball (Inoue *ea al.*, 2005) and the EYE BALL R1 (Remington Arms Company), a device with all of the above-mentioned functions as well as an ability to construct WSNs does not yet exist. Thus, we have designed and developed a prototype for a new sensor node satisfying these criteria.

4.2 Development of spherical sensor node equipped with passive pendulum mechanism

The sensor node that we developed consisted of a main controller with wireless communication capability, various sensing devices, and a passive control mechanism for maintaining constant sensor direction. Figure 6(a) shows the configuration of the sensor node. A small Linux computer, Rescue Communicator, produced by Mitsubishi Electric Information Technology Corporation was used as the main controller of the sensor node. Many various sensor devices could be connected togather because the Rescue Communicator had many input and output sites. The sensor node was equipped with a compact flash memory card, wireless LAN card, omni-directional vision camera connected with a LAN cable and mounted with a fish-eye lens for captureing 2π sr images of its surroundings, a 3-degree acceleration sensor for measuring the postural sway of the camera and a GPS system for localization. The Rescue Communicator and all of the sensors could be diriven by the sensor node's battery.
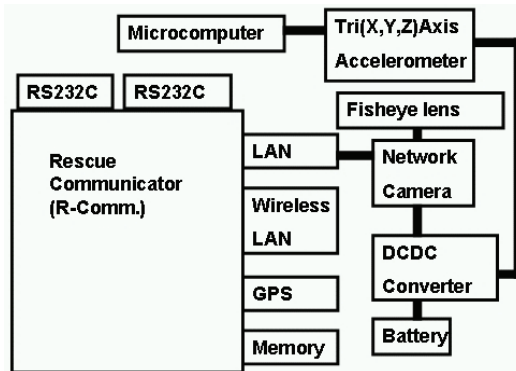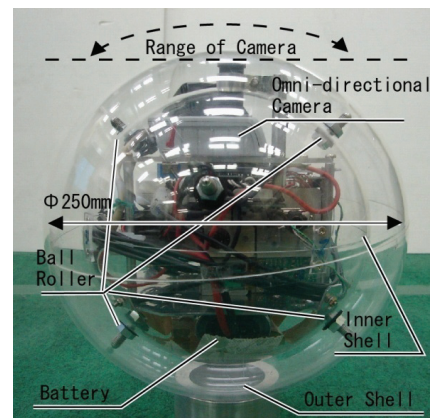
(a) Configuration of the sensor node   (b) Prototype of spherical sensor node

Fig. 6. Spherical sensor node equipped with a passive pendulum mechanism

Moreover, we designed the sensor node as shown in Fig. 6(b) to enable low-cost sensor postural control. The sensor node was designed so that the main body (inner shell) was surrounded by a spherical acrylic shell (outer shell) supported by the six ball rollers. The sensors (camera, etc.) were placed in the upper part of the inner shell, and heavy

Deployment of Wireless Sensor Network using Mobile Robots to Construct
an Intelligent Environment in a Multi-Robot Sensor Network

323

components such as batteries were placed in the bottom. The inner shell rotated freely inside the outer shell by way of the ball rollers. Thus, the camera always remained upright within the outer shell because the heavy load was placed opposite the sensors, creating a passive pendulum mechanism and keeping the camera view in the upward direction. Therefore, it was possible to obtain omni-directional images from the same point regardless of the direction from which the device was deployed. AODV-uu was installed on the Rescue Communicator to eneble construction of an ad-hoc network.

4.3 Functional verification of prototype sensor node model

An experiment was executed in order to confirm the information-gathering functions and ad-hoc networking capabilities of the developed sensor node. Figure 7(a) shows the experimental environment. In this experiment, an ad hoc network was constructed in an the outdoor containing a building, and the image data acquired by the sensor node was transmitted to the host PC in two hops. The sensor node transmitted image data of about 10 kbytes in the size of 320×240 pixels, along with information on the time the image was taken and latitude and longitude data recorded by the second.

(a) Experimental environment   (b) Example image captured
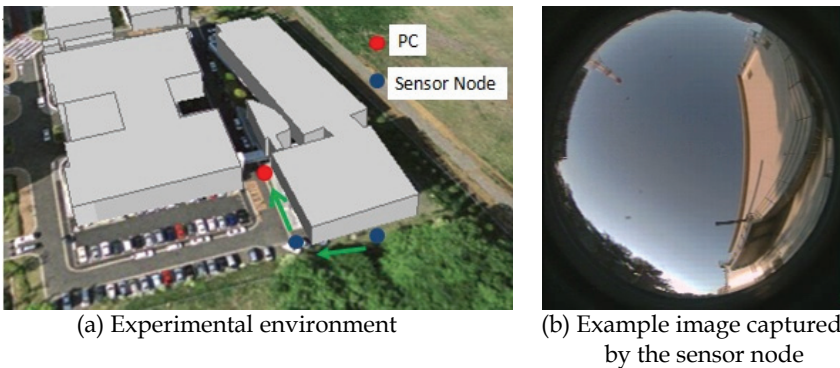   by the sensor node

Fig. 7. Experimental set-up for functional verification of the sensor node

Table 1 shows the results of the data received on the host PC. Information on the sensor node, including time, latitude, longitude and transmitted image (Fig. 7(b)) was transmitted to the host PC. Figure 8 shows the time required to receive on the host PC the images sent by the sensor node. At points where the image capture time was notably longer, communication between the sensor nodes and host PC was interrupted. However, the host PC was able to receive images in an average of 2.3 seconds.

Total number of images received	427
Number of transmissions including time and location	81
Average time taken to receive an image (sec/data)	2.3
Total time spent capturing image data (sec)	975

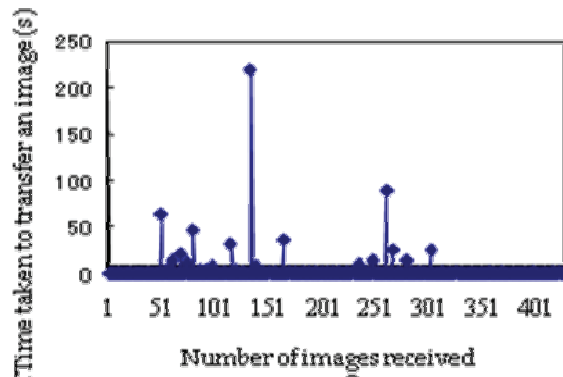Table 1. Results: Time taken for host PC to receive data from sensor node

Fig. 8. Time taken to receive images on host PC

4.4 Design and development of transportation and deployment mechanism for spherical sensor node

A device was developed to enable mobile robots to carry and deploy the spherical sensor node. This device was designed to roll spherical sensor nodes onto the ground using sloped guide rails because the sensor node was spherical and could be deployed easily without regard to the direction of installation.
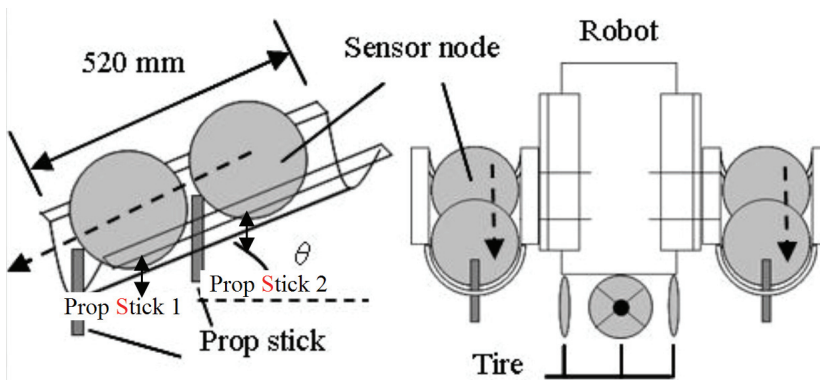

Fig. 9 Concept of the device for the transportation and deployment of spherical sensor nodes

Figure 9 shows an outline of the device. Two sloped giude rails were mounted on to the right and left sides of a robot. The robot was able to carry and deploy four spherical sensor nodes because each guide rail was able to hold two spherical sensor nodes at once in current system. The robot was able to deploy the sensor nodes by controlling prop sticks using the solenoid in this device. The first, lower sensor node rolled out by using its own weight to pull down Prop Stick 1 without moving Prop Stick 2, so that only one node was deployed into the environment. Next, the second, upper sensor node rolled down, by pulling down Prop Stick 2, to a position in front of Prop Stick 1 and was stopped the pushed-up Prop Stick 1. The second sensor node then rolled out by pulling down Prop Stick 1, and was deployed

into the environment. This transportation and deployment of the sensor node was made possible by taking advantage of the node's shape and characteristics and did not require an actuator or active control over position and attitude.
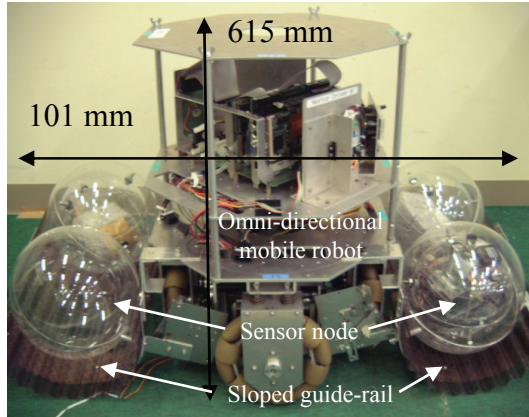

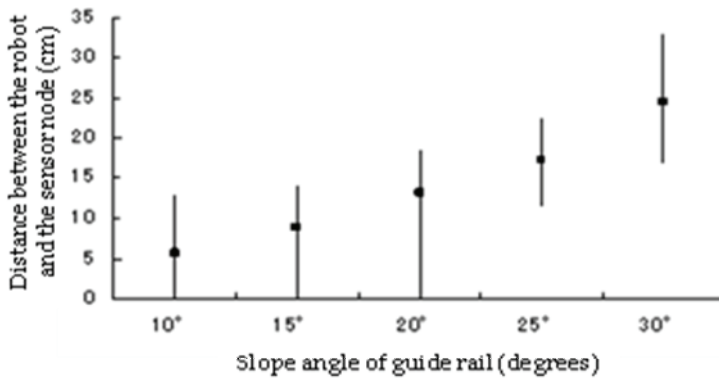Fig. 10 Prototype of the transportation and deployment device

Fig. 12. Distance between the robot and the sensor node as a function of guide rail slope angle

Figure 10 shows an image of the transportation and deployment prototype device mounted onto the omni-directional mobile robot. The guide rail was made from a corrugated polycarbonate plate in order to provide strength and reduce the contact surface area between the rail and the sensor node. The slope angle was adjustable. This design enabled the sensor node to roll easily from the guide rail into the environment.

Figure 12 shows the distance between the robot and the final positon of the sensor node after deployment as a function of the guide rail slope angle. It was possible to deploy a sensor node to within about 35 cm of a target position on a plain floor. Figure 13 shows the change in the visibility of a target image as a function of the error distance d from the target sensor node position. It was possible to recognize the surroundings of a target position when d was within 1.4 m.
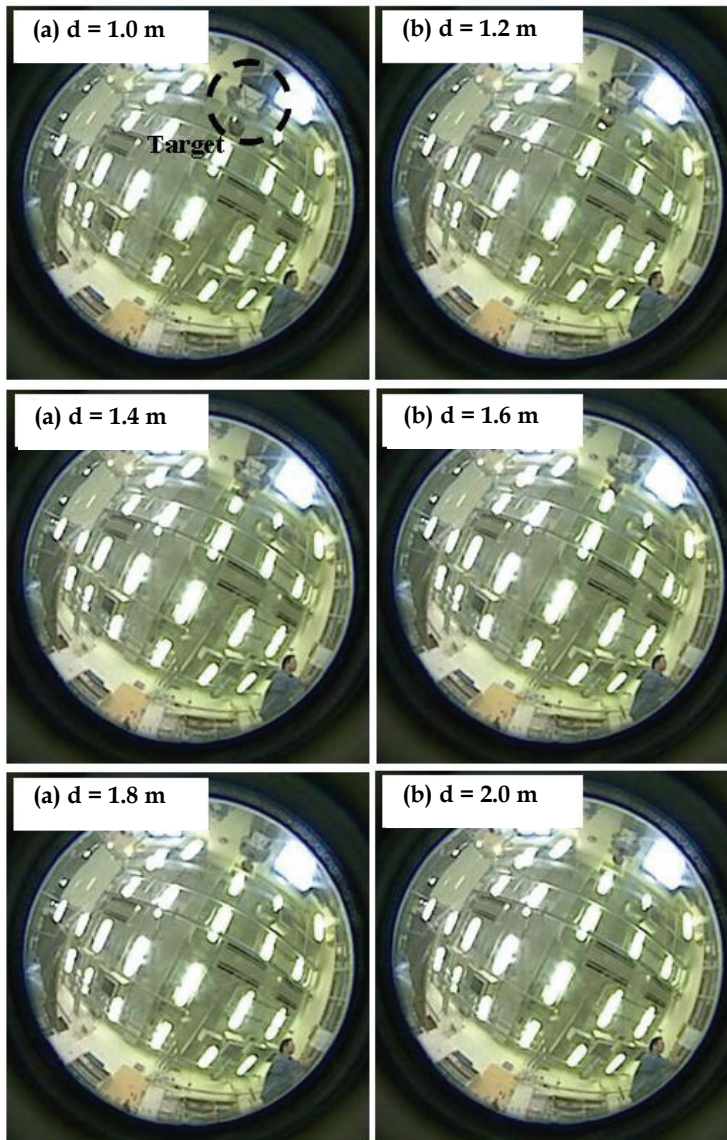
Fig. 13. Images captured at different sensor node deployment positions

5. Conclusion

This chapter has described the issues relevant to MRSNs consisting of WSNs and multiple robot systems. Additionally, we have introduced our work, which aims to develop support systems for information gathering in disaster areas via the application of MRSNs. Section 3

showed that a robot was able to construct and manage a WSN autonomously and adaptively by measuring electric field strength. Section 4 covered the design and development of a sensor node and its manipulation system for supporting disaster area information gathering. The system introduced here was a prototype; thus characteristics such as the robot's shape and the sensor node's environmental resistance must be further improved and developed to enable its practical application. Our future aims include the integration and upgrade of the component technology, as well as the improvement of the system so as to enable its use within realistic environments such as the outdoors and disaster areas. In addition, we will examine the communication protocol, information management, data transfer routing and the integration and processing of a large flow of information that would be appropriate for the proposed method.

MRSNs can construct WSNs adapted to their environments, and WSNs enable the robot mobile sensor nodes to gather and communicate a wide rang of environmental information to one another without relying on an existing network infrastructure. We expect that MRSNs will be applicable within adaptive sensing, the adaptive construction of information networks and various intelligent robot systems.

6. Acknowledgments

7. References

Asama, H. (1995) Development of an Omni-Directional Mobile Robot with 3 DOF Decoupling Drive Mechanism, *Proceedings of IEEE International Conference on Robotics and Automation*, pp.1925-1930, 0-7803-1965-6, 1995

Batalin, M. A. & Sukhatme, G. (2002) Sensor coverage using mobile robots and stationary nodes, *Proceedings of the SPIE, volume 4868 (SPIE2002)* pp. 269-276, Boston, MA, USA, August 2002

Dantu, K.; Rahimi, M., Shah, H., Babel, S., Dhariwal, A. & Sukhatme, G.S (2005) "Robomote: enabling mobility in sensor networks, *Proceedings of Fourth International Symposium on Information Processing in Sensor Networks*, pp. 404-409, 0-7803-9201-9, UCLA, Los Angeles, California, USA, April 2005

Howard, A.; Matari'c, M. J. & Sukhatme, G. S. (2002) Mobile Sensor Network Deployment using Potential Fields: A Distributed, Scalable Solution to the Area Coverage Problem, *Proceedings of Distributed Autonomous Robot System 2002*, Fukuoka, Japan, June 2002

Inoue, K.; Yamamoto, M., Mae, Y., Takubo, T. & Arai, T. (2005) Search balls: sensor units for searching inside rubble, *Journal of Advanced Robotics*, Vol.19, No.8, pp.861-878

Kurabayashi, D.; Asama, H., Noda, K. & Endo, I. (2001) Information Assistance in Rescue using Intelligent Data Carriers, *Proceedings of 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2294-2299, 0-7803-7398-7, Maui, Hawaii, USA, October 2001

McMickell, M. B.; Goodwine, B., Montestruque, L. A. (2003) MICAbot: A Robotic Platform for Large-Scale Distributed Robotics, *Proceedings of the 2003 IEEE International Conference on Robotics & Automation*, pp.1600-1605, 0-7803-7736-2, Taipei, Taiwan, September 2003

Miyama, S.; Imai, M. & Anzai, Y. (2003) Rescue Robot under Disaster Situation: Position Acquisition with Omni-directional Sensor, *Proceedings of the 2003 IEEWRSJ International Conference on Intelligent Robots and Systems*, pp. 3132-3137, 0-7803-7860-1, Las Vegas, Nevada, October 27-31, 2003

Pac, M. R.; Erkmen, A. M., Erkmen, I. (2006) Scalable Self-Deployment of Mobile Sensor Networks: A Fluid Dynamics Approach, *Proceedings of International Conference Intelligent Robots and Systems*, pp. 14460-1451, 1-4244-0259-X, Beijin, Chaina, October 2006

Parker, L. E.; Kannan, B., Fu, X. & Tang, Y. (2003) Heterogeneous Mobile Sensor Net Deployment Using Robot Herding and Line of Sight Fromations, *Proceedings of the 2003 IEEWRSJ International Conference on Intelligent Robots and Systems*, pp. 2488-2493, 0-7803-7860-1, Las Vegas, Nevada, October 2003

Sato, T.; Nishida, Y. & Mizoguchi H. (1996). Robotic Room:Symboiosis with human through behavior media, *Robotics and Autonomous Systems 18 International Workshop on Biorobotics: Human-Robot Symbiosis*, pp.185-194, Tsukuba, Japan, May 1995, ELSEVIER, New York

Sugano, M.; Kawazoe, T., Ohta, Y. & Murata, M. (2006) An indoor localization system using RSSI measurement of a wireless sensor network based on the ZigBee standard" , *Proceedings of the sixth IASTED International Multi-Conference on Wireless and Optical Communication*,   pp.503-508, Banff, Canada, July 2006

Tadokoro, S; Matsuno, F., Onosato, M. & Asama, H. (2003) Japan National Special Project for Earthquake Disaster Mitigation in Urban Areas, *First International Workshop on Synthetic Simulation and Robotics to Mitigate Earthquake Disaster*, pp.1-5, Padova, Italy , July 2003

Modularity in Service Robotics

Sami Ylönen
Helsinki University of Technology
Finland

1. Introduction

Versatile service robots are highly complicated systems. A breakthrough of a mobile service robot market requires substantial advancement in the standardization of the modules used in their structures.

At the moment, extensive technical requirements lead to expensive tailor-made realisations, which in turn prevent generic solutions, and thus the advantages of mass production cannot be utilised. The implementation of modular design, both in software and in hardware, is needed if we want to get good results at reasonable costs. Also, we need to be able to standardise the interfaces between different modules and towards environment. The final goal is that different plug-and-play subsystems and modules, such as navigation or machine vision modules, will become commercially available for everybody. In that case, resources could be focused on developing real applications and the actual product development would become cheaper, easier and faster. The concept of modularity might just be the decisive step for a breakthrough in mobile robotics; currently, there simply are not enough resources available for every research group to develop everything by themselves.

1.1 Definition of module

A module is an elementary functional unit that can easily be exploited in a different kind of application. A module for mobile robots is defined in Virk 2003a as follows: "A module for mobile robots is described as any functionally complete device, or sub-assembly, that can be independently operated and can be readily fitted and connected to, or in combination with, additional modules to comprise a complete and functionally reliable system." For example, a plain sensor component is not a module because the use of it requires signal processing and programming. If a digital databus is implemented to the sensor, this brings it closer to the definition. When the sensor is a fully plug-and-play component (e.g. USB-bus adaptive), it fulfils the module definition perfectly. A module is typically an independent versatile unit that can be connected to different kinds of devices. Also, for example, an analog sensor with remote software fulfils the definition of a module. The remote software should include not only the drivers but also some upper-level components that can, for example, process and analyse the data.

The Oxford English Dictionary (http://dictionary.oed.com/) defines the term module as "a component of a larger or more complex system. Any of a series of independent units or

parts of a more complex structure, produced to a standard design in order to facilitate assembly and allow mass production. More generally: any more-or-less self-contained unit, which goes to make up a complete set, a finished article, etc."

Modules consist of mechanics, electronics and software or some combination of these. Generally, modules that include mechanics also have at least electronics and, very often, software as well. For example, robot joints could be considered to be this kind of module. Sensor modules typically include electronics and software. User interface software, path-planning software and speech recognition software are examples of software modules. These modules are typically operated in close coordination with modules that include electronics and mechanics, but the modules might also act as clearly separated modules interacting only with other software modules. Each module has its own tasks, but several modules might have the same task. Most modules should have such roles that damage to one module would not paralyse the whole machine, but merely limits its capabilities to some degree.

Modules are connected to each other and to the rest of the system using interfaces. Figure 1 shows a set of interfaces that must be taken into account when attaching new modules to the system. The model has been developed in the CLAWAR[1] network. These interfaces include power (electrical, pneumatic, hydraulic), databus (communication databus, e.g. CAN, RS232, IEEE 488), mechanics (mechanical connectors, physical connections, joints types, range of movements), analogue (analogue signals), digital (digital signals, 0 or 1) and environment (surrounding environment and medium; air, water, dust, pollution, radioactive).

Figure 1 gives an example of how different modules can be connected to the various interfaces. The research work carried out to promote the modularity aims to standardise these interfaces. If this were to be achieved, the module developers would know exactly what kind of interfaces the modules should have. Standard interfaces would thus make the development of new modules much easier. The significance of modularity and standardisation as a driving force for the future expansion of a service robot market was emphasized by Bill Gates in Scientific American article (Gates, 2007).

Modules in the case study, presented in more detail in Chapter 4, are classified on the basis of the modified CLAWAR representation. There exist also decentralised modules, such as analog sensors, whose sensor and software are located separately. Super modules include several basic modules (Figure 2). Super modules correspond to superstates in UML (Unified Modelling Language). See, for example, Larman 2002.

Modules communicate with each other. They can also perform independent continuous functions (e.g. environment perception), in which case they can produce information continuously for the other modules and the system. Some modules produce services on request; these include actuator modules, for example.

---

[1] The CLAWAR network was active in 1998-2005. It has been funded by the European Union as one of the first industry-led "thematic networks" investigating state-of-the-art technologies in Europe. The purpose of CLAWAR is to investigate and report upon all aspects of technology and systems relating to mobile robotics. (http://www.clawar.com)
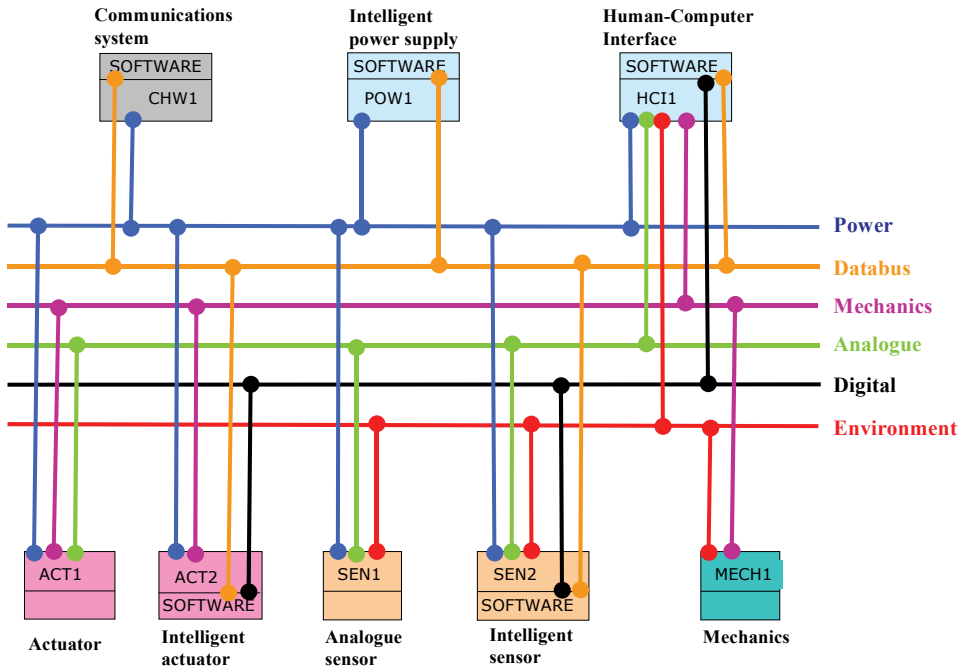
Fig. 1. Interfaces of modules (Virk, 2003b). The model has been developed in the CLAWAR network of the European Union.
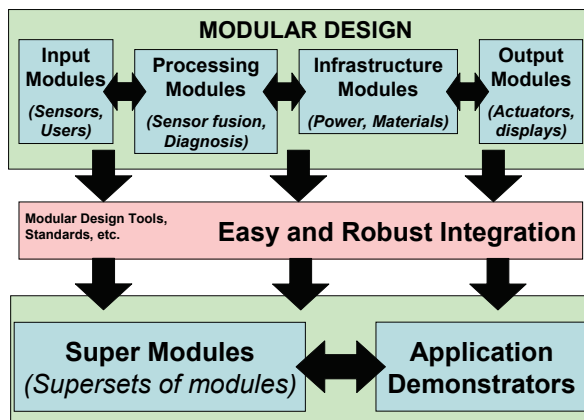

Fig. 2. CLAWAR design of modular mechatronic systems

Figure 3 presents system architecture typical of a modular service robot. Its idea is to demonstrate modules that are normally included in a service robot. Power (Pwr), Databus (Data), Mechanics (Mec), Analogue (Ana), Digital (Digi), Environment (Env) represent connections that might be needed for the modules. A CPU-unit is the central computer of the robot and all the modules inside it are software modules. The CPU-unit is not a module,

because it is a common platform that can offer services for several software or hardware modules. The presented modularity structure is based on the WorkPartner, but similar structures can be found from most of the current service robots. Minor differences can be found in how the modules are implemented the practice and in the nature of their interfaces.
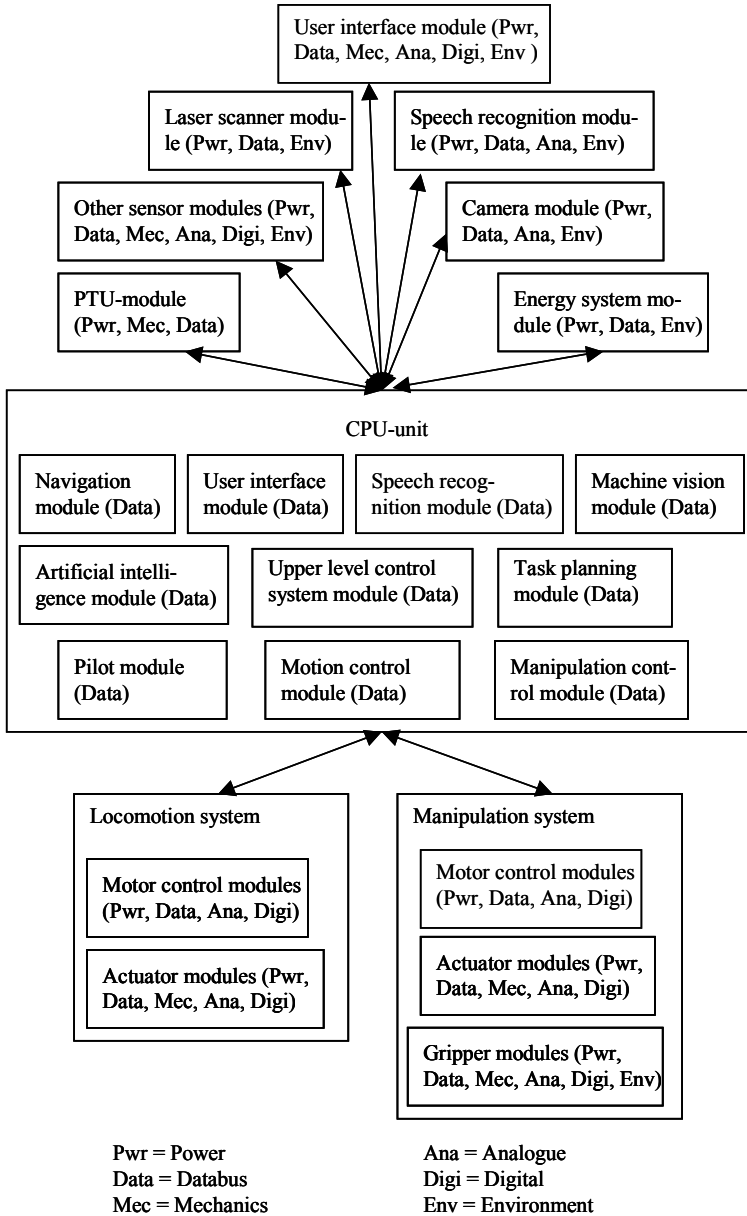

Fig. 3. System architecture of a modular generic service robot

There is a huge variety of different kinds of camera modules, data buses and mechanical connections available for the robot designer. From the standardisation work point of view, it is not so relevant that there are so many type of modules. It is far more important to focus on the standardisation of the connections, so that these modules can easily be connected to the robots.

2. Motivation for modularity

A significant amount of money has recently been invested in the research and development of service robotics around the world. Individual research groups, due to the lack of commercial off-the-shelf modules, have developed most of the applied subsystems separately. In an optimal situation, various plug-and-play modules would be commercially available. Utilisation of these kinds of modules would boost the service robot development and big savings could be gained. This would push the commercial supply and make cheaper prices possible, and again increase the demand. Thus, the coupling between demand and supply would operate in a healthy way and the service robotics industry could meet with real success. As in most branches of the engineering industry nowadays, in order to have a real success, the manufacturer has to pay a lot of attention not only to the design and manufacturing phases, but also to the commercialisation of optional accessories and maintenance services during the lifetime of the robots. The concept of modularity supports these operations perfectly. Additional features can be purchased and implemented easily due to the modular design, and the maintenance procedures are more straightforward when modularity has been applied to the product.

2.1 Needs in the area
The generalisation of consumer applications in service robotics still needs a lot of work. The biggest challenges are the complexity of the technology and finding proper applications. The complexity of technology increases the challenges in development, for example, as well as the development costs and the price of the end product. If the product is very complex and production amounts are low, the price will be very high. Standardisation of robot modules and their interfaces would help significantly. With standardised modules, a situation where there were different developers for robot modules and for robot applications could be achieved. The module market would have a larger volume, because the same modules could be used in different applications. Robot developers could focus on developing real applications, as they would not be forced to use so many resources for developing low-level techniques.

Developing and manufacturing non-modular service robots is technically very complex and challenging, so it costs a lot. Designing a modular service robot is much more straightforward.

2.2 Impacts of the modularity

Development speed of service robotics

High-level modularity would greatly boost the development speed of service robotics. Nowadays, service robot developers have to develop almost all subsystems and modules by themselves. If several modules with standardised interfaces were commercially available, it would help very much. Service robot developers could buy commercial modules from the

market, implement modules to the system and devote their main effort to the development of real applications. Much time and money would be saved.

Fault tolerance

Modularity greatly increases the fault tolerance of systems. The the system dependence of single modules should be minimised. So, if a single module were to be damaged, the system should notice it, while the other parts of the system still can do their own work tasks. In that case, the system could re-route the communications. The most critical modules should be doubled.

Commercialisation of service robots

Modularity would make the commercialisation of service robots much easier and would help to develop much better robots than would be possible without it; it would also assist in making cheaper service robots because of savings in development and manufacturing costs. Manufacturing costs would be lower due to reasonable module prices. Modularity would also make manufacturing faster and cheaper because there would be many fewer different parts in the assembly phase.

Commercialisation and markets of modules

The same commercial modules could be used in several different applications and they would also boost the marketing of service robots, so appropriate modules would reach real mass markets. This would also assist in obtaining low prices for modules.

2.3 Advantages and disadvantages of modularity

Modular structures and modularity would in any case provide many advantages to service robotics, such as, for example, all impacts mentioned in the previous chapter. Smaller design costs and bigger volumes per module leading to cost reduction and the possibility of making more complex devices are also remarkable advantages. Reliability of modules and thereby also reliability of service robots could improve, because more time and effort could be used in the development work. Modularity and standard interfaces would also make modification of the robots easier.

The only clear disadvantage of modularity is limitations in the design. This comes from a limited selection of modules. The robot has to be built using the modules that are available. It could have an effect on connections, control software, size, outlook and features of the robot.

3. State of the art in modularity of service robots

3.1 The history of modularity

Modularity has a long history. During World War II, Otto Merker investigated how modularity could be implemented in the construction of German U-Boats. It was the first prominent case where modularity was implemented in technology. (Arnheiter & Larren 2006)

Modular production – this term was used for the first time in 1965 by Martin Starr in the seminar article "Modular production – a new concept". He compared traditional mass production to modular production. (Arnheiter & Larren 2006)

In 1980, IBM launched modular architecture for personal computers. Nowadays, modularity is very strongly present in many sectors of industry. Modularity has been present in service robot development in some form from the very beginning, but no standardised service robot interfaces have been developed yet.

3.2 Interfaces of modules

Today there do not seem to be standard interfaces developed for service robotics. Standard interfaces are one of the most important things for promoting modularity in service robotics. Big technology companies like Sony and iRobot have their own interfaces between robot modules, but detailed information about these modules is company confidential.

In the research projects, it is reasonable to use standard interfaces, which exist in the other sectors. Several useful interfaces can be found in information technology. For example, USB, Ethernet, CAN-bus and RS-232 hardware data connections are used in many robots. Software interfaces for previous hardware interfaces are often manufacturer orientated and there are not any dominant standards.

3.3 Current standards

Several standards exist in industrial robotics currently. Most of them are safety standards for both stationary and mobile industrial robots. Driverless trucks are almost the only mobile industrial robots in use currently. For mobile service robots there are not any standards, while it is a very urgent need to establish at least safety requirements for them. Under ISO/TC184/SC2, an Advisory Group worked on standards for mobile service robots. The main goal of the group was to promote safety standards for mobile service robots.

3.4 Imaginary modular service robot

Designing modular service robots can be divided into individual phases. Figure 4 describes the starting point in service robot design. Robot, task and environment have tight interactions between each other. For example, if the task and environment are known, the robot should fill their requirements. The later phases from the modularity view are:

1. Listing of needed functions
2. Technology segmentation into subsystems
3. Segmentation into modules
4. Identification and searching of modules.

The list of needed functions includes all tasks that the robot should be able to do. The functions define the subsystems that are required for the realisation of tasks. Next, the subsystems can be divided into different modules. After this, a search should be made to ascertain whether commercial modules are available or whether tailor-made modules should be used.

Figure 4 presents a wide range of different kinds of commercial modules that could be used the service robots. It shows that the starting point is quite complex and that there are several different kinds of modules available. With these commercial modules, it could be possible to realise a service robot that could do simple locomotion and manipulation tasks. First, the wheeled mobile platform can be found from the modules. Stereovision, laser scanner, compass, gyro, arm and gripper modules can be mounted onto it. Hardware needs also a power module that offers the needed voltages for the listed modules.

Software is a much more challenging task to be solved using commercial modules. Mobile Vision Technologies GmbH sells the ERSP3.1 software module, which has been developed for the control of vision, navigation and user interface modules. However, central software, which controls the operation of the whole robot, has to be programmed. Central software controls all the other modules and is responsible for decision-making inside the robot.
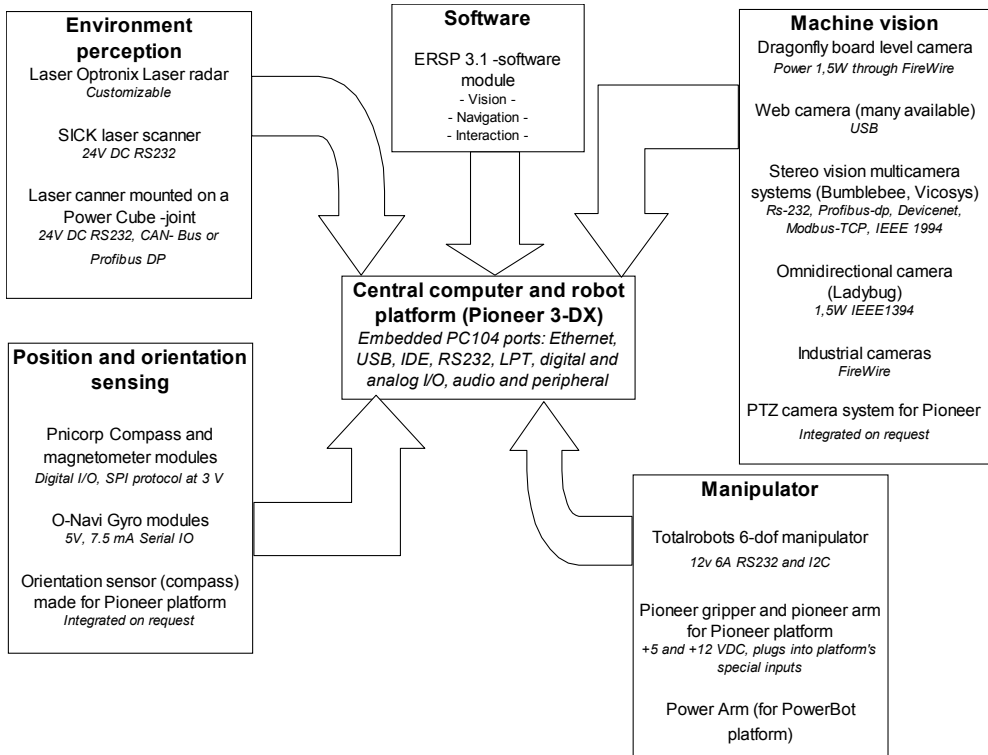
Fig. 4. Structure and interfaces of the imaginary modular service robot. The figure presents a group of potential modules and their interfaces.

As a conclusion, it can be noted that already several different modules are commercially available, but it is not possible to build a working robot out of these. There are big differences between the mechanical interfaces of modules too. So even mechanical assembling needs good workshop and a professional mechanic. Central software programming and work-task generation needs a large amount of work, even from the field's expert. The designer of the robot should program an advanced user interface too. Modular work-task generation, which is introduced in (Terho et al., 2006), would help a lot if it could be made available as a standardised product.

4. Case study: WorkPartner

This following analysis presents an example of modularity using WorkPartner service robot as an example (see Figure 5). WorkPartner is a multifunctional service robot for outdoor tasks. Some possible work tasks are garden work, guarding, cleaning, transporting lightweight objects and environment exploration including mapping. Mobility is based on a hybrid system, which combine the benefits of both legged and wheeled locomotion to provide at the same time good terrain negotiating capability and large velocity range. The working mechanism is a two-hand human-like manipulator, which can be used for a variety of tool manipulation tasks.

The robot is divided into functional subsystems that are relatively independent as to their software and hardware. The main subsystems are: Locomotion subsystem, Manipulator, Energy subsystem, Navigation and perception subsystem, Upper level task control and monitoring system and Human-machine interface.

Fig. 5. WorkPartner picking up litter

Next, two of the subsystems of WorkPartner are described in a more detail. The purpose of the analysis is to demonstrate the ideas of modular design through a case example. More information of the modular design of WorkPartner can be found in (Ylönen, 2006).

Manipulator sub-system

The platform is equipped with a two-hand manipulator system. The manipulator is made of aluminium and weighs only about 30 kg. The manipulator can handle loads of up to 10 kg. A two-hand human-like and human-size manipulator was chosen because similarity to human tasks and close co-operation with people are required.

The manipulator consists of a 2-DOF (degrees-of-freedom) body, two 3-DOF arms and a 2-DOF camera and distance measuring laser pointer head. The manipulator's body is joined to the platform with two joints that allow orientation in horizontal and vertical directions. Fig. 6 describes modules of the manipulator and their connections. Mechanics modules are connected with joints to each other, except the head, which is connected to the pan-and-tilt-unit. As an exception of the mechanics modules, wrist modules have an additional connection to the environment, because they can be used in handling objects. In the manipulator there are 10 supermodules, which are equipped with DC-motors, harmonic type gearings, mechanical breaks (wrists and gripper excluded), potentiometers and motor

drives. Each motor drive controls one or more joints. The supply voltages for the joint motors are 48 V, 12 V for the brakes and ±12 V for the gripper. Motor drives are based on Texas Instrument DSP processors. Drives are equipped with a CAN interface, analogy inputs and digital inputs and outputs. Message structure on the CAN-bus includes message type and the data itself.
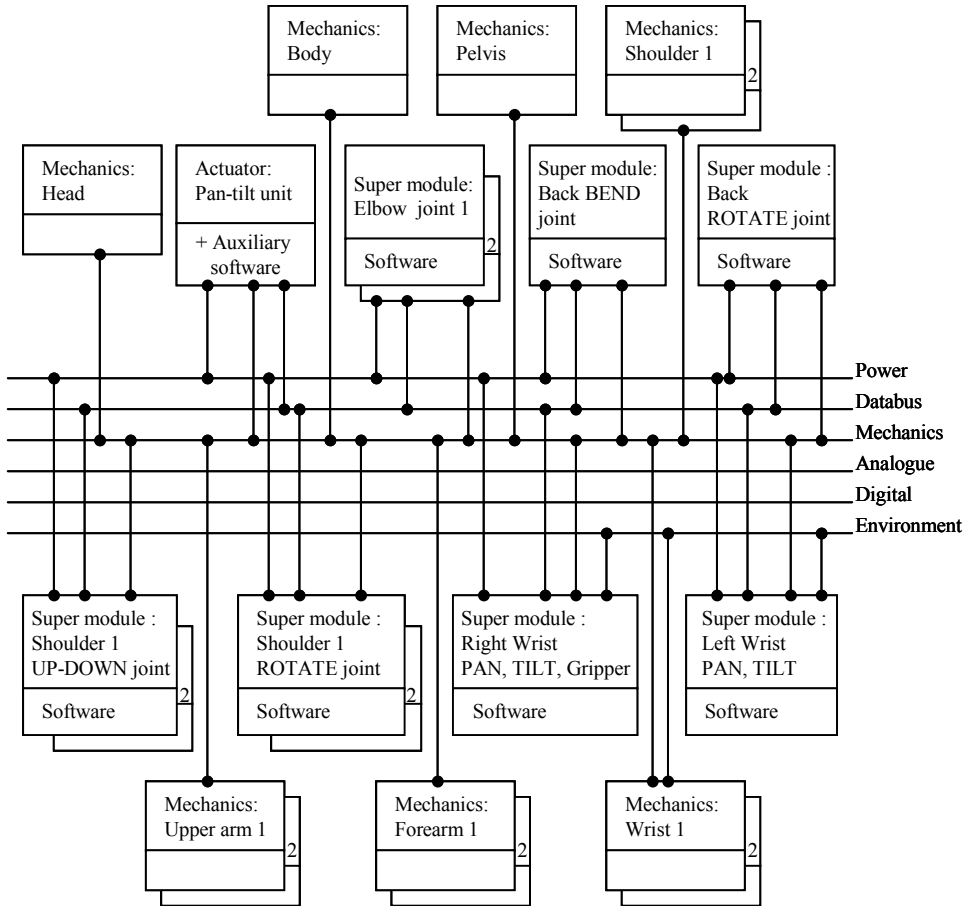

Fig. 6. Modules of WorkPartner's manipulator according to modified CLAWAR model

Human Machine Interface

The Human-Machine-Interface (HMI) of WorkPartner is designed for operators working in parallel to the robot and, in some cases, co-operating very closely with it. However, the remote control mode is also possible when the robot is accessible via the Internet. Symbolic representation in communication is based on the underlying idea that both the operator and the robot perceive the same environment and interpret it through a commonly understood virtual model. The model is a simplified 3D description of the environment, which includes the objects relevant for performing work tasks. Modules of the HMI are presented in Fig. 7.
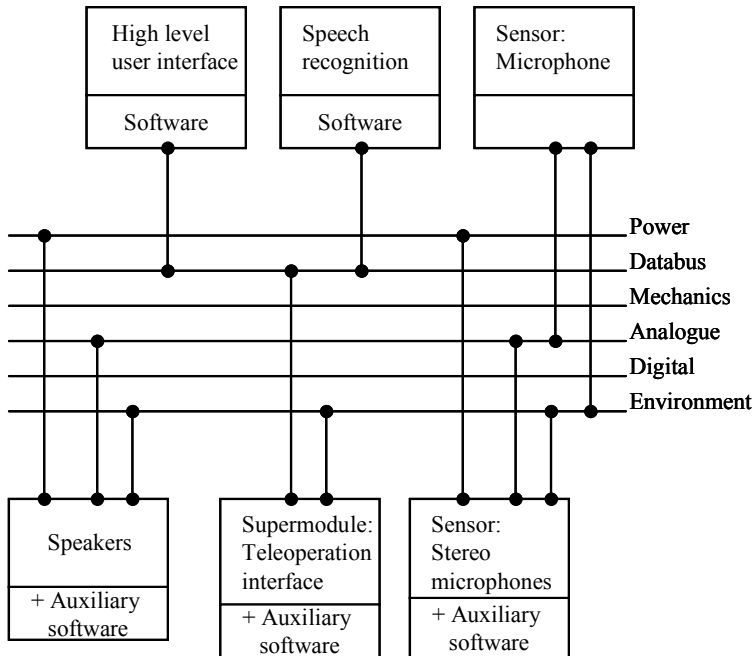
Fig. 7. Modules of WorkPartner's human-machine interface, according to the modified CLAWAR model

Speech synthetisation software generates a signal for the speakers. Speech recognition software, currently Microsoft Speech, interprets microphone signal as words.

Stereo microphones are located at the shoulders of the manipulator. They are used for detecting the direction of sound. When having a "conversation" with the human operator the robot turns its head towards the direction of the sound. A key part of the HMI is the wearable teleoperation device, which is classified as a super module. The teleoperation interface is presented in Fig. 8.

It can be used for teleoperation, teaching of movements and for giving commands by gestures. It is connected to the user interface laptop through an RS-232 serial bus. Software calculates arm orientations and generates different kinds of control commands.

5. Actions for improving modularity

For some time, there has been a strong demand for common mobile robot standardisation. Current robotics standards have been focused on the stationary industrial robots. Safety issues in the industrial robotics have been based on the fact that people cannot go to the working area of robots. Therefore, these standards cannot be utilised for mobile robots, whose essential idea is to act and work in the same environment as people.

Some safety standards developed for automatic forklifts could be adapted to service robotics also. These standards define, for example, that maximum locomotive speed is 1 m/s. For stopping the vehicle, it has been stipulated that a person must not in any case be run over. Standards stipulate also that, even if a person lies on the driving way, the automatic truck has to observe him/her and stop early enough (SFS-EN 1525).

Fig. 8. Teleoperation of the robot with the wearable teleoperation interface.

ISO established the Advisory Group on Standards for mobile service robots. The group worked under ISO/TC184/SC2. The work is introduced in (Virk 2006). The size of the group was about 30 experts from around world and there were also delegates from IFR (International Federation of Robotics), IEEE Robotics and Automation Society (IEEE is Institute of Electrical and Electronics Engineers) and IEC/TC 44 (International Electrotechnical Commission / Technical Committee 44). The goal was to start the project that defines safety standards so that these standards can be taken into use. The first goals of service robotics standardisation are to define terms relating to how service robots should act. A big problem at the moment is that there is not this kind of standards, so it is not possible to commercialise service robots that fill such standards. From this it follows that a commercial mobile service robot would not be legal. After safety standards for mobile service robots are ready, robot manufacturers can put service robots that meet these standards, and are thus legal, onto the market.

Standardisation work has to be continued with respect to module interfaces too. Standards, which have been developed in the other sectors, should be used as much as possible in software interfaces. For example, software standards in Internet-related applications, the car industry and work machines. Complementary standards can be developed in addition.

Modularity should also be improved in co-operation networks, which could be sponsored by EU or which could work under a currently existing robotics association. Information about modularisation benefits should be shared so that as many as possible become highly motivated towards modularity development. This topic should be taken note of in education too. Different kind of competitions is good way to teach new things. Public example cases could show in reality the benefits that can be derived from modularity.

6. Conclusion

Service robotics has been an active research and development area for more than ten years. However, most of the subsystems needed to create a fully operational service robot have been developed separately by individual research groups. Modularity could be the key factor to success by reducing the development time and by increasing the technical advancement level. Thus, it could directly help to gain significant savings on the total development costs. The reliability of the subsystems would also be improved, because more time and money could be used for that. Modularity, and especially the work with standard interfaces, would also make the modification of these robots much easier.

The case study in which the concept of modularity has been studied is that of the WorkPartner robot. This is an advanced service robot, in which most of the important modules in service robotics have been integrated together successfully. The utilisation of modularity in this project was essential due to the high hardware and software complexity of the robot required to complete the given working tasks. This case example also demonstrates very clearly that significant savings in the development time and costs could be gained if commercial modules for service robots were available.

Standardisation work is needed for boosting modularity. If the interfaces of the modules could be standardised, this would create a far better operational environment for the current and future module producers. Safety standards are required before more generic service robots can be released to the wider market. Currently, there are safety standards for industrial robots and automatic forklift trucks, but these standards would require at least a considerable amount of modification before they could be more widely applied to the field of service robotics.

Modularity is clearly essential for the successful development of reasonably priced service robots. Right prices will boost the demand for service robotics and that, in turn, will direct more effort to service robotics development in general, and thus a positive cycle would evolve. One could safely state that modularity will be one of the key factors that is required to guarantee that the service robotics industry will reach global success in the near future.

7. References

Arnheiter, D. & Harren, H. (2006). Quality management in a modular world, *The TQM Magazine*, Emerald Group Publishing Limited, Volume 18, Number 1, 2006, pp. 87-96, ISSN 0954-478X.

Gates, B. (2007). A robot in every home, *Scientific American*, January, 2007, pp. 44-51.

Larman, C. (2002). *Applying UML and Patterns*, Prentice-Hall Inc., 2002, 627 p.

SFS-EN 1525 (1997). *Safety of industrial trucks. Driverless trucks and their systems*, European Standard EN 1525:1997.

Terho, S. ; Heikkilä, M. ; Taipalus, T. ; Saarinen, J. & Halme, A. (2006). A framework for graphical programming of skilled tasks with service robots, *Proceedings of 9th International Conference on Climbing and Walking Robots (CLAWAR 2006),* Brussels, Belgium, September 12-14, 2006.

Virk, G.S. (2003a). CLAWAR Modularity for Robotic Systems, *The International Journal of Robotics Research*, Sage Publications, Vol. 22, No. 3-4, March-April 2003, pp. 265-277.

Virk, G.S. (2003b). CLAWAR Modularity: The Guiding Principles, *Proceedings of 6th International Conference on Climbing and Walking Robots (CLAWAR 2003),* pp. 1025-1031, Catania, Italy, September 17-19, 2003.

Virk, G.S. (2006). Standards for mobile service robots, *Proceedings of 9th International Conference on Climbing and Walking Robots (CLAWAR 2006),* Brussels, Belgium, September 12-14, 2006.

Ylönen, S. (2006). *Modularity in Service Robotics - Techno-economic Justification through a Case Study*, Doctoral thesis, Helsinki University of Technology, ISBN-13 978-951-22-8502-0, Espoo, Finland.