Researches on Autonomous Mobile Robot's Navigation Using Topological Approach and Web Monitoring System

웹 모니터링 시스템과 토폴로지컬 방법을 이용한 자율이동로봇의 주행에 관한 연구

Advisor: Prof. Doo Sung Ahn

Gyu Jong Choi

A thesis submitted in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy

in Department of Mechanical Engineering, The Graduate School,
Pukyong National University

February 2006

Researches on Autonomous Mobile Robot's Navigation Using Topological Approach and Web monitoring System

A dissertation
by
Gyu Jong Choi

Approved by:

Chairman Prof. Won-Chang Lee

J.Y.Kim
Member Prof. Jin-Young Kim

Member Prof. Shang-Woon Shin

Member Prof. Young-Seok Jung

Member Prof. Doo-Sung Ahn

February 28, 2006

Contents

List of Figures

List of Tables

List of symbols

r_i	range value or i_{th} sensor
$\theta_{incidence}$	incidence angle of ultrasonic sensor
$r_{\min}$	minimum sensing value
$r_{\max}$	maximum sensing value
X_r	robot position vector
d_{obs}	obstacle orientation
d_r	robot direction
${}^r\hat{X}_{obs}$	obstacle position vector respect with robot
${}^r\widehat{v}_{obs}$	obstacle velocity respect with robot
d_t	target direction
v_r	robot linear velocity
w_r	robot angular velocity
$v_{\max}$	maximum linear velocity of robot
$w_{\max}$	maximum angular velocity of robot
d_m	movable direction
M_{dist}	moving distance
n	number of data
x_n, y_n	sensed coordinate values about obstacle
E	estimation error
a, b	estimated coefficients for least square approximation
S_{xx}	summation of x_i^2
S_x	summation of x_i

S_{xy}	summation of x_iy_i
S_y	summation of y_i
$r(t)$	first link of manipulator
l	second link of manipulator
θ_1	first joint angle of manipulator
$\theta_2(t)$	second joint angle of manipulator
J	jacobian of two link manipulator
q	joint angles of manipulator
r_e	difference between local minimum values
d_e	direction error between d_r and d_t
K_1, K_2, K_3	gain values
r_{safe}	safe range
r_{env}	range according to distributed density of obstacles
s	state values
A, B	triangular fuzzy membership function
d_{diff}	difference between d_t and d_m
d_{th}	threshold value of d_{diff}
N_i	ith node
N_{nbr}	neighbour node
E_{ij}	edge between N_i and N_j
T	sampling time for internet control
T_d	internet latency
T_p	sampling time by command filter

Researches on Autonomous Mobile Robot's Navigation
Using Topological Approach and Web Monitoring System

Gyu Jong Choi

Department of Mechanical Engineering, The Graduate School,
Pukyong National University

Abstract

To build a human-like topological map about the unknown environment, we propose a new localization and map-building method similar to human being's memory in the navigation problem. Human being seems to solve a problem through four processes(*Exploration process, Decision process, Behavior process and Learning process*). We define the processes as human being's capability for solving problems. In this paper, we will try to solve the navigation problems by transferring this human being's capability into a mobile robot. Firstly, in the *Exploration process*, a mobile robot collects the environmental information and builds the map using the *Graph* and *Turn side selector*. Secondly, in the *Decision process*, a mobile robot selects a proper action based on the analysis of this information and the generated *Graph*. Thirdly, in the *Behavior process*, the selected action is implemented in accordance with the output of Fuzzy Inference System. Finally, in the *Learning process*, the parameters of *Graph* are updated by visiting repeatedly.

To build the topological map about the unknown environment, we will utilize the ultrasonic sensors. An ultrasonic sensor inherently has the range error due to the specular reflection. To decrease this error, we estimate the obstacle states(position and orientation) using the local minimum sensor values and Jacobian. Estimated states are used to avoid the obstacles and build the topological map similar to the type that human being memorizes a map. When a mobile robot is faced with three problems(corner way, cross way and dead end), it can sense the movable directions using FIS(Fuzzy Inference System). Among these directions, it can also select the target direction using 'Turn Side Selector' such as binary decision tree.

The communication system of mobile robot used in the implementation is based on the wireless communication system. Six ultrasonic sensor modules are respectively equipped with the bluetooth modules and a mobile robot can get the obstacle states via MSP(Multi Serial Port). The mobile robot constructed with the server system is equipped with the wireless LAN to transfer the information to the clients. To control a mobile robot in remote area through TCP/IP, we also have to choose the proper control architecture which enable to send the commands to the mobile robot. All the clients can real-timely monitor the states of mobile robot and send the commands using the robot's IP. So the mobile robot system is configured with complete WPAN(Wireless Personal Area Network) and basic ubiquitous system so that the clients can get the desired information regardless of any time and any place.

In this paper, we have accomplished three simulations and three implementations. First is to find the states of the obstacles using the proposed algorithm. Second is to build the map about the unknown environment using the states of the obstacles. So, when it is given a start position and a final position, the mobile robot navigates into the target position based on the builded map. Here, all the commands and information are respectively transferred and received through TCP/IP. We will show that this approach is very effective for the navigation problems through the simulations and implementations.

I . Introduction

From the manipulator that used to a major means of production in the factory automation, the needs of human being friendship robot has been raise according to the rising quality of our life. Preferentially, *Human being friendship robot* have to explore and learn information about the unknown environments by itself and also decide and behave by learned information. These four capabilities will be essential part in the future of robot industry. Nowadays, these mobile robots such as the clearance robot, the guidance robot and the surveillance robot that have four capabilities(exploration, learning, decision and behavior) were already developed, but these are poor technique on a commercial viewpoint.

In order to realize the pre-mentioned four capabilities, The robot has to learn the information about the unknown environments and localize its position by itself. We generally define that the former is map building, the latter is localization[1~9]. When a mobile robot navigates, it may be faced with several problems in the unknown environment. That is, a mobile robot has to explore and learn the states about the unknown environment and it has to effectively avoid the collisions when the obstacles are distributed around the robot. A robot has to also select the movable directions at cross way and learn it. To intelligently and autonomously navigate to the target position in the unknown environment, these four capabilities are essential.

1. Background

When an event was occurred, a human being makes a proper decision through the exploration process and takes a real action. After learning by the evaluation of the action, he/she will take a best action in case of similar situation. These are divided into four processes. That is *Exploration process*, *Decision process*, *Behavior process* and *Learning process*. We will define these four processes into *problems-solving capability of human being*. Thus, we focus in transferring the human being's capability into a mobile robot in the navigation problem[10~20].

There have been developed a number of possible solutions to the robot navigation problem. The representative algorithms are EPF(Electrostatic Potential Field)[21~24], VG(Vornoi Graph)[25~30] and Cell decomposition[31]. However, these methods are basically different from approaches which use the problems-solving capability of human being and result in the suppression of the mobile robot's autonomous property. For example, human being memorizes the environment information qualitatively rather than quantitatively. Most of previously mentioned algorithms overlooked this point. To incorporate the problems-solving capability of human being into navigation problems, the new type of environment modeling method, which can memorizes the information qualitatively, is required.

To solve these problems, two questions must firstly be analyzed. The first question is about how to express the unknown environment exactly. The second is about how to mimic a human being's memory system similarly. The key to the solution of the first question is to memorize a lot of information about environment. The key to the solution of the second question is just to memorize the feature values of the environment and express relationship of that feature values. For these purpose, *Graph*, which can memorize a variety of data efficiently, is used in this paper. The *Graph* consists of *nodes* and *edges*. The map building about the unknown environment is performed with the generation of nodes with features(passage way, corner way, cross way and dead end) and the generation of edges connecting nodes. Nodes and edges can memorize a variety of feature values about the passages. In this paper, to efficiently verify the proposed approach, we will restrict that a node memorizes the neighbor nodes and directions and distances towards those nodes. This approach can apply the dynamic environment and also decrease the quantity of memory and moreover, increase the autonomous property of the mobile robot[32~34]. Using the *Graph* and *Turn side selector*, we will build the map about the unknown environment in the navigation problem. To solve the localization problem, we will estimate the current position of a mobile robot using the *Sub-graph matching method.*

2. Motives

A mobile robot means that a robot can change its position by itself. A mobile robot includes the multi-ped robots similar to the bug robots, the biped robots similar to the human and the wheeled mobile robot. That is, the mobile robot can move any position by its actuator that can change the mobile robot's current position. Therefore, these robots basically have to learn the information about the unknown environments and localize its current position by itself. We have defined that the former is map building, the latter is localization. In order to realize these implementation, a mobile robot needs to the pre-mentioned four capabilities. When a mobile robot explores the unknown environment by means of these four capabilities, it can find the movable directions at cross way and memorize the movable directions and the moved distance at cross way and avoid the collision when the obstacles are distributed around the robot. A robot has to also select desired directions at cross way and learn it. Therefore, Through the first visitor's navigation about the unknown environment, we will apply to the mobile robot.

3. Objectives

To implement the map building and localization based on four capabilities, a mobile robot can recognize the unknown environment with the reliable information about obstacles. In general, a mobile robot uses the vision sensor[35~46] that acquires the image information in the environment and ultrasonic sensor(or infra radiation sensor) that measures the distance to the obstacles. In special, the ultrasonic sensor that measures the distance using the ultrasonic can accurately detect more environment states than the vision sensor in the poor condition of environment(outdoor). It has been used in a mobile robot because of its compact size and low expensive. And it also has the advantages that can measure the distance to obstacles regardless of its surface and the measurable distance is relatively long. Therefore, using the mobile robot equipped with six ultrasonic sensor modules, we propose the simple and topological map building method in the unknown environment.

However, due to the measuring method based on the time of the flight principle, there always exists the distance error caused by the specular reflection. That is, if the incidence angle between the ultrasonic direction and the obstacle's surface as shown in figure 1.1 is more big, the value sensed by sensor, r_6, would be more long than real value. Therefore, to solve this problem, we will utilize the local minimum value(r_5) among six sensor data and jacobian of manipulator. The local minimum values are free to the specular reflection because they are big signal strength, small reflection angle and there is one at least. We also estimate the states of the obstacles using jacobian. Using this information, we can implement the collision avoidance and the topological map building.

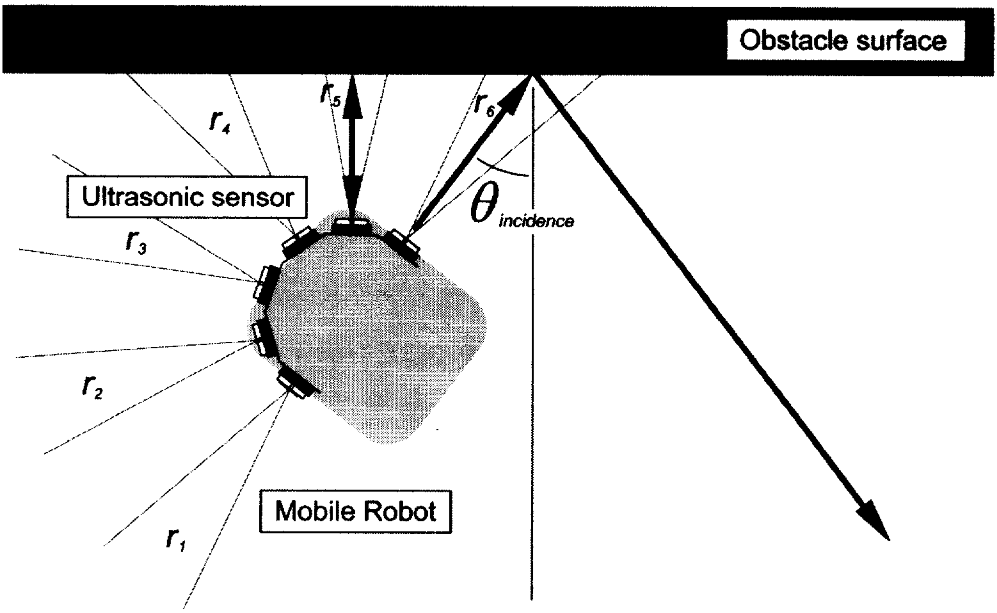


Fig. 1.1 The effect of specular reflection of ultrasonic sensor

Ⅱ. Human-like Navigation

When a human being memorizes an environment map, he/she doesn't numerically save all the environmental information. A human being just remembers the important feature values of the environment. However, most of the existing approaches are static in the environment construction and require much memory. These approaches are also essentially different from the *problems-solving capability* of human being and result in the suppression of the mobile robot's autonomous property. So, we have assumed that human being follows four processes(*Exploration process, Decision process, Behavior process and Learning process*) for solving navigation problem in this paper. Our purpose is to transfer these four processes into a mobile robot and mimic human being's capability in a similar way.

1. Exploration Process

When someone tries to find his/her path to a destination in new environments, he/she memorizes firstly the distinctive places as many as possible and becomes aware of the environment and his/her position in it from the basis of the *rough feeling,* not accurate numerical data about the distinctive places. To incorporate these characteristics into our approach and build an unknown environment map, we use the *Graph* in building map of unknown environments. *Graph* consists of *nodes* and *edges.* When a mobile robot is navigating, a node is assigned at distinctive features and edge is used to connect these nodes. *Graph* is increasingly updated by *Exploration process* and a robot can easily select an optimal path using *Graph* and *Minimum cost method.*

2. Decision Process

Once an environment map is generated by *Graph*, then the mobile robot selects the desired direction using the *Turn side selector*. To reach the final target position, there exist many nodes(via-points). A mobile robot can must pass through these nodes and arrive at target position. However, there are two problems to be solved. First problem is to find whether a mobile robot pass through nodes or not. Second is that if a mobile robot lost its way on the optimal path, it has to find the nearest via-point. To solve these problems, we use *Sub-graph matching method* so that a mobile robot can estimate a current position.

3. Learning Process

An environment information is updated through repetitive visiting of the same place. This process makes a mobile robot to more exactly find out an environment information that was not found by previous visits. Through the continuous updates of parameters, a mobile robot can more exactly construct an environment information. That is, a mobile robot can learn an environment through continuous updating of nodes and edges.

4. Behavior Process

An actual behavior is performed in this process. Behaviors are divided into two types. First is *consciousness behavior* resulting from the actions decided according to the proposed processes. Second is *unconsciousness behavior* which is independent of the proposed processes and generated unconsciously only when an unexpected event occurs. The former decide an action which enables the mobile robot to pass through vertexes. The latter is necessary for avoiding unexpected obstacles during navigation.

The four processes are closely connected as a mobile robot is navigating. The mobile robot equipped with the six range sensors can recognize the obstacles. *Turn side selector* can build the environmental map using the sensed values. And then, *Graph parameters* are updated through the *Exploration process* and *Learning process*. It also accomplishes with the *Decision process* in the cross way.

5. How to Transfer into Robot

To incorporate the *problems-solving capability* of human being into navigation problems, the new type of environment modeling method memorizing the information qualitatively is required. To deal with these problems, the following question must firstly be answered : how are the model of robot's environment and a human being's memory system represented? Actually, several world model representations have been used by their degree of abstraction : from raw data maps with no abstraction, to metric maps with geometric features and their spatial dependencies, via topological maps holding relationships between distinct locations. For our purpose, a new type model is proposed based on graph theory. Although 'Graph' in our proposed method is apparently similar to general graphs, it has fundamental differences in their development and algorithms. *Graph* proposed in our paper consists of *nodes* and *edges*. The nodes and the edges can memorize a variety of feature values about many paths in the environment. The building of an environment map is performed with the generation of nodes memorized the features and the generation of edges connecting nodes. In this paper, to efficiently verify the applicability of the proposed approach, we will restrict that a node merely memorizes the direction and the distance towards the neighboring nodes. This approach can apply even the non-stationary environments, demonstrating the memory compactness and the extended autonomous properties of the mobile robot. In this paper, for the purpose of building maps and localization, the algorithms called by *Turn side selector* and the *Sub-graph matching method* are also proposed.

Ⅲ. Overall System Configuration

A mobile robot consists of the *sensor system* that convert the ultrasonic sensor signal into the distance value and the *dead reckoning system* that estimate the current position using the robot's velocity value. And the overall system is divided with two levels. One is the collision avoidance and another is the map building and localization. The collision avoidance belongs to the low level and high level includes the intelligent parts[47~50]. Using these parts including the decision, exploration, learning, a mobile robot can build the map about the unknown environment. The overall system configuration is shown in figure 3.1.

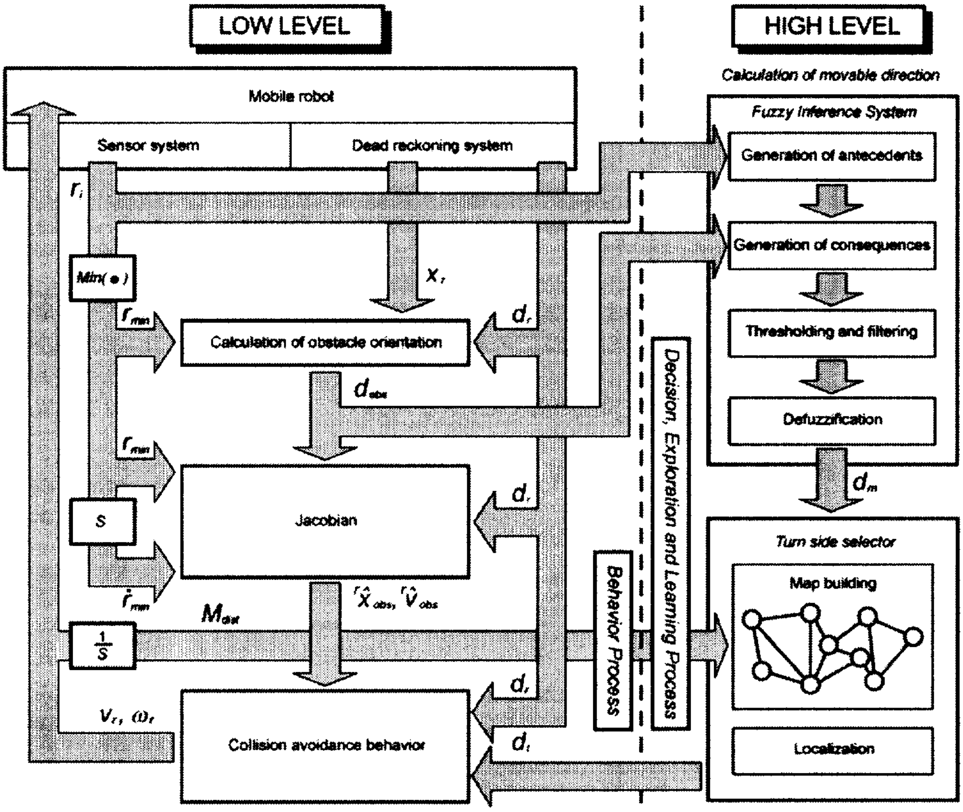


Fig. 3.1 Overall system configuration

1. Mobile Robot System

A mobile robot consists of the sensor system that is low expensive SRF04 ultrasonic sensors(r_1~r_6) and the dead reckoning system. As shown in figure 3.2, the equipment position is the mobile robot's front to acquire more precise information. The dead reckoning system estimate the current robot's position and direction. When the mobile robots change its orientation, it is categorized several types according to the mobility how to move freely and the steerability how to steer freely. The front two wheels are operated by the stepping motors and the *instantaneous center of rotation*, which is intersection point of the extend line of wheel rotation axes, can be generated only one using the *off-centered orientable wheel* of rear wheel. So the mobile robots can freely steer the arbitrarily direction.

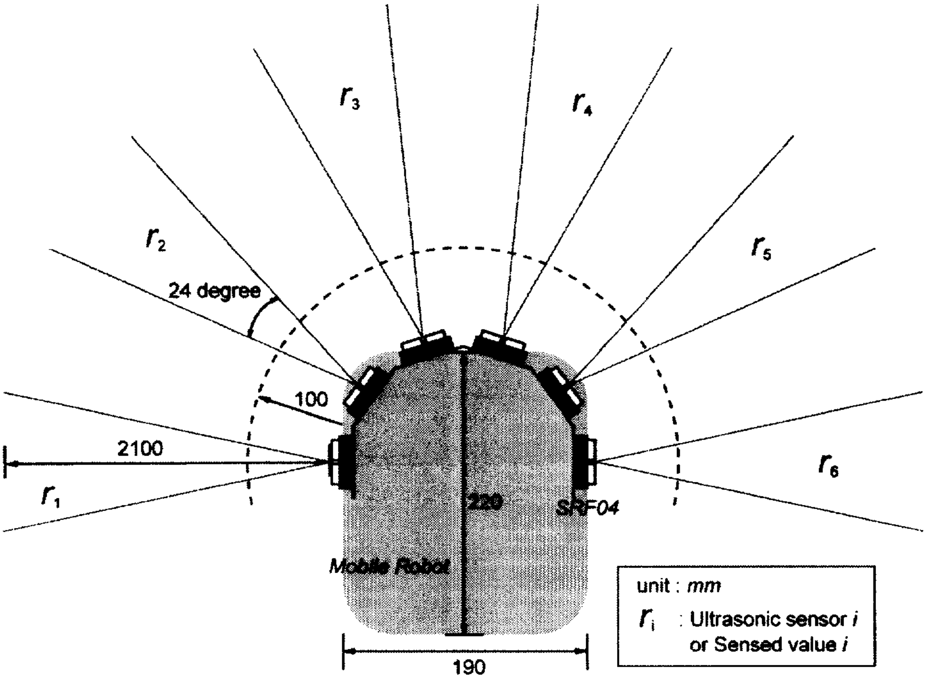


Fig. 3.2 Architecture of mobile robot system

2. Sensor System

A mobile robot consists of the sensor system that is low expensive SRF04 ultrasonic sensors(r_1~r_6) and the dead reckoning system. We have limited the maximum sensing distance of ultrasonic sensors with 2100mm and its minimum sensing distance with 100mm. As shown in figure 3.2, the equipment position is the mobile robot's front to acquire more precise information. This is similar to human's eyesight. The dead reckoning system estimate the current robot's position and direction using the stepping motor's velocities in both wheels.

3. Environment Information

In real world, for example, a human being memorizes the environment information *qualitatively* rather than *quantitatively*. In other words, when someone tries to find his/her path to a destination in new environment, he/she memorizes firstly the distinctive places as many as possible and becomes aware of the environment and his/her position in it from the basis of the *rough feeling*, not accurate numerical data about the distinctive places. Most of previously mentioned algorithms have overlooked this point. Main idea of our method is to incorporate this qualitative property of human being's perception into our navigation algorithms of a mobile robot.

Ⅳ. Navigation Algorithms

1. Calculation of Obstacle Orientation

In order to estimate the obstacle's states around the robot, it is essential that the robot's current position(X_r), direction(d_r), and the distance(r_i) to the obstacle. Using this information, we calculate the obstacle positions respect with the world coordinate system and then the obstacle positions are stored on memory in realtime. Firstly, n sensor data included the current local minimum value($r_{\min}$) as shown in figure 4.1 among the overall sensor data stored in *Data storage* as shown in figure 4.2 are extracted and then obstacle directions are estimated using the *linear least squares approximation* such as equation (1), respect with the world coordinate system.

$$\begin{aligned} E &= [f(x_1 - y_1)]^2 + [f(x_2 - y_2)]^2 + \cdots\cdots\cdots [f(x_n - y_n)]^2 \\ &= [ax_1 + b - y_1]^2 + [ax_2 + b - y_2]^2 + \cdots\cdots\cdots + [ax_n + b - y_n]^2 \end{aligned} \tag{1}$$

Here, when $\frac{\partial E}{\partial a} = 0$, $\frac{\partial E}{\partial b} = 0$, gradient is the states of obstacles. Followings are necessary values to calculate the states.

$$S_{xx} = \sum_{i=1}^{n} x_i^2 \tag{2}$$

$$S_x = \sum_{i=1}^{n} x_i \tag{3}$$

$$S_{xy} = \sum_{i=1}^{n} x_i y_i \tag{5}$$

$$S_y = \sum_{i=1}^{n} y_i \tag{6}$$

$$aS_{xx} + bS_x = S_{xy} \tag{7}$$

$$aS_x + b(n) = S_x \tag{8}$$

,where (x_i, y_i) is the obstacle position in the world coordinate system and $'a'$ is a state of obstacle, this is calculated by following equation (9).

$$a = \frac{nS_{xx} - S_x S_y}{nS_{xx} - S_x S_x} \tag{9}$$

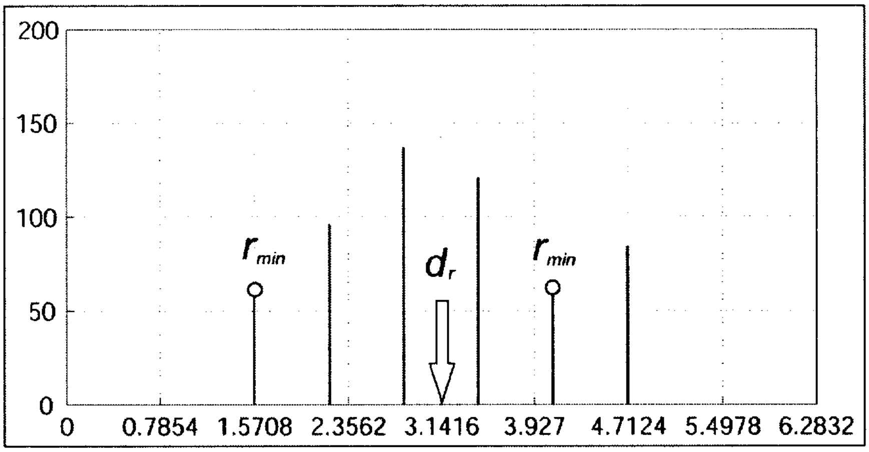


Fig. 4.1 Finding of local minimum values

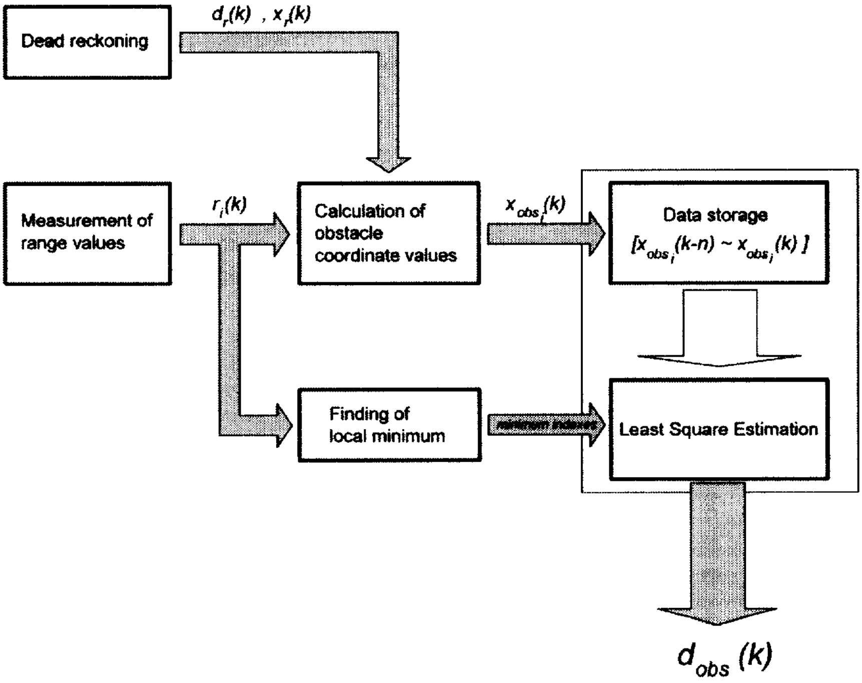


Fig. 4.2 Calculation of an obstacle orientation

2. Collision Avoidance using Jacobian

In this paper, we apply the *jacobian method* to efficiently avoid the obstacles through the acquisition of environment information around the robot. As shown in figure 4.3, the coordinate system is divided into the world and the robot coordinate system and we consider each sensor with two-link manipulator. And then, using the jacobian about the variables named in figure 4.4, we can estimate the position and velocity about the *interesting point.* Here, $r(t)$and l are considered with the links of manipulator. The length of the first link is the sensed distance to an obstacle and that of the second link is variable according to the current velocity of the mobile robot. That is, in case of the high speed of the robot, l is increased, otherwise l is decreased to precisely control the robot motion. Joint angle, θ_1, is the attached angle of sensors and that is constant due to the fixed position respect with the robot. And through the calculation of jacobian using $\theta_2(t)$ transformed the world coordinate system into the robot coordinate system, we can estimate the approach velocity($^r\widehat{v_{obs}}$) and position($^r\widehat{x_{obs}}$) respect with robot coordinate system.

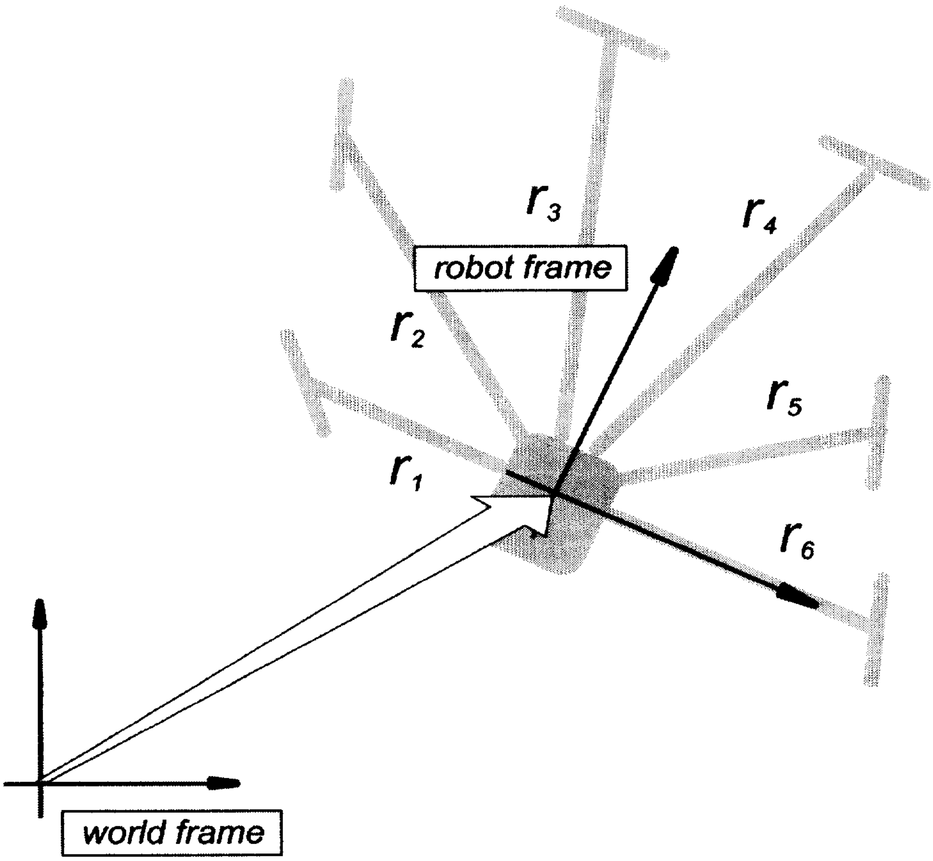


Fig. 4.3 Mobile robot equipped with six two-link manipulators

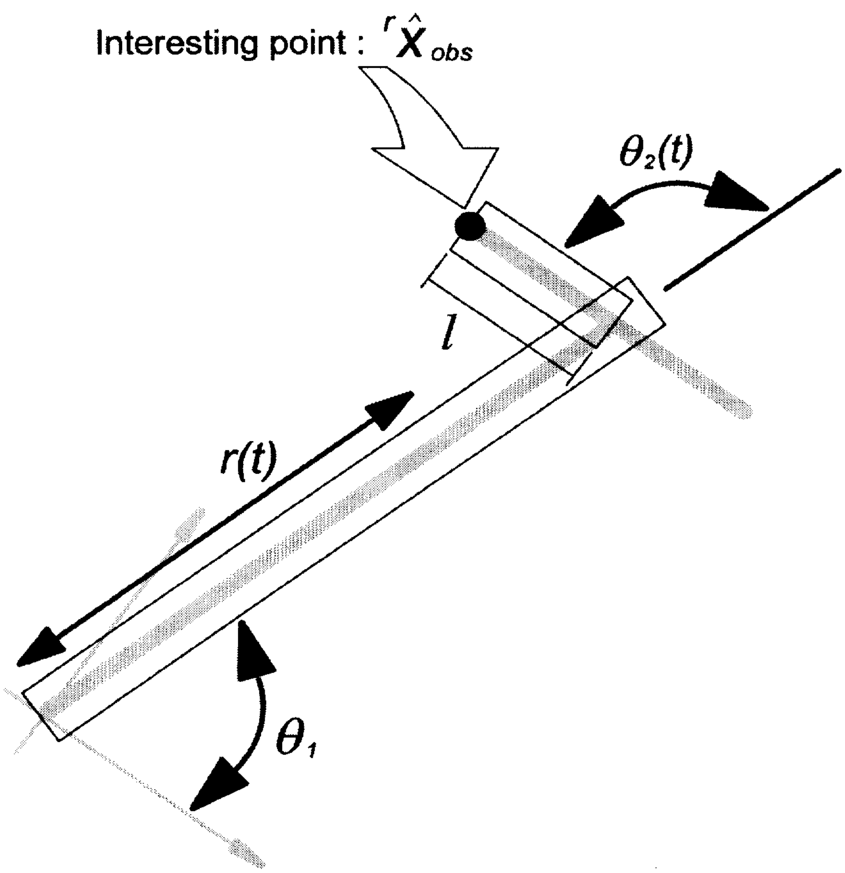


Fig. 4.4 Jacobian parameters for two-link manipulator

Following equations (10) ~ (13) show the calculation process of jacobian.

$$\begin{aligned} x &= r(t) \cdot cos(\theta_1) + a \cdot cos(\theta_1 + \theta_2(t)) \\ y &= r(t) \cdot sin(\theta_1) + a \cdot sin(\theta_1 + \theta_2(t)) \end{aligned} \tag{10}$$

$$\begin{aligned} \dot{x} &= \dot{r}(t) \cdot cos(\theta_1) - a \cdot \dot{\theta}_2(t) \cdot sin(\theta_1 + \theta_2(t)) \\ \dot{y} &= \dot{r}(t) \cdot sin(\theta_1) + a \cdot \dot{\theta}_2(t) \cdot sin(\theta_1 + \theta_2(t)) \end{aligned} \tag{11}$$

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} cos(\theta_1) & -a \cdot sin(\theta_1 + \theta_2(t)) \\ sin(\theta_1) & a \cdot cos(\theta_1 + \theta_2(t)) \end{bmatrix} \cdot \begin{bmatrix} \dot{r}(t) \\ \dot{\theta}_2(t) \end{bmatrix} \tag{12}$$

,where

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \dot{X},$$

$$\begin{bmatrix} cos(\theta_1) & -a \cdot sin(\theta_1 + \theta_2(t)) \\ sin(\theta_1) & a \cdot cos(\theta_1 + \theta_2(t)) \end{bmatrix} = J,$$

$$\begin{bmatrix} \dot{r}(t) \\ \dot{\theta}(t) \end{bmatrix} = \dot{q}$$

$$\dot{X} = J \cdot \dot{q} \tag{13}$$

J is utilized to estimate the states of obstacle.

To avoid the obstacles and arrive at the target position, we utilize following equation (14).

$$w_{sum} = K_1 \cdot r_e + K_2 \cdot d_e + K_3 \cdot {}^r\hat{v}_{obs} \tag{14}$$

In equation (14), using three information, the angular velocity(ω_r) of mobile robot is calculated to avoid the obstacles as shown in figure 4.5. First term is the relative distance error, r_e, between local minimums and third term is the approach velocity(${}^r\widehat{v_{obs}}$) forward to the mobile robot. Second term is the direction error, d_e, between the robot's current direction(d_r) and target direction(d_t) and this term plays a role that a mobile robot can move forward to the target direction. Second and third terms play a role of collision avoidance. Here, the relative distance error, r_e, has to be appropriately limited for effective motion of robot due to the environment condition. Following equation (15) is the simple calculation method of linear velocity according to the robot's angular velocity.

$$v_r = v_{\max} \pm \frac{v_{\max}}{w_{\max}} \cdot w_r, \quad \begin{pmatrix} +: w_r \leq 0 \\ -: w_r > 0 \end{pmatrix} \tag{15}$$

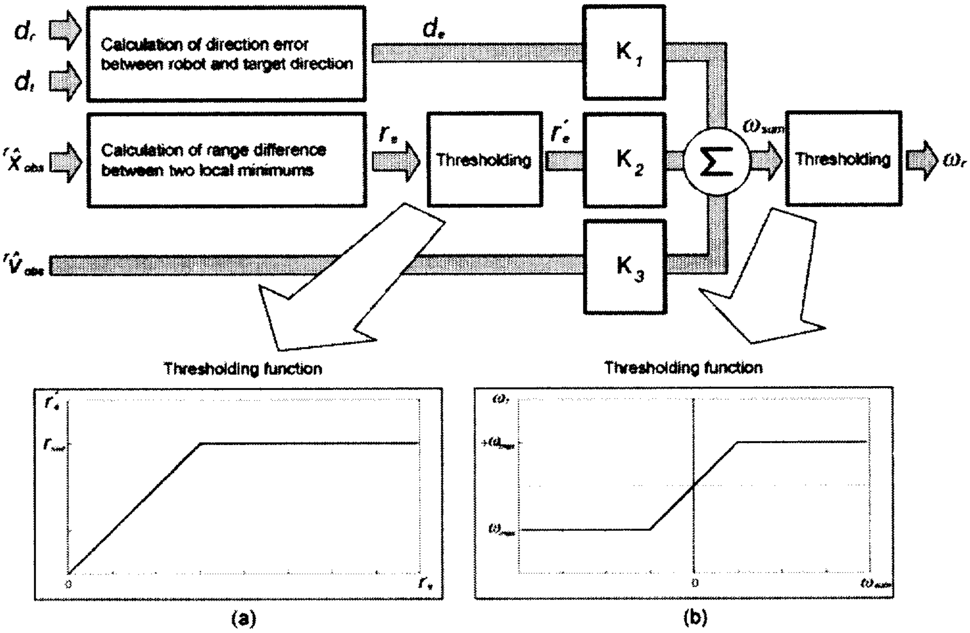


Fig. 4.5 Calculation of angular velocity for collision avoidance

3. Calculation of Movable Direction using FIS

When a mobile robot faces to three problems(corner way, cross way and dead end) in the navigation, it has to estimate the movable direction and move into the desired direction. In this paper, we utilize the Fuzzy Inference System for solving three problems such as figure 4.6. Fuzzy inputs are the sensed values(r_1~r_6) and outputs are the movable direction(d_m) of the mobile robot. And r_{safe} and r_{env} are respectively limited according to the distributed degree of obstacles and the maximum sensed distance to the obstacles in the unknown environment. Through the normalizing of the sensed values, it is transformed into the state values(s) and it is linearly interpolated again. We utilize this interpolated state with the *antecedence*, A. The *consequence*, *B*, is a *triangular membership function* and the number of that is variable according to the number(n) of states of the sensed obstacles. Finally, a fuzzy output is inferred by the *max-min composition* of *mamdani* and a defuzzification method is applied with COA(Center Of Area). Figure 4.7 (a) is antecedence and consequence at corner way and figure 4.7 (b) is those of cross way.

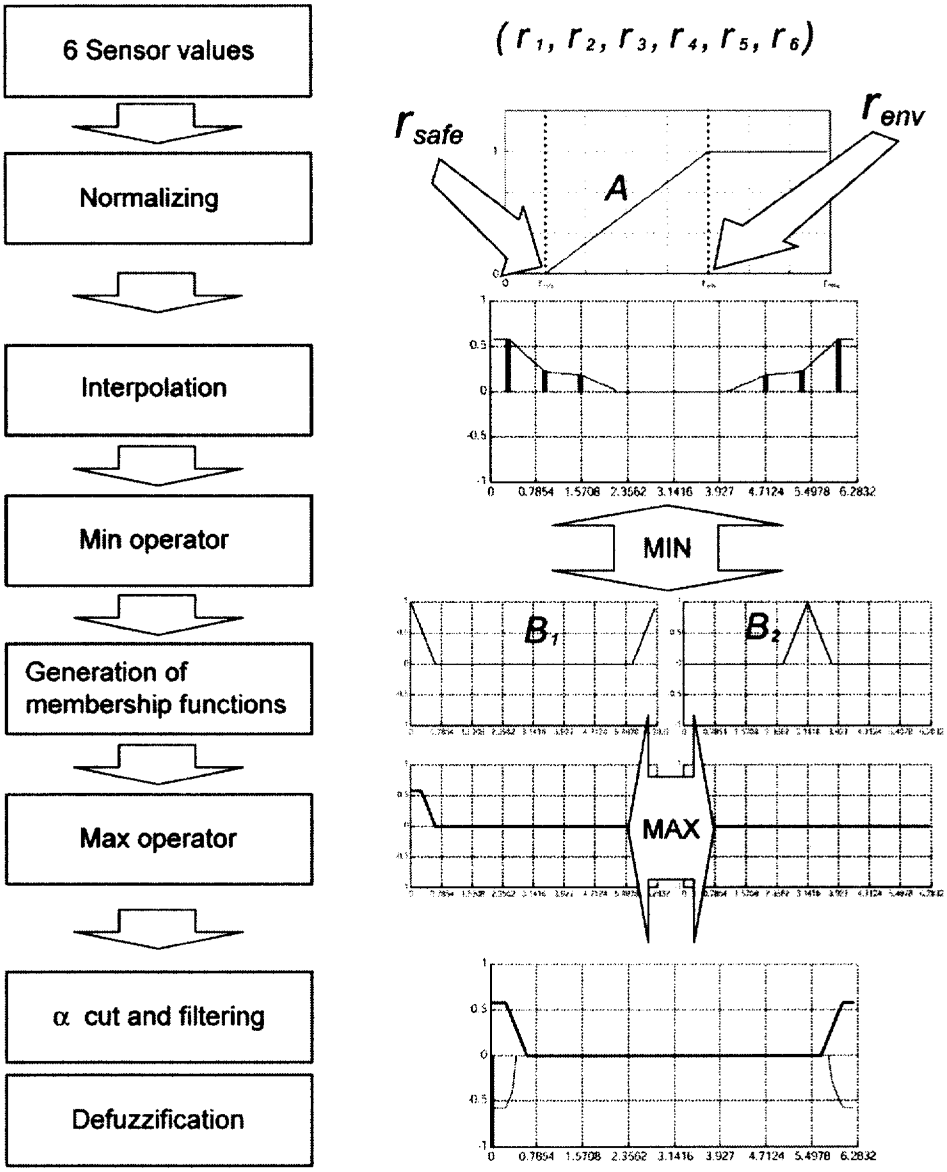
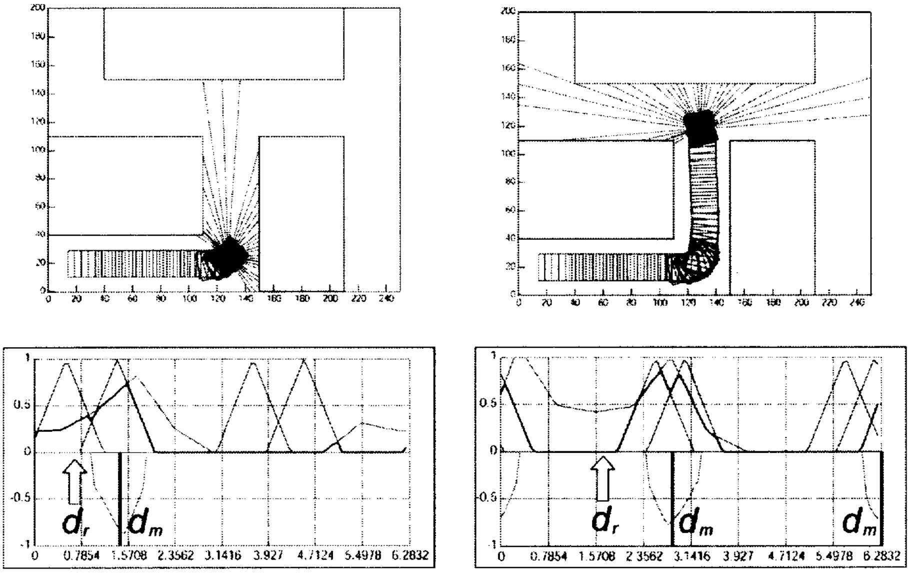


Fig. 4.6 Estimation of movable directions using Fuzzy Inference System



(a) corner way

(b) cross way

Fig. 4.7 Estimation of movable directions at corner and cross ways

4. Turn Side Selector using Binary Decision Tree

Turn Side Selector enables the mobile robot to select the desired direction respect with the mobile robot's current position and direction. When it selects the desired direction, we consider that there are two situations. In case a mobile robot pass through the passage assigned with an edge, it selects the safe direction using the sensor's values with the fuzzy input. A mobile robot must generally select the one of two directions(right and left). During the exploration process, the mobile robot explores the unknown environment in regular sequence. After finishing the exploration process, the mobile robot moves to the via-points on the optimal path.

The mobile robot's desired direction is selected by six range values. Figure 4.8 shows the membership functions of fuzzy inference system for the *Turn side selector* in accordance with the obstacle states, which determine the number of membership function. The universe of discourse of the antecedents is the range value and the universe of discourse of the consequence is the angle for the radian unit. The turn side selector plays two roles. First is to decide that the current position is node(if d_{diff} is more than d_{th}) or edge(if d_{diff} is less than d_{th}) according to the difference, d_{diff}, between the movable direction(d_m) and target direction(d_t) as shown in figure 4.8. Second is to select the moving direction of robot according to the moving mode(exploration and navigation mode). As shown in figure 4.9, we can consider the seven cases at the *Binary decision tree*.

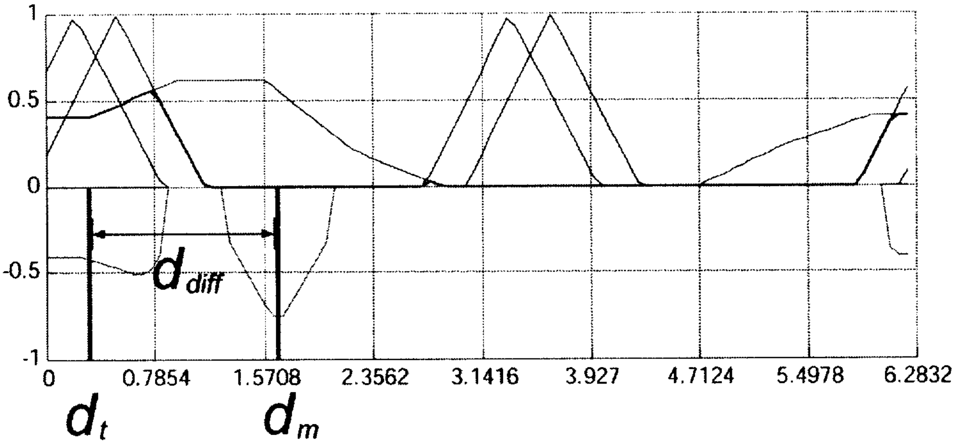


Fig. 4.8 How to determine a node using target and movable directions

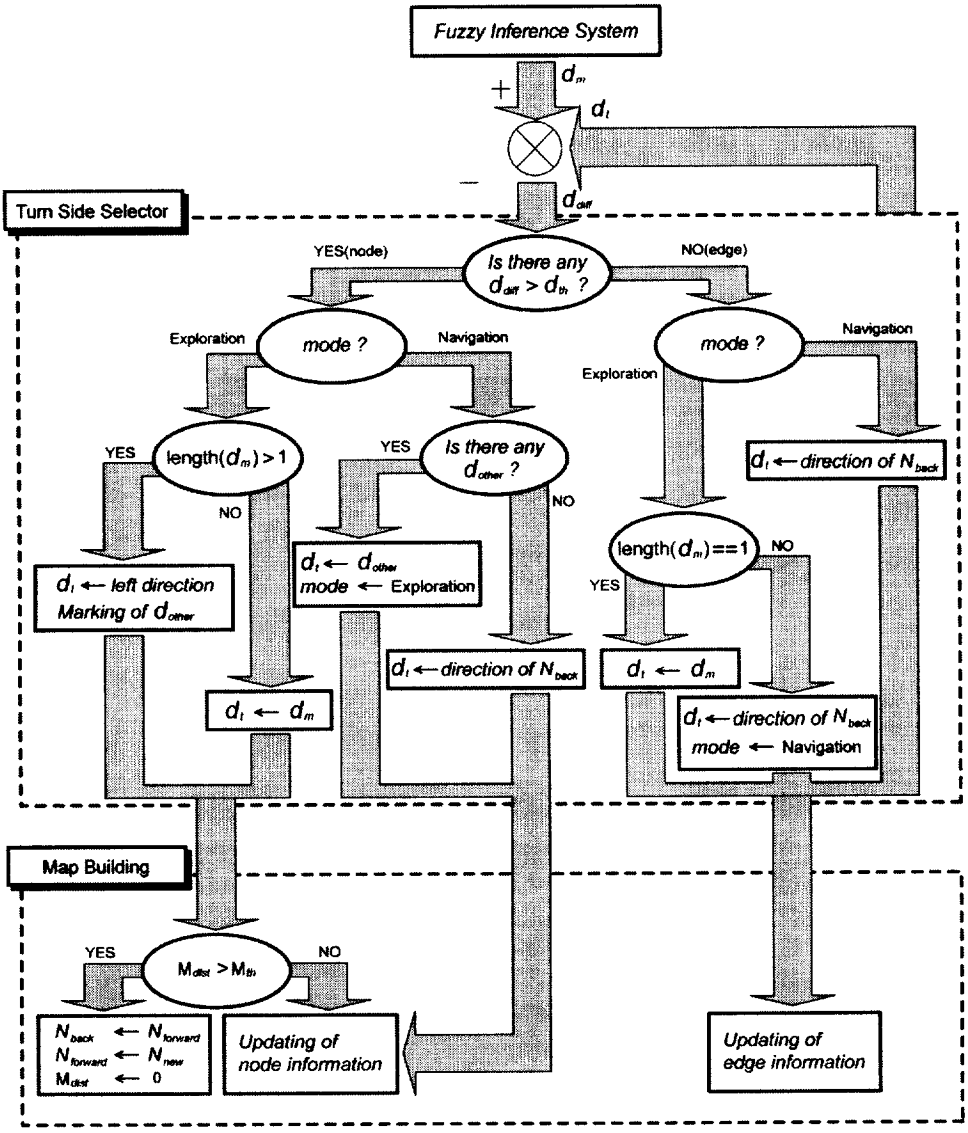

Fuzzy Inference System
+
d_m
d_t
−
d_{diff}
Turn Side Selector
YES(node)
Is there any $d_{diff} > d_{th}$?
NO(edge)
Exploration
mode ?
Navigation
mode ?
Navigation
Exploration
YES
length(d_m) > 1
NO
YES
Is there any d_{other} ?
NO
$d_t \leftarrow$ direction of N_{back}
$d_t \leftarrow$ left direction
Marking of d_{other}
$d_t \leftarrow d_{other}$
mode ← Exploration
length(d_m)==1
NO
YES
$d_t \leftarrow$ direction of N_{back}
$d_t \leftarrow d_m$
$d_t \leftarrow d_m$
$d_t \leftarrow$ direction of N_{back}
mode ← Navigation
Map Building
YES
$M_{dist} > M_{th}$
NO
$N_{back} \leftarrow N_{forward}$
$N_{forward} \leftarrow N_{new}$
$M_{dist} \leftarrow 0$
Updating of node information
Updating of edge information

Fig. 4.9 Turn Side Selector and map building using binary decision tree

5. Map Building and Localization

In the explored environment or the non-explored environment, *Turn side selector* allows a mobile robot to select the desired direction. In case of the *Exploration process*, the mobile robot explores the overall unknown environment with marking the explored direction and the non-explored direction. During the *Exploration process*, a mobile robot memorizes the angle and length of the path. This information is used to select the optimal path for arriving at the target point. *Graph* is expressed by the *adjacency matrices method*. After it has finished the *Exploration process* of map, the mobile robot must find the optimal path. To select optimal path, we use the *Minimum cost method*. Once the optimal path is selected, a mobile robot moves from the start position to the target position. To arrive at final target position, a mobile robot must pass through many via-points(nodes). These via-points correspond to local target positions. To arrive at the local target positions, the mobile robot must travel along many edges. The actions along edges are decided by the autonomous property of mobile robot and the direction information of via-point. After a mobile robot traverses all the via-points, it will arrive at the final target position. Map about unknown environment is builded using *Graph method* that is consisted of nodes and edges. A mobile robot selects the moving direction and saves the information about environment. This information is based on the topological map building. The nodes are included with neighbour nodes and their directions and edges show the interconnection between nodes. Using the generated graph(topological map), a mobile robot can accomplish with localization.

5.1 Topological Map Building using Graph theory

Figure 4.10 conceptually shows environmental map modeling using the Graph approach. N_i means a ith node with the feature values of environment. E_{ij} means an edge connecting between a node i and a node j. If there are no edge between two nodes, it shows there exist no way directly connected. As shown in figure 4.10, the nodes and the edges are generated in accordance with the number of the local minimum value among the sensed values. If the number of the local minimum value is less than two, this is an edge and if more than three, this is a node. Consequently, the environment expressed by the *Graph* is consisted of the nodes and the edges. In this paper, after exploring the environment, the optimal path is selected by the *Minimum cost method*(smallest distance) from the start position to the target position.

As shown in figure 4.10, the mobile robot explores the unknown environment. During the exploration, the environment information is memorized by *Graph* in accordance with the principle such as the right-bottom rectangle. That is, if the number of the local minimum value is less than two, we assign the edge and if more than three, we assign the node. The edges are the trajectory of a mobile robot. After exploring, we can easily select the optimal path using by the generated Graph.

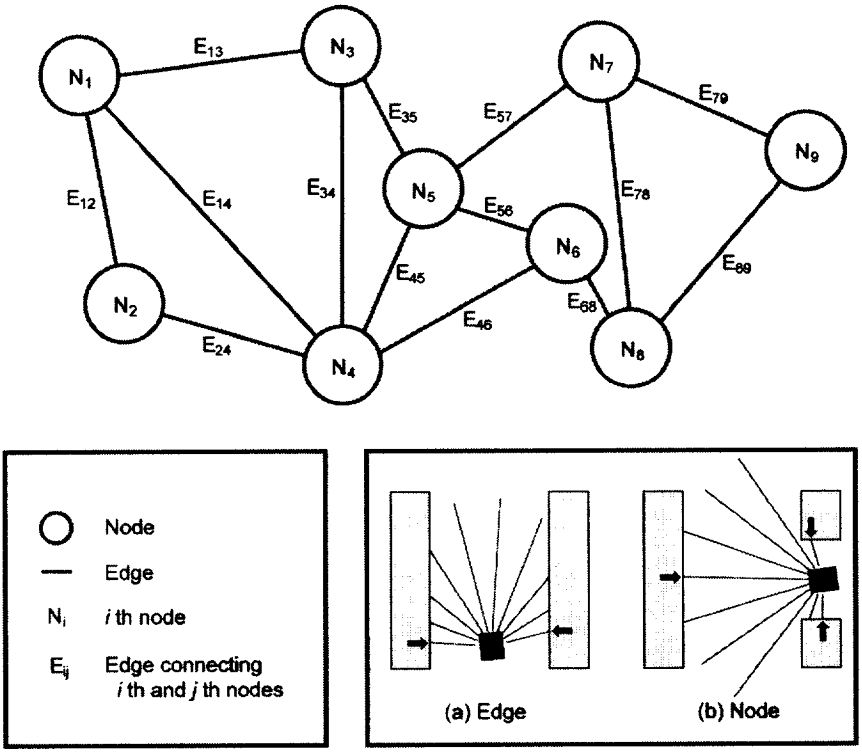


Fig. 4.10 Graph generation method using sensor data

5.2 Localization using Sub-graph Matching Method

To arrive at the target position, the mobile robot must exactly recognize its current position. This is the localization problem. To build the map about the unknown environment map, we have proposed the graph method. Complete graph is consisted of many nodes and edges. These nodes are memorized the information which is the passage direction and distance for arriving at the neighbour nodes. Thus, the mobile robot can estimate its current position using this information and sub-graph matching method about the complete graph. Complete graph expressed the environment map can be divided into many sub-graphs. As shown in figure 4.11, the mobile robot utilizes the *Sub-graph matching method* to estimate its current position at the overall environment map.

For examples, let's localize when the mobile robot's current position is V_5 such as figure 4.11. Firstly, we select the candidates with the information similar to the acquired information at its current position(step 1). If more than two candidates, we select the candidates by comparing its neighbour nodes again(step 2). Consequently, the mobile robot can estimate its current position through the repetition of these processes(step 3~n). Even though the mobile robot estimate the wrong position, it can find the right position when arrive at the next node. In that case, the mobile robot can modify the path through the reestimation.

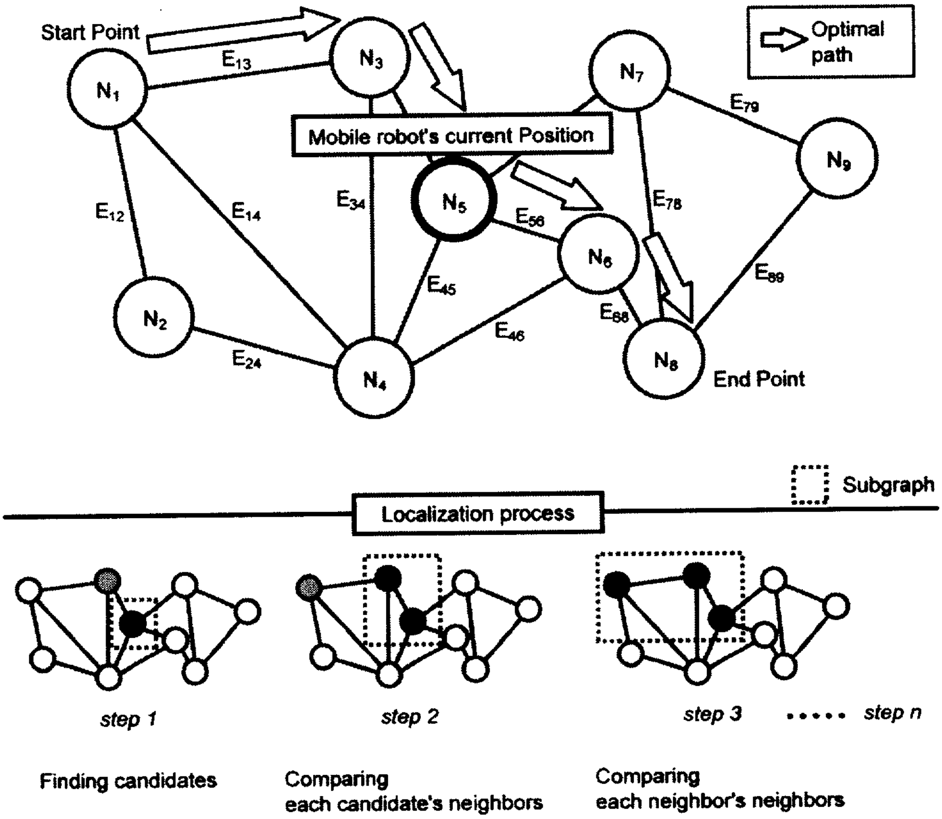


Fig. 4.11 Localization method using Sub-graph Matching Method

V. Web Monitoring System

1. Internet Protocol

The protocol means the standard that explain how to pass on the data, how to check up and correct the errors, how to send the respondence when the data is transferred among the computers. The first network about the packet data communication is ARPANET that built at the *Department of Defense* of United States of America in 1968. The most important problem to actually link with other computers using ARPANET is that there is not the standard transport layer protocol to link the terminal hosts via some sub-network. In order to solve this problem, the *Department of Defense* establishes the TCP/IP protocol. The IP and TCP protocols are essential in TCP/IP protocol. The TCP/IP protocol is consisted of not only TCP and IP but also UDP(User Datagram Protocol), ICMP(Internet Control Message Protocol), ARP(Address Resolution Protocol), RARP(Reverse ARP). After in 1993, WWW(Word Wide Web) service based on the TCP/IP explosively spread, therefore the TCP/IP was substantial standard in the communication network of computers. The TCP/IP is consisted of four layers such as Network access , Internet, transport and application as shown in figure 5.1.

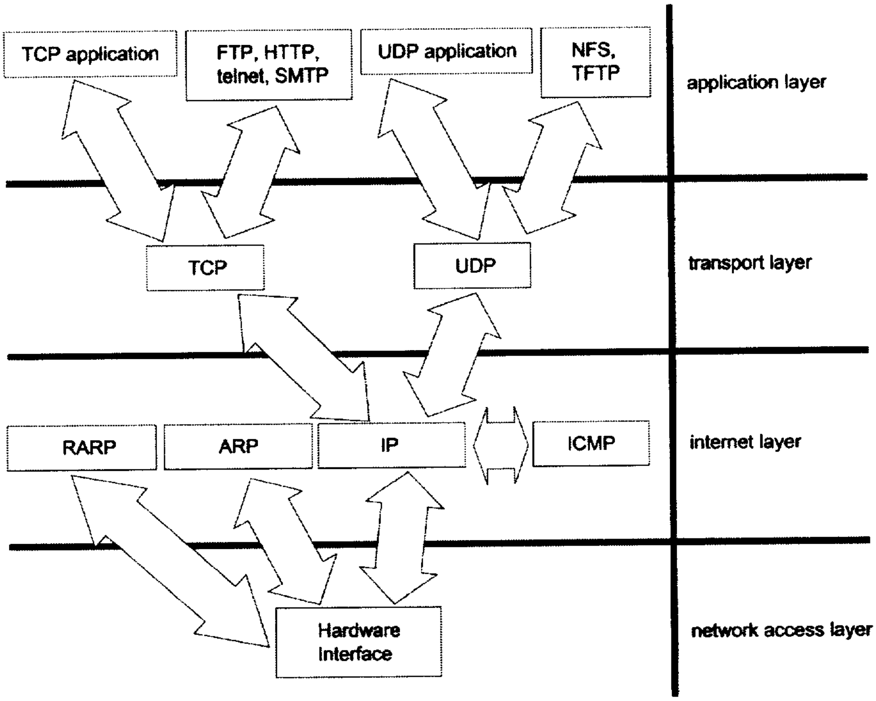


Fig. 5.1 Four conceptive layers of TCP/IP software

2. Ubiquitous Computing environment

The word "ubiquitous" comes from a Latin word meaning "omnipresence." According to the Ministry of Public Management, Home Affairs, Posts and Telecommunications, the market size of the ubiquitous networking industry is projected to reach about 3 billion yen by 2005 and 8 billion by 2010. As this shows, the market of the ubiquitous networking industry is growing faster than most other IT markets. In a ubiquitously-networked society, not only mobile phones, PCs, PDAs and other communications terminals, but also intelligent appliances, home robots, and many other home electronics will be linked together, resulting in an extensive communications environment entirely different from what we know today. So the mobile robot is based on the ubiquitous system and WPAN(Wireless Personal Area Network) to get the robot's information regardless of any time or any place.

3. Definition of Internet Control

The manipulators used on the factory productive automation mean the robots that move heavy material or accomplish with simple repetition tasks in stead of human being. Existing fixed idea controlling the robot at the same space is disappeared due to the appearance of the remote control of the robotics. This remote control technique is gradually generalized and the one of the part necessary with the remote control method is the personal robot associated with the home network. The mass-media that the possibility of generalization is high and it is possible to transfer the variety of information is internet. It is connectable everywhere and inexpensive. Internet program and GUI(Graphic User Interface) can be standardized through web browser. So all the implementations in this paper are based on the internet control that transferred or received the necessary information through TCP/IP.

4. Latency

Internet latency is defined according to the number of via-node and its load. Physical distance connecting between two areas have directly no relation. However, the latency seems to be generally increased according to the distance because the far distance has generally many via-nodes. Latency is less in the night because there are little internet users and latency is more in the day because there are more internet users. However, it is impossible to express the latency according to a period of time. That is, latency is impossible to statistically express according to a time period or a day of the week. This latency is the most important thing when we control the mobile robot through internet because the transferred command is lost due to the latency as shown in figure 5.2. So to solve this problem, we may use the *command filter* as shown in figure 5.3.

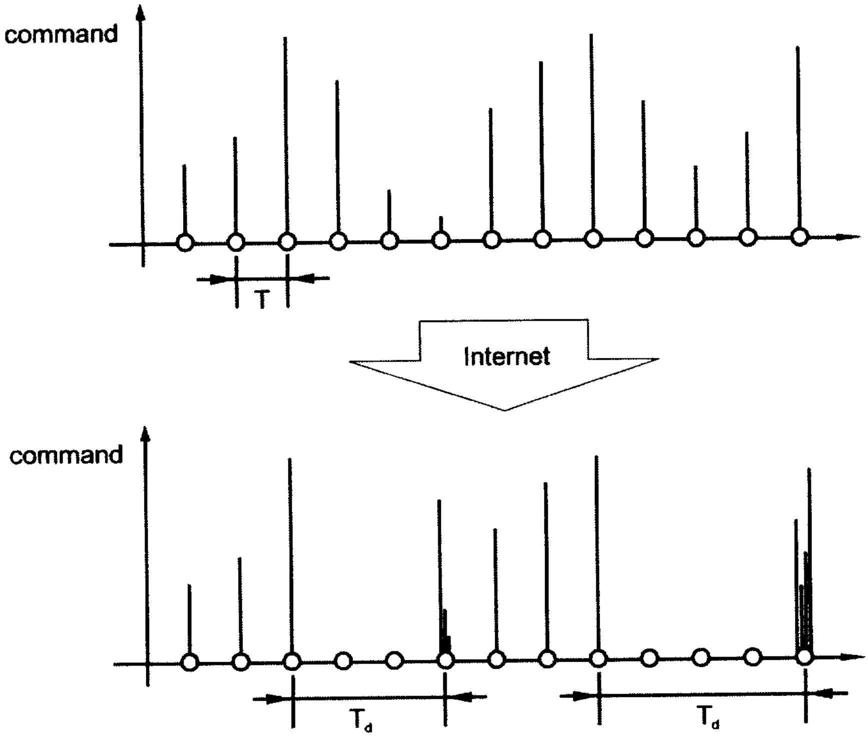


Fig. 5.2 Effect of internet latency

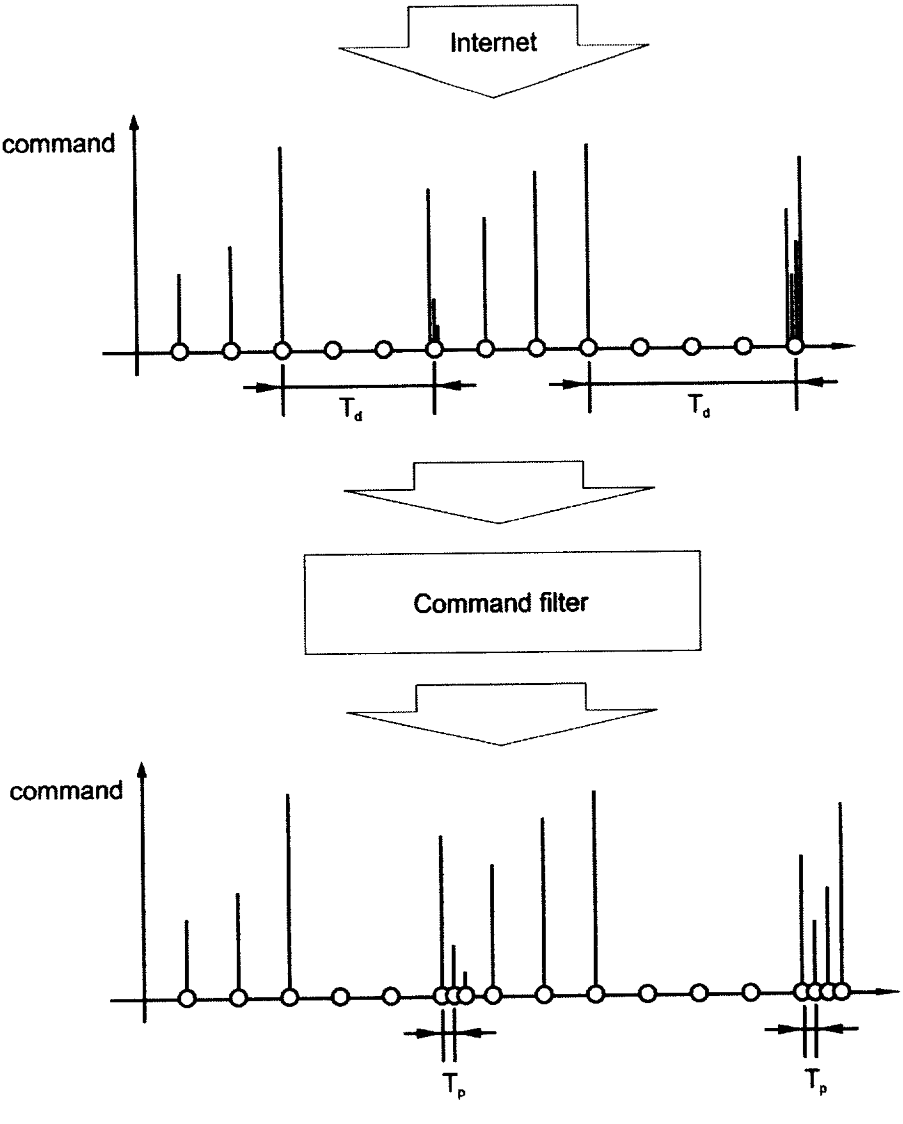


Fig. 5.3 Role of command filter

5. Control Architecture

To control a mobile robot in remote area through the internet, we have to choose the proper control architecture which transfer the commands into a mobile robot. In case of two wheels mobile robots, the low level command is a linear velocity and an angular velocity according to the right wheel and left wheel respectively. To directly control the robot in remote area using a keyboard or joystick, the control architecture must be constructed as architecture transferring the low level commands. Even though the operators can control a robot using the joystick, he/she want to be automatically operated the mobile robot itself if it knows the target position. In this situation, if we use only the pre-mentioned control architecture, it is difficult to get the satisfactory results. In next, we can consider the sequential commands that the robots have to do. If the user can not take enough time to monitor the robot, user will give the entire task to the robot. In this case, a robot needs another control architecture. In short, we can propose three control architectures as follows.

- Direct Internet Control Architecture
- Supervisory Control Architecture
- Job-scheduling Control Architecture

Direct internet control architecture is proper architecture for the low level control and *Supervisory control architecture* is that an user gives only the final position to the robot and monitors the robot's movement in real time. Finally, *Job-scheduling control architecture* is that the sequential commands generated by an user is transferred into the robot and the robots accomplishes with the commands in order. In this paper, we utilize with Job-scheduling control architecture so that a mobile robot can accomplish with commanded tasks in regular sequence.

6. Design of Control Architecture

In this research, because we utilize the internet with the communication method and user can connect to the mobile robot(server) using the arbitrary internet computer. The robot can also communicates with the external clients through wireless LAN and transfers the necessary data into each peripheral unit based on WPAN(Wireless Personal Area Network) constructed using the bluetooth module. Figure 5.4 shows the overall flow of wireless communication system.

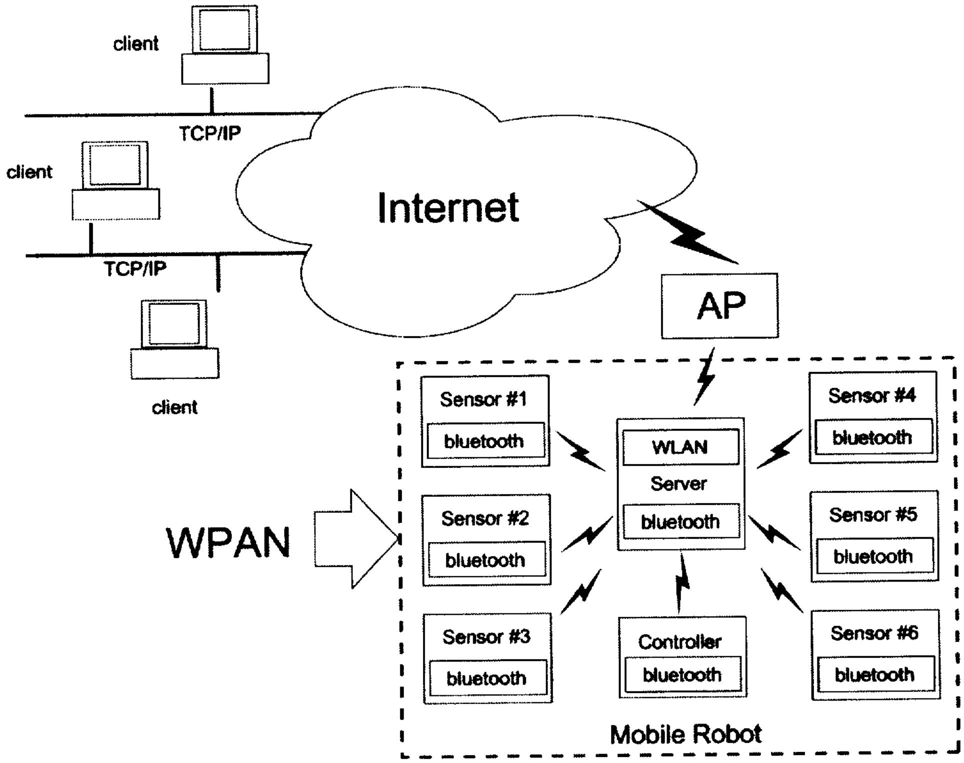


Fig. 5.4 Flow of overall wireless communication interface

Ⅵ. Experiment System

The mobile robot system is consisted of six ultrasonic sensor modules. The ultrasonic sensors are utilized with inexpensive SRF-04 and we measure the range value using Atmega 8 MCU sensor controller. The measured range values are transferred into the server(mobile robot system) using the bluetooth modules. Using the transferred sensor values and the proposed algorithm in this paper, we also transferred the control inputs into the motor controller based on the same wireless communication modules. The stepping motors(SST58D5200) are operated by these control input through the servo driver. The mobile robot also is equipped with the wireless LAN which other clients are enable to connect into the mobile robot through TCP/IP.

1. Modelling of mobile robot body

To model our mobile robot body, we have used *Autodesk Inventor Professional*. *Autodesk Inventor Professional* not only is it a powerful 3D tool, but it's also the best way to connect design teams with manufacturing engineers. Because it can integrate existing 2D designs into 3D design environment, it offers a risk-free path to specialized 3D design.

Followings are the advantages of *Autodesk Inventor Professional.*

○ Compatibility

We can convert 2D drawings to parametric model of Autodesk Inventor and rightly check synthesized motion. We can also easily and quickly transform 3D part drawings into 2D drawings.

○ Productivity

Using Diagnostic Design DoctorTM function or help function, we can easily, quickly utilize and learn Autodesk Inventor.

○ Adaptive technology

Using Adaptive technology, we can model the process of work with adjusting of shape and position through the analysis of function before fixing of shape.

There are many advantages of Autodesk Inventor Professional. Using *Design Virtual Models of Wiring, Cables, and Harnesses With Autodesk Inventor Professional*, the cumbersome process of designing cables and wires is now quick, easy, and painless. Designing rigid tubing, flexible hose, and piping systems is no longer a time-consuming chore. Autodesk Inventor Professional helps you communicate your final design to your manufacturing team, with fully detailed assembly drawings. By importing and automatically generating accurate circuit board information through 'Create Printed Circuit Board Geometry using IDF (Intermediate Data Format) files', you can maximize space use, detect potential part interference, check mounting hole locations, and ensure proper fit without the need for a physical prototype.

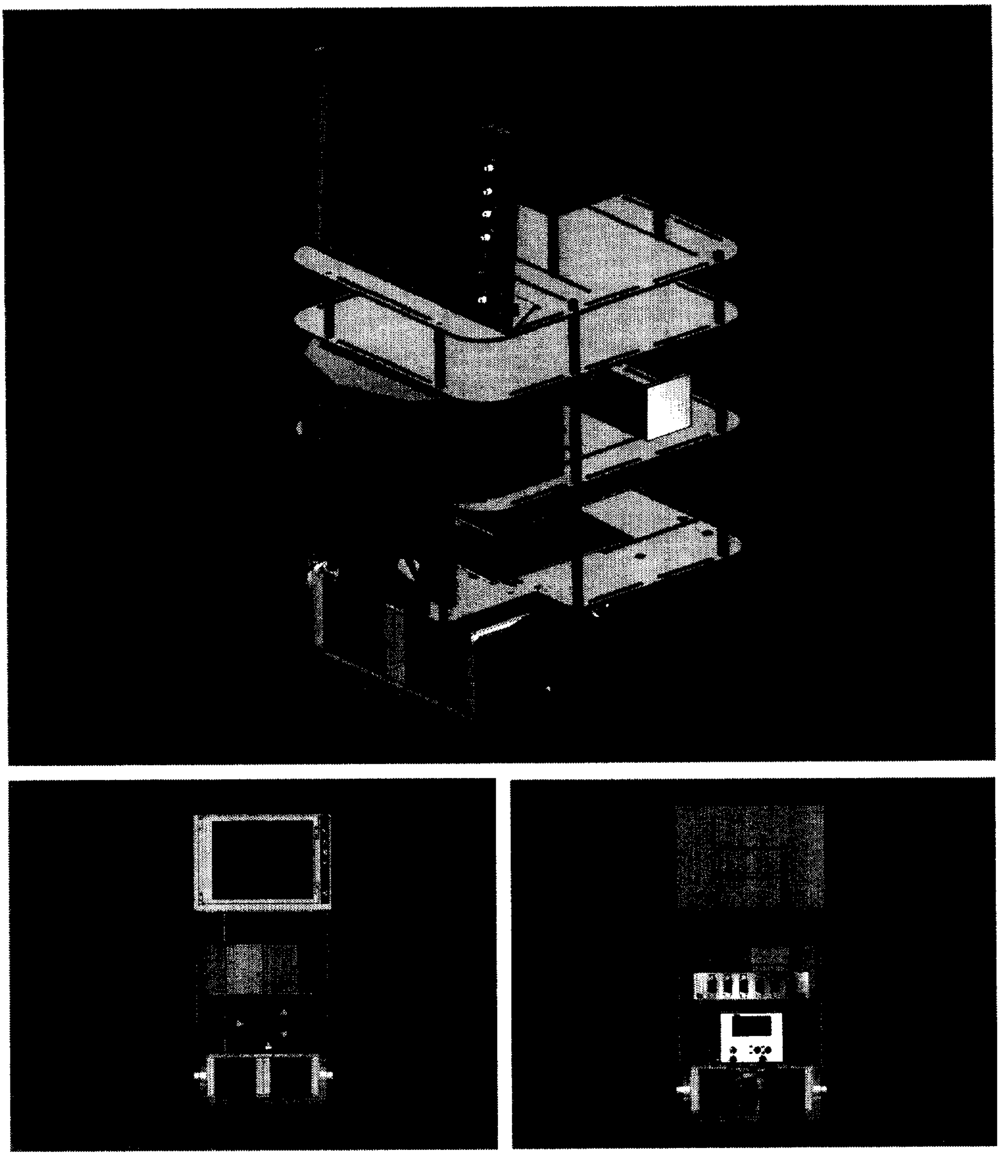


Fig. 6.1 Body frame designed with Autodesk Inventor

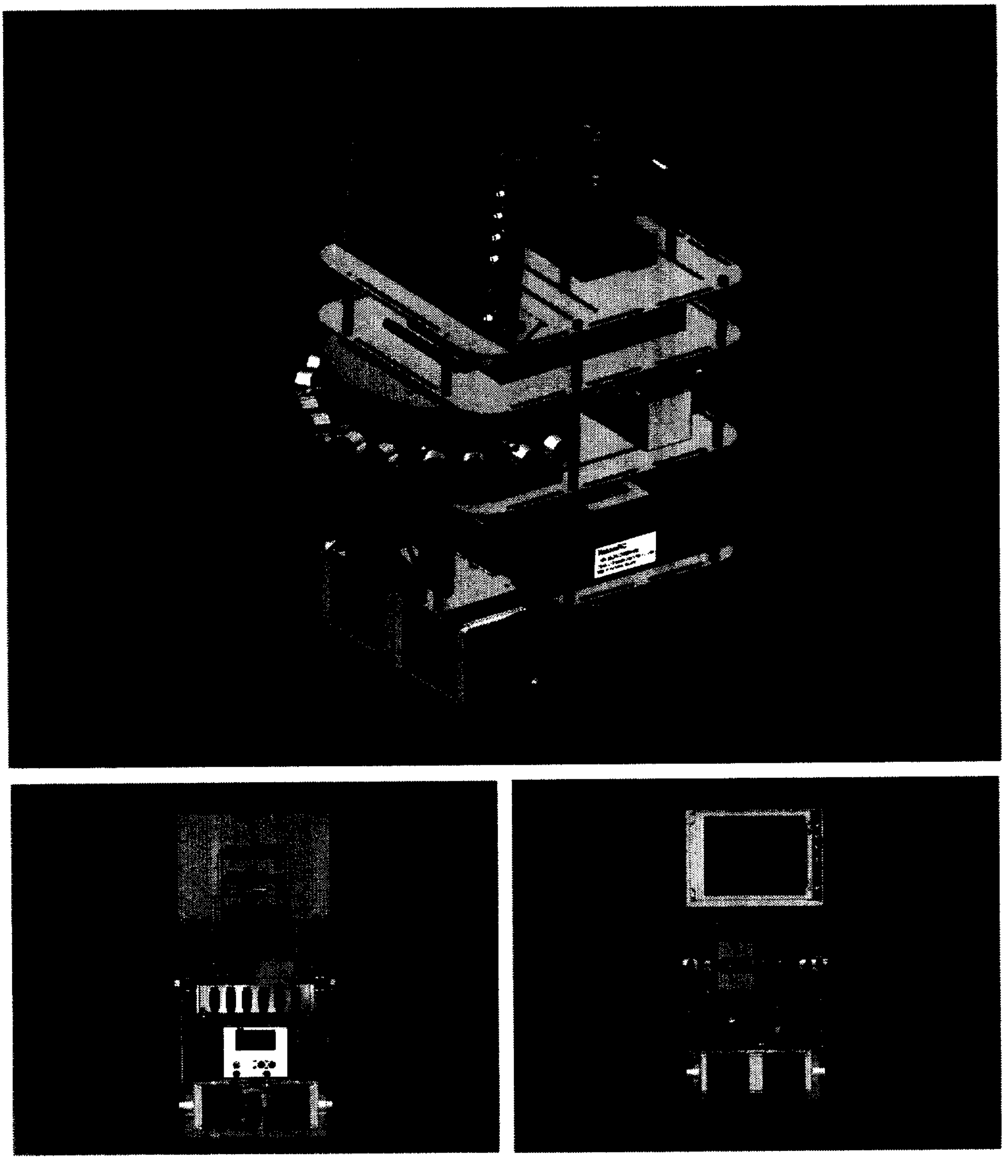


Fig. 6.2 Robot body assembled by Autodesk Inventor

2. Mobile Robot System

When the mobile robots change its orientation, it is categorized several types according to the mobility how to move freely and the steerability how to steer freely. The front two wheels are operated by the stepping motors and the instantaneous center of rotation, which is intersection point of the extend line of wheel rotation axes, can be generated only one using the off-centered orientable wheel of rear wheel. So the mobile robots can freely steer the arbitrarily direction.

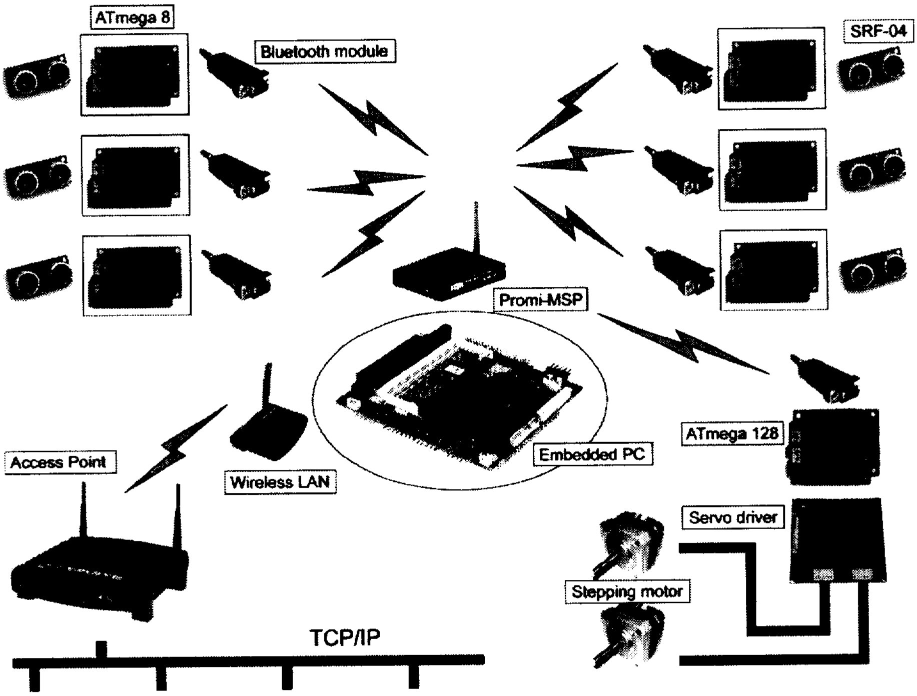


Fig. 6.3 Architecture of mobile robot system

3. Graphic User Interface

Figure 6.4 is GUI(Graphic User Interface) that an operator can monitor the movements and states of the mobile robot. In figure 6.4, index 'A' shows the real-time displayed sensing values and index 'B' is the button that can operate the stepping motors and the view of the mobile robot's velocities. And index 'C' is real-timely shown the states of robot(position, direction, target direction and moving distance of mobile robot). 'D' means the mobile robot and 'E' is the connected client IPs when client connects into the mobile robot using TCP/IP. This GUI is coded by C# language to apply .NET environment. Figure 6.5 is the mobile robot *WEBO*.

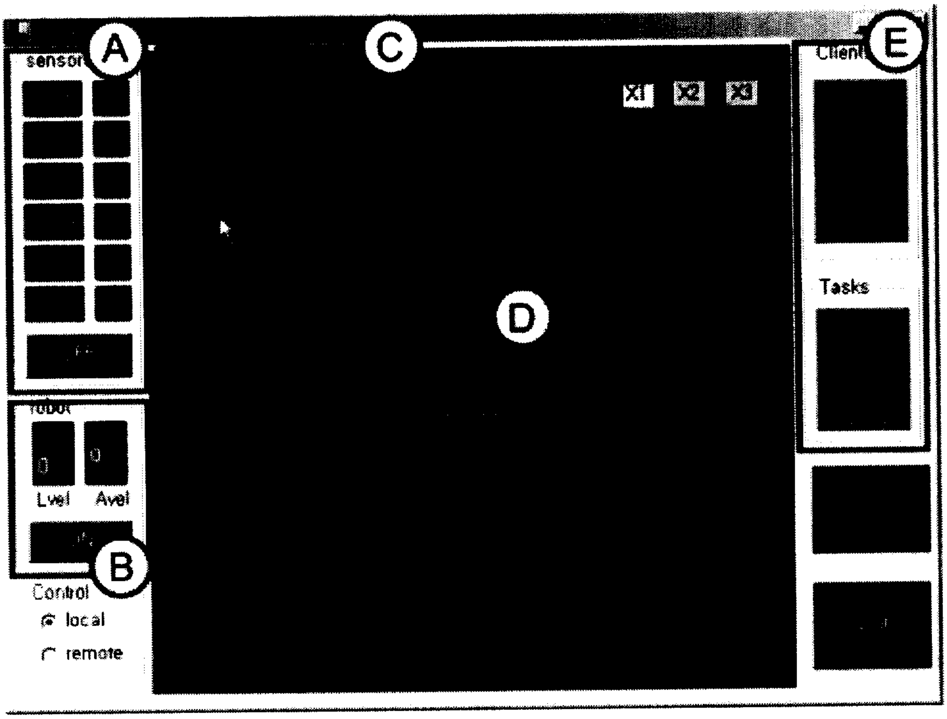


Fig. 6.4 Graphic User Interface for monitoring

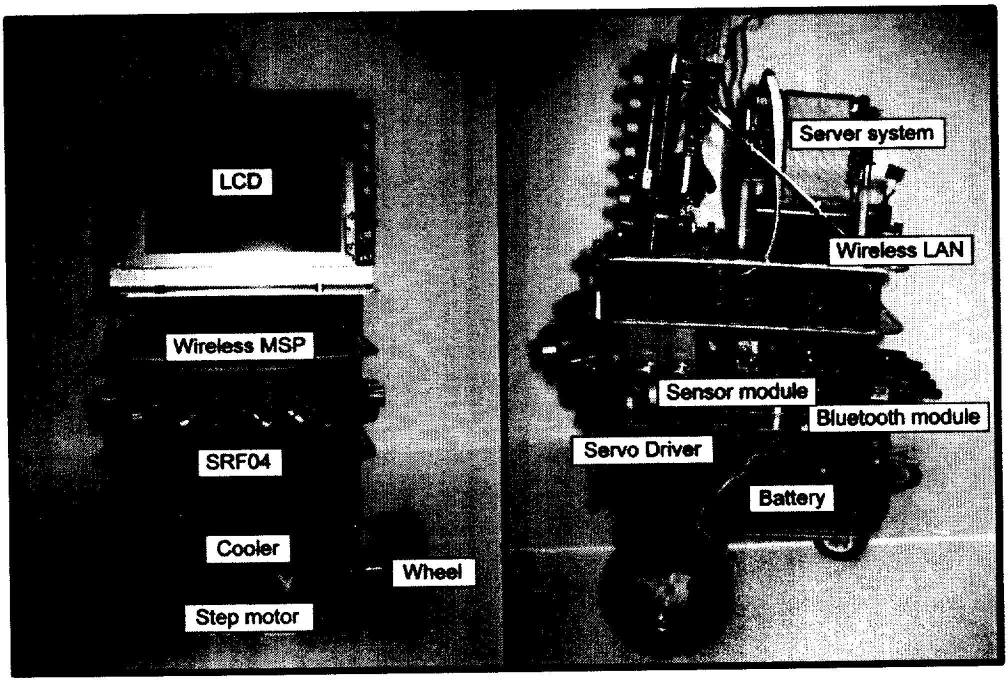


Fig. 6.5 Mobile robot system 'WEBO'

Ⅶ. Simulation and implementation

In this paper, the simulator is coded by *Matlab* language. Above part at figure 7.1 is the unknown environment that the mobile robot should avoid the obstacles and build map. In unknown environment map, small rectangulars mean the distributed obstacles. The overall map is limited into width 4500 *mm*, height 2000*mm*, respectively. The rectangular of left bottom in the unknown environment means the mobile robot equipped with the six ultrasonic sensors at its front. Graph at left bottom at figure is real-timely shown the result of FIS(Fuzzy Inference System), which it enable the mobile robot to avoid the obstacles and move into the target position. The mobile robot used at simulation is the same as the experiment device. Through comparing the simulation results with the experiment results, we will verify the validation of the proposed algorithm

In this paper, we have accomplished with three simulations and implementations. First is to find the states of the obstacles using the proposed algorithm. Second is to build the map about unknown environment using the states of the obstacles. So, when it is given the start position and the final position, final is to navigate into the target position based on the builded map.

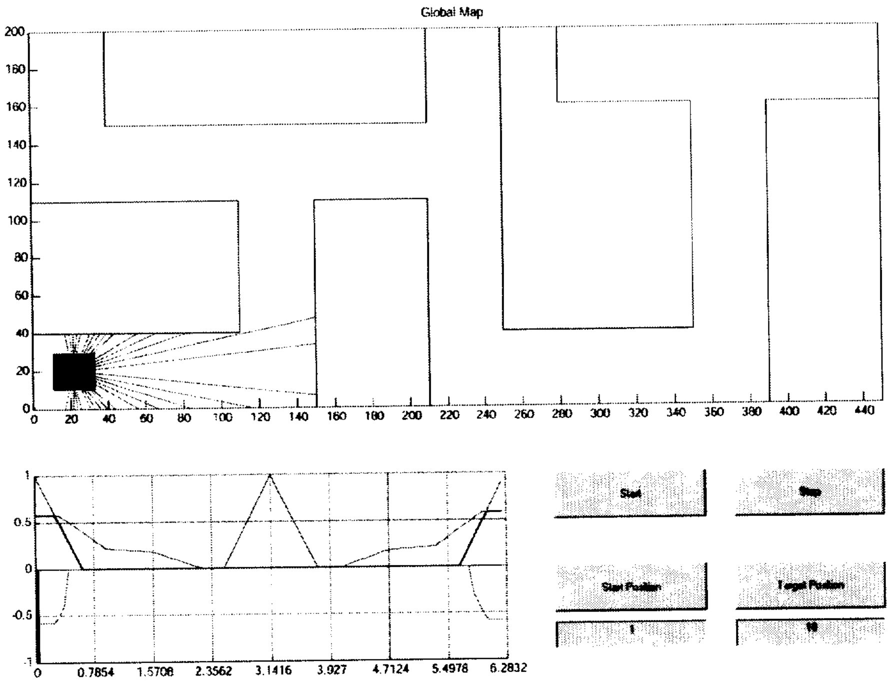


Fig. 7.1 Simulator coded by Matlab

1. Calculation of movable direction

We have estimated movable direction(d_m) about two cases. Figure 7.2 and 7.3 are shown the simulation results in case corner way and cross way that are consisted of from 0 to $\pi/2$ and from $\pi/2$ to π direction, respectively. Figure 7.4 is implementation result and figure 7.5 and 7.6 are shown those in corner way and cross way that are consisted of from 0 to $\pi/2$ and from $\pi/2$ to π direction. As shown in result graphs, each figure is shown directions similar to real direction of passage. In figure 7.3 and 7.6, we can see other direction after a mobile robot has passed by the cross way. Through this point, we can know there is other direction. Therefore, when a mobile robot is faced with the dead end, it has to come back this cross way and explore again.

To validate the proposed algorithm, we assume the simple obstacles such like the walls. The mobile robot navigate into the exploration mode at the start position. When the feature values enabled to set the node are sensed, the mobile robot memorizes the information such like the moving distance, the moving direction and the neighbor nodes of current node. To analysis how exact the mobile robot recognize the states of obstacles, it transfers the experiment data(target direction) into the connected client. The target direction is the states of obstacles in other word. The implementation results included three graphs show the states of obstacles at passage way, corner way and cross way, respectively.

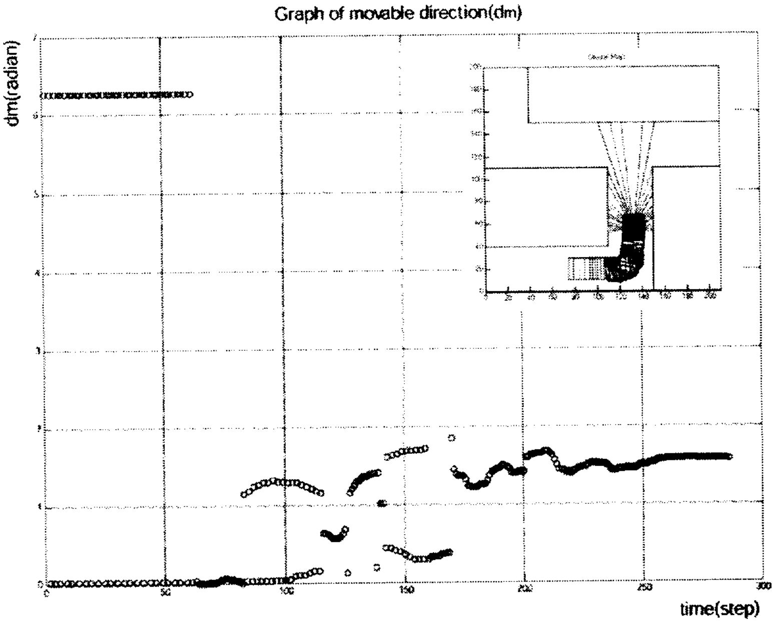


Fig. 7.2 Simulation result of d_m at corner way

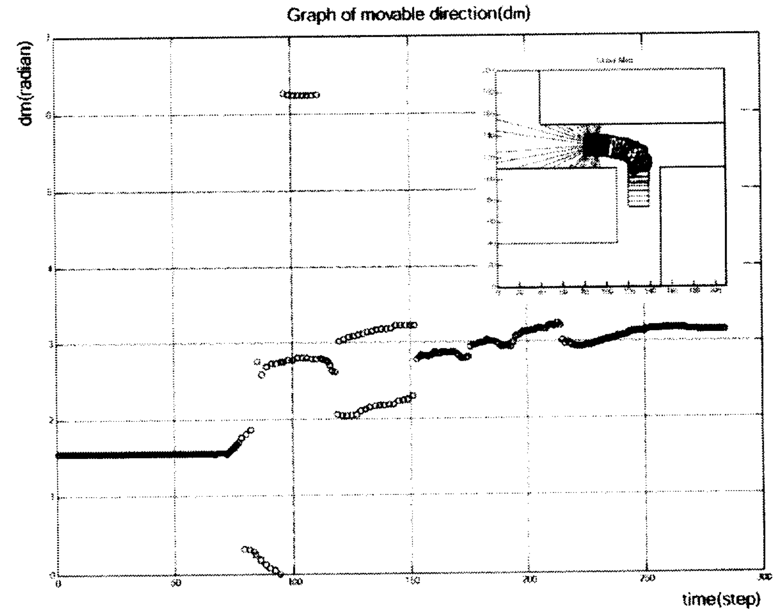


Fig. 7.3 Simulation result of d_m at cross way

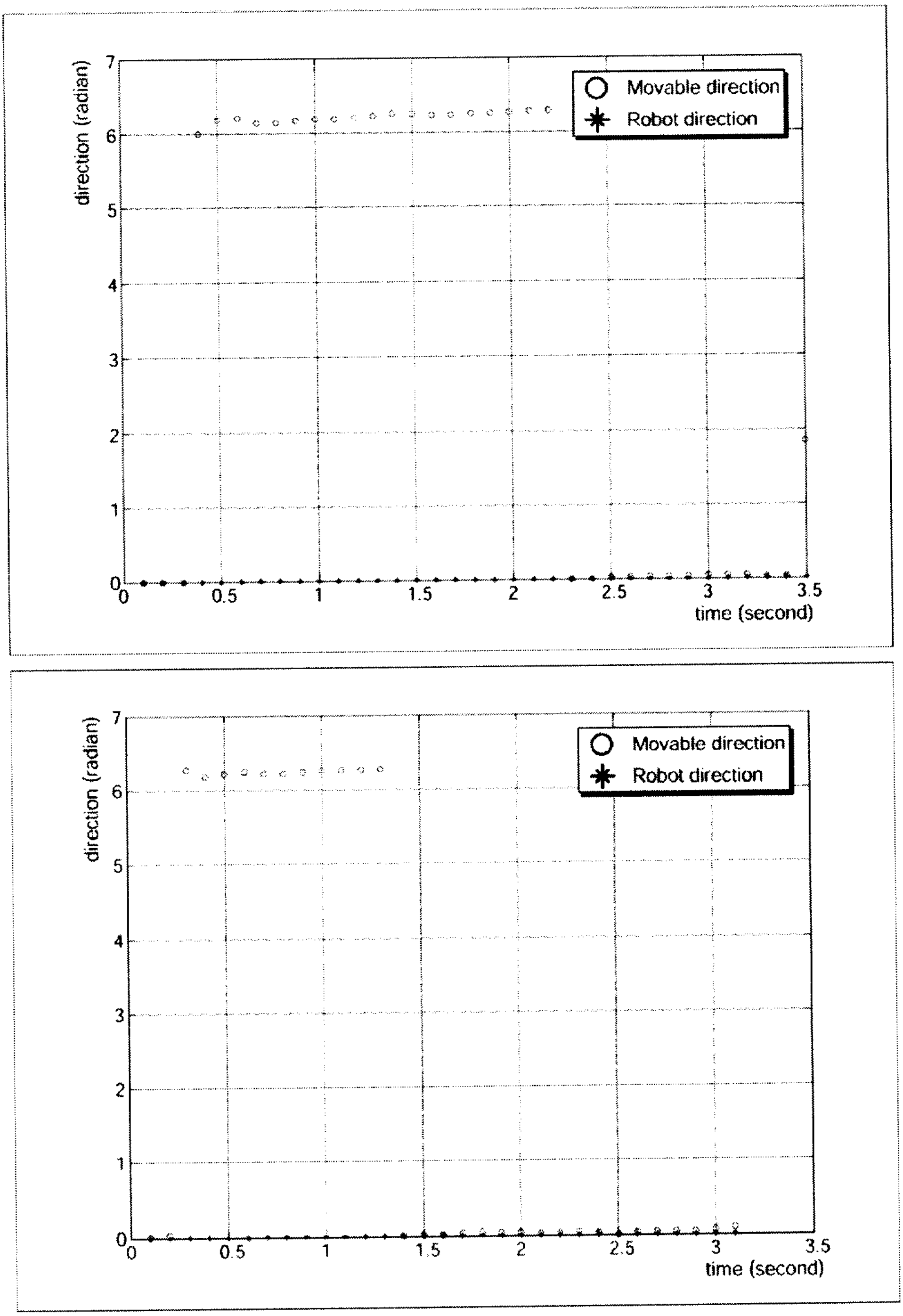


Fig. 7.4 Implementation results of d_m at passage way

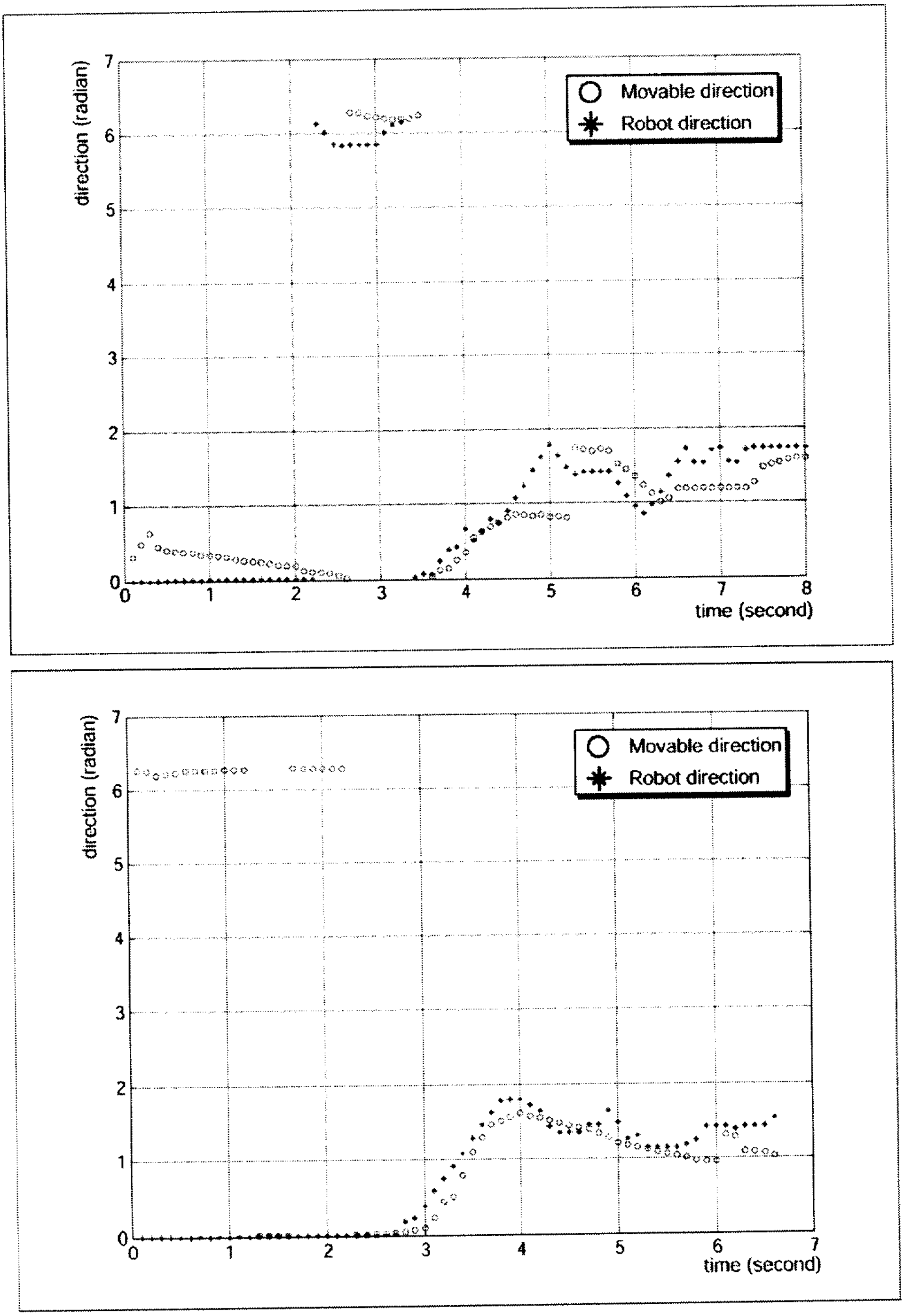
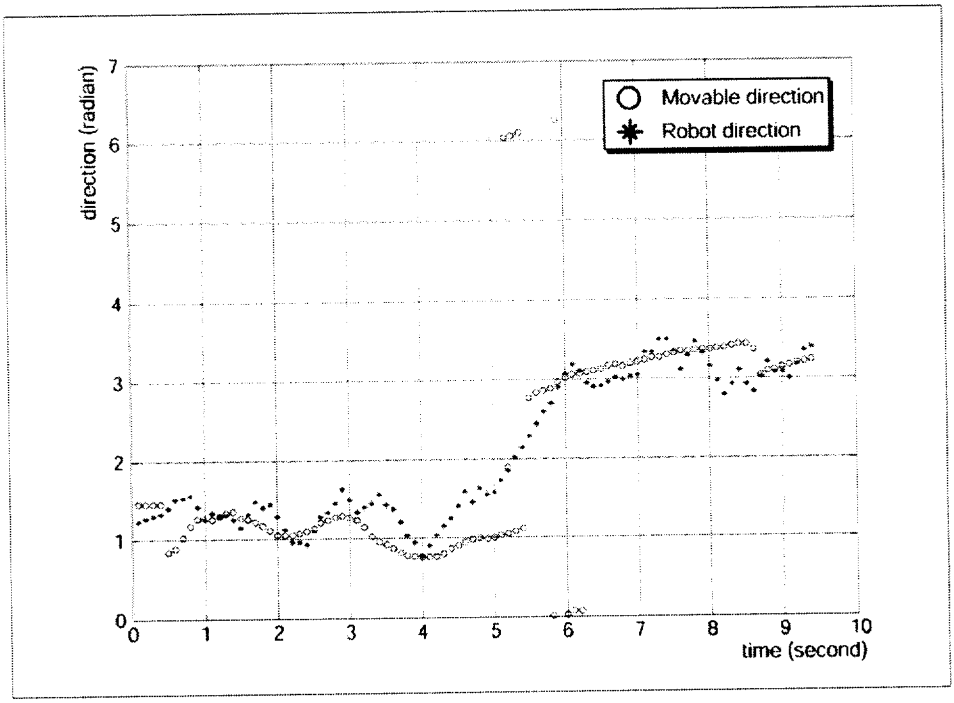


Fig. 7.5 Implementation results of d_m at corner way



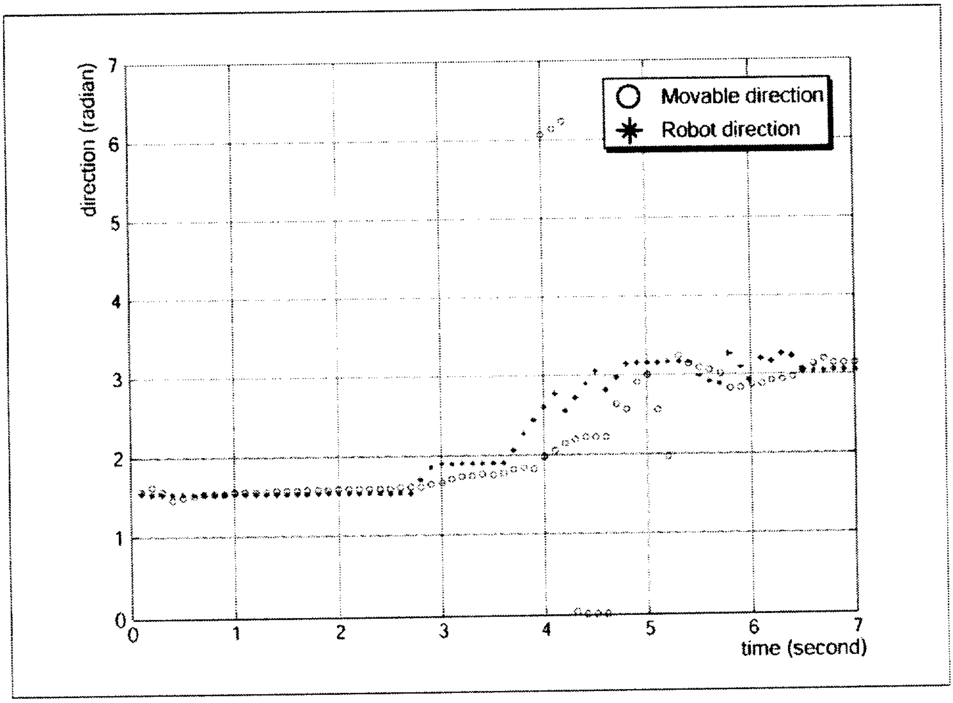


Fig. 7.6 Implementation results of d_m at cross way

2. Topological map building

Simulation and implementation are accomplished with the same size obstacles and environment. The map builded by method verified at simulation and implementation results of 'Calculation of movable direction' is shown in figure 7.7 and 7.8. The numbers included figure 7.7 and 7.8 are the generated nodes. Initially, a mobile robot begin with the exploration mode and it move with the navigation mode in the path such as 5-4-3. As shown in figure 7.7 and 7.8, the start position(node 1) of the navigation is (20, 20) and explore from node 1 to node 12. With saving the visited nodes to stack, a mobile robot can come back the start position or unexplored position after complete exploration or in case the dead end, respectively. After complete exploration, the unknown environment is consisted of 12 nodes that included the neighbour nodes, directions and distances to those. Figure 7.9 and 7.10 are to display the memorized information using *Graph method* after exploration process and table 1 and 2 are complete node information.

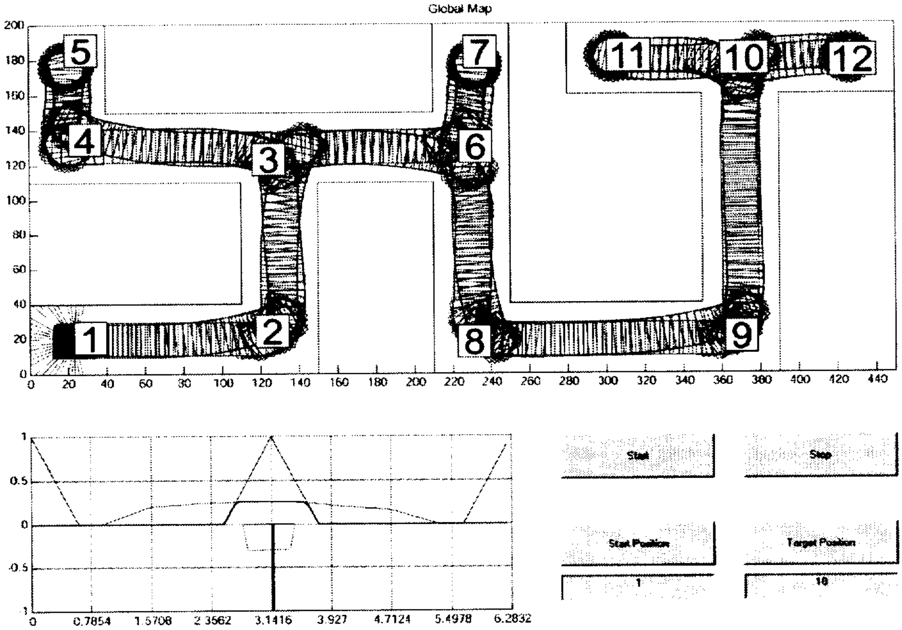


Fig. 7.7 Simulation results of map building

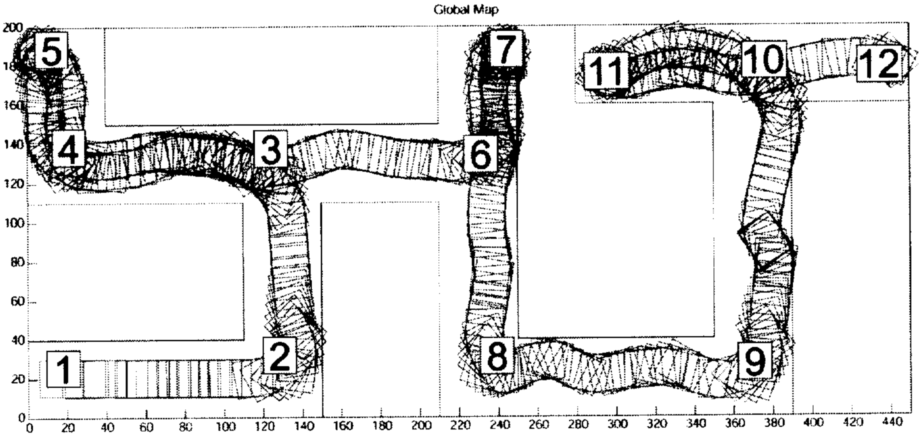


Fig. 7.8 Implementation results of map building

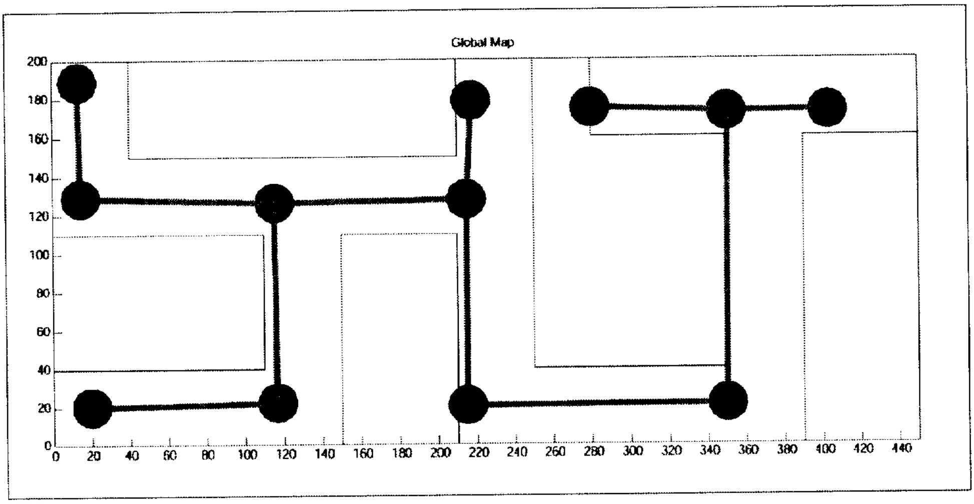


Fig. 7.9 Topological representation of unknown environment(simulation)

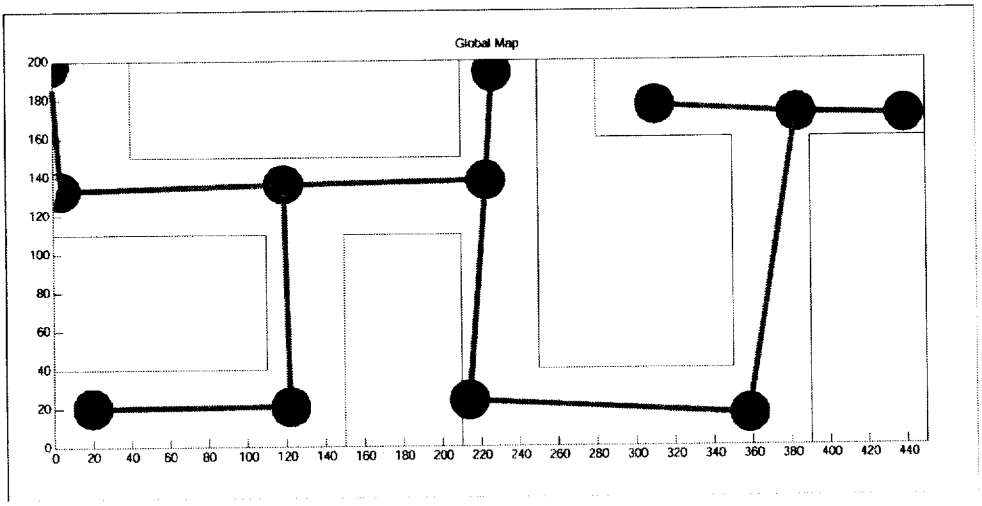


Fig. 7.10 Topological representation of unknown environment(implementaion)

Table. 1 Simulation results for node information

N	N_{nbr}	d	M_{dist}
1	2	0.0199	96.9983
2	1	3.1615	96.9983
	3	1.5814	104.2536
3	2	4.7229	104.2536
	4	3.1162	101.0130
	6	0.0154	99.2006
4	3	6.2578	101.0130
	5	1.5945	59.8667
5	4	4.7361	59.8667
6	3	3.1570	99.2006
	7	1.5203	51.2484
	8	4.7133	107.3223
7	6	4.6619	51.2484
8	6	1.5717	107.3223
	9	0.0054	135.2632
9	8	3.1470	135.2632
	10	1.5703	151.7058
10	9	4.7119	151.7058
	11	3.1141	70.8920
	12	0.0044	52.7674
11	10	6.2557	70.8920
12	10	3.1460	52.7674

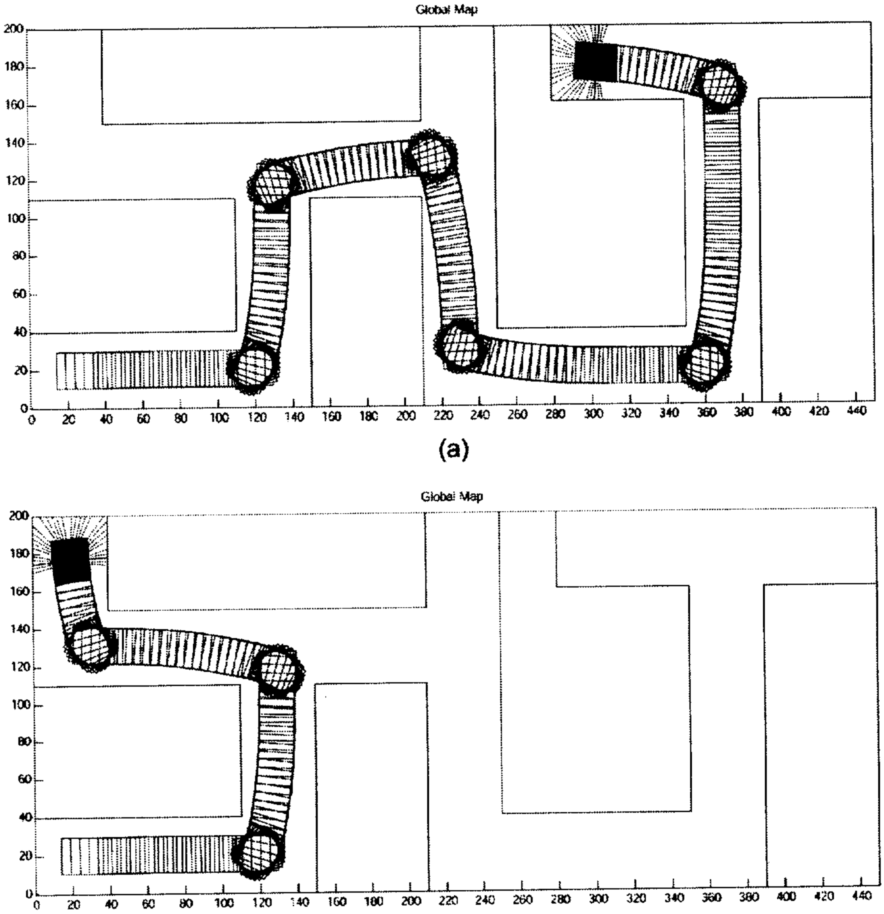
Table. 2 Implementation results for node information

N	N_{nbr}	d	M_{dist}
1	2	0.0049	102.4521
2	1	3.1465	102.4521
	3	1.5995	115.5316
3	2	4.7411	115.5316
	4	3.1707	115.0013
	6	0.0155	104.1010
4	3	0.0291	115.0013
	5	1.6512	65.0167
5	4	4.7928	65.0167
6	3	3.1571	104.1010
	7	1.5101	56.1824
	8	4.6331	114.0951
7	6	4.6517	56.1824
8	6	1.4915	114.0951
	9	6.2301	144.5112
9	8	3.0885	144.5112
	10	1.4120	157.8571
10	9	4.5536	157.8571
	11	3.0813	73.0701
	12	6.2714	55.8902
11	10	6.2229	73.0701
12	10	3.1298	55.8902

As shown in figure 7.9 and 7.10, graph shows the map displayed the links among the nodes based on table 1 and 2. There are some errors among the measured node position respect with the simulation environment. This is different on how to measure the distance between the nodes. However, these errors are not important on that the mobile robot recognizes the unknown environment. That is, as human beings do not move based on the exact numerical data, so does the mobile robot. The mobile robot localizes its current position based on the feature values of each node and select the direction using that. So these errors do not play a role of misdirection.

3. Navigation

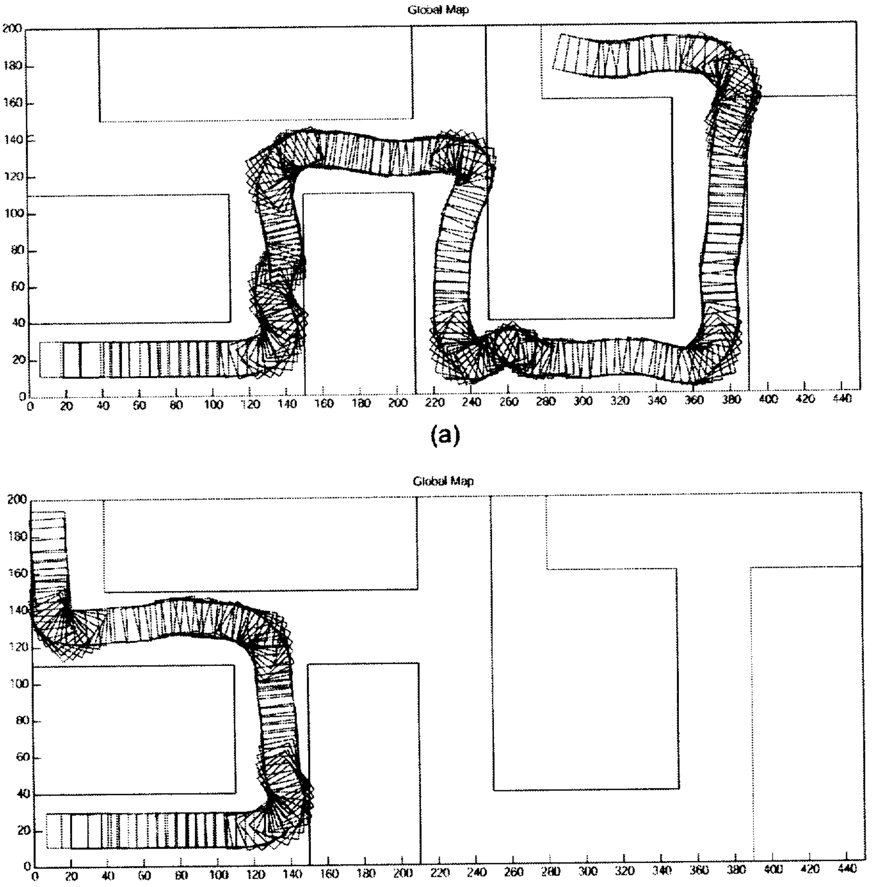
When the start position and the target position are node 1 and node 11 respectively, navigation results are shown in figure 11, (a) and 12, (a). We can see more smooth motion at the direction decided by the node information in builded map than complex and unstable motion in exploration process. When the target position is node 5, the results are shown in figure 11, (b) and 12, (b).



(a)

(b)

Fig. 7.11 Navigation using a topological map(simulation)



(a)

(b)

Fig. 7.12 Navigation using a topological map(implementation)

Ⅶ. Analysis of implementation results

To verify the algorithm proposed in this paper, we have assumed the obstacle with simple one such as wall. And the implementation environment is same with simulation to compare implementation results with simulation results. The movable directions estimated at the first simulation and implementation are the states of obstacles. That is, the obstacles around the mobile robot is how those orient. It is important information to build the map about unknown environment. As compared the simulation results with the implementation results, a robot motion is somewhat complex due to the hardware property and sensor noise. Therefore the moving distance of the mobile robot is much far than it of simulation. There also exists somewhat direction error between the real moving direction and the estimated moving direction due to the wheel slip. The more a mobile robot moves distant, the more it is accumulated much direction error. However, it is not so much error that a mobile robot mis-recognize the environment. That is, a mobile robot can arrive at the target position using rough direction and distance because it has topological map, and it can also decrease the error through revisit.

IX. Conclusions

The ultrasonic sensors inherently have the range error due to the specular reflection. To decrease this error, we have estimated the obstacle states using the local minimum values among sensed values and jacobian. Through the simulations and implementations, we have verified the efficiency about collision avoidance using the estimated states of obstacles. And this information was also utilized to calculate the movable direction through Fuzzy Inference System and the desired direction is selected by *Turn side selector*(binary decision tree). Simultaneously, the states of obstacles are based on the topological map building. Even though the builded map has the difference between the simulation and implementation due to the property of real system, we can analyze this is not the problem of proposed algorithm but hardware problems and we can confirm that a mobile robot can reach the target position using rough direction and distance through the implementation.

In this paper, we focus in transferring the human being's capability into a mobile robot. We divided the capability into four processes(Exploration process, Decision process, Behavior process and Learning process). The Exploration process is to generate topological map by ultrasonic sensors. The Decision process is to select a proper action based on *Turn side selector* and the builded map. The Behavior process is to actually implement in accordance with fuzzy rules. The learning process is to update parameters of the node information of *Graph*. Simulation and implementation results show that our proposed algorithms works well in the navigation problems.

References

[1] J. Modayil, P. Beeson, and B. Kuipers, "Using the topological skeleton for scalable global metrical map-building," in Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Sendai, Japan, pp. 1530-1536, 2004.

[2] Francesco Savelli and Benjamin Kuipers, "Loop-closing and planarity in topological map-building," in Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Sendai, Japan, pp. 1511-1517, 2004.

[3] Choset, H., Nagatani, K., "Topological Simultaneous Localization and Mapping (SLAM): Toward Exact Localization Without Explicit Localization," In IEEE Trans. Robot. Automat, vol. 17, No. 2, pp. 125-137, 2001.

[4] Sylvia C. Wong and Bruce A. MacDonald, "A topological coverage algorithm for mobile robots", In Proc. IEEE/RSJ Int Conf. on Intelligent Robots and Systems, Las Vegas, USA, vol. 4, pp 1685-1690, 2003.

[5] Nicola Tomatis, I. Nourbakhsh, and R. Siegwart, "Simultaneous localization and map-building: a global topological model with local metric maps," Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Maui, Hawaii, USA, pp.421-426, 2001.

[6] T.Bailey, E. M. Nebot, J. K. Rosenblatt and H. F. Durrant-Whyte, "Data

Association for Mobile Robot Navigation : A Graph Theoretic Approach," Proc. IEEE Int. Conf. on Robotics and Automation, San Francisco, pp.2512-2517, 2000.

[7] Chang-Hyuk Choi, Jae-Bok Song, Woojin Chung and Munsang Kim "Topological Map Building Based on Thining and Its Application to Localization," Proc. IEEE/RSJ Int. Conf on Intelligent Robots and systems, Switzerland, pp. 552-557, 2002.

[8] Bang-Yun Ko, Jae-Bok Song, Sooyong Lee, "Real-Time Building of a Thinning-Based Topological Map with Metric Features," Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 1524-1529, 2004.

[9] Chang-Hyuk Choi, Jin-sun Lee, Jae-Bok Song, Woojin Chung, Munsang Kim, Sung-Kee Park, Jong-Suk Choi, "Topological Map Building for Mobile Robot Navigation," Journal of Control, Automation, and Systems, Vol. 8, No. 6, pp.492-497, 2002.

[10] Takehisa Yairi and Koichi Hori, "Map Building By Mobile Robots With Incomplete and Qualitative Observation", SICE Annual Conference, pp.1482-1486, 2003.

[11] Young-Geun Kim and Hakil Kim, "Map Building for Path Planning of an Autonomous Mobile Robot Using a single Ultrasonic Sensor", Trans. KIEE. vol. 51D, No. 12, pp.577-582, 2002.

[12] A. Elfes, "Sonar based real-world mapping and navigation," IEEE J. Robotics Automat. vol. RA-3, No. 3, pp.249-265, 1987.

[13] Byung Kook Kim, Hyoung Jo Jeon, "A study on world map building for mobilerobots with tri-aural ultrasonic sensor system", IEEE Int. Conf. on Robotics and Automation,, pp.2907-2912, 1995.

[14] Y. Han and H. Hahn, "Localization and classification of target surfaces using two pairs of ultrasonic sensors," in Proc. IEEE Int. Conf. Robot and Automation, Detroit, Michigan, pp.637-643, 1999.

[15] Alessandro Correa Victorino, Patrick Rives, Jean-Jacques Borrelly, "Mobile robot navigation using a sensor based control strategy," in Proc. IEEE Int. Conf. Robot and Automation, Seoul, Korea, pp.2753-2758, 2001.

[16] M. Beetz, T. Arbuckle, T. Belker, M. Bennewitz, W. Burgard, AB Cremers, D. Fox, H. Grosskreutz, D. Haehnel, D. Schulz, "Integrated Plan-based Control of Autonomous Robots in Human Environments," IEEE Intelligent Systems, vol. 16, Issue. 5, pp.56-65, 2001.

[17] Haris Baltzakis and Panos Trahanias, "An iterative approach for building feature maps in cyclic environments," in Proc. IEEE Int. conf. on Intelligent Robots and Systems, Lausanne, Switzerland, pp.576-581, 2002.

[18] J.A. Janet, S.M. Scoggins, M.W. White, J.C. Sutton, III, E. Grant and

W.E. Snyder, "Using a Hyper-Ellipsoid Clustering Kohonen for Autonomous Mobile Robot Map Building, Place Recognition and Motion Planning," Proc. IEEE Int. Conf. on Neural Networks, pp. 1699-1704, 1997.

[19] J. A. Castellanos, J. M. M. Montiel, J. Neira, and J. D. Tardós, "The SPmap: A probabilistic framework for simultaneous localization and map building," IEEE Trans. Robot. Automat., vol. 15, No. 5, pp.948 - 952, 1999.

[20] F.H. Wullschleger, K.O. Arras, and S.J. Vestli, "A flexible exploration framework for map building," In Proc . of the Third European Workshop on Advanced Mobile Robots, pp.49-56, 1999.

[21] Kimon P. Valavanis, Timothy Hebert, Ramesh Kolluru, and Nikos Tsourveloudis, "Mobile Robot Navigation in 2-Dynamic Environments Using an Electrostatic Potential Field," IEEE Trans. Syst., Man, Cybern., vol. 30, pp.187-196, 2000.

[22] Nikos C. Tsourveloudis, Kimon P. Valavanis, and Timothy Hebert, "Autonomous Vehicle Navigation Utilizing Electrostatic Potential Fields and Fuzzy Logic," IEEE Trans. Robot. Automat., vol. 17, pp.490-497, 2001.

[23] Y. K. Hwand and N. Ahuja, "A potential field approach to path planning," IEEE Trans. Robot. Automat., vol. 6, pp.23-32, 1992.

[24] E. Rimon and D. Koditschek, "Exact robot navigation using artificial

potential functions," IEEE Trans. Robot. Automat., vol. 8, pp.501-518, 1992.

[25] Choset, H., and Burdick, J. W., "Sensor based planning, Part Ⅰ: The Generalized Voronoi Graph," Proc. IEEE Int. Conf. on Robotics and Automation, Nagoya, Japan, pp.1649-1655, 1995.

[26] Choset, H., and Burdick, J. W., "Sensor based planning, Part Ⅱ: Incremental Construction of the Generalized Graph," Proc. IEEE Int. Conf. on Robotics and Automation, Nagoya, Japan, pp.1643-1648, 1995.

[27] Choset, H., Konuksven, I., "Sensor based planning, A control law for generationg the generalized voronoi graph," Proc. IEEE Int. Conf. on Robotics and Automation, Monterey, CA, pp.333-338, 1997.

[28] Choset, H., Konuksven, I., Burdick, J., "Mobile robot navigation: Issues in Implementating the generalized voronoi graph in the plane," Proc. IEEE/SICE/RSJ Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems, pp.241-248, 1996.

[29] Roque, W. L. and Doering, D., "Multiple Visiting Stations Trajectory (Re)Planning for Mobile Lab Robots Based on Global Vision and Voronoi Roadmaps," in Proc. of the 33rd International Symposium on Robotics, Stockholm, Sweden, 2002.

[30] Keiji Nagatani,Yosuke Iwai, Yutaka Tanaka, "Sensor Based Navigation

for car-like mobile robots based on Generalized Voronoi Graph," Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Maui, Hawaii, USA, pp.1017-1022, 2001.

[31] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," IEEE Computer. vol. 22, No. 6, pp.46-57, 1989.

[32] T. Fukuda and N. Kubota, "An intelligent robotic system based on fuzzy approach," Proc. IEEE, vol. 87, pp.1448-1470, 1999.

[33] C. Barat, and N. Ait Oufroukh, "Classification of indoor environment using only one ultrasonic sensor," IEEE Instrumentation and Measurement, Technology Conference, pp. 1750-1755, 2001.

[34] C. Barat, E. Colle, N. Ait Oufroukh, " Toward a versatile ultrasonic sensor", 8th IEEE Emerging
Technologies and Factory Automation (ETFA), vol. 1 pp. 383-391, Juan les pins, France, 2001.

[35] Takahiro Sugiyama, Keiichi Abe. "Edge Detection Method Based on Edge Reliability with Fixed Thresholds: Consideration of Uniformity and Gradation," 15th International Conference on Pattern Recognition (ICPR'00), vol. 3, pp.656-659, 2000.

[36] P. Parodi and V. Torre, "A Linear Complexity Procedure for Labelling

Line Drawings of Polyhedral Scenes Using Vanishing Points," Proc. of 4th ICCV, Berlin, pp.291-295, 1993.

[37] M. Straforini, C. Coelho, M. Campani, and V. Torre, "The Recovery and Understanding of a Line Drawing from Indoor Scenes," IEEE Trans. Pattern Anal. Mach. Intell. vol. 14, No. 2 pp.298-303, 1992.

[38] Morimichi Nishigaki, Masakazu Saka, Tomoyoshi Aoki, Hiromitsu Yuhara, Makoto Kawai, "Fail Output Algorithm Of Vision Sensing," In Proc. IEEE Intelligent Vehicles Symposium, Detroit USA, pp. 581-584, 2000.

[39] David Beymer, "Person counting using stereo," In Workshop on. Human Motion, pp.127-133, 2000.

[40] M. Xie, C. M. Lee, Z. Q. Li, S. D. Ma, "Depth Assessment Using Qualitative Stereo Vision," IEEE First International Conference on Intelligent Processing Systems, Beijing, vol. 2, pp.1446-1449, 1997.

[41] K. Yamaguchi, Y. Nagaya, K. Ueda, H. Nemoto, M. Nakagawa, "A method for identifying specific vehicles using template matching," Proc. IEEE/IEEJ/JSAI Int. Conf. on Intelligent Transportation Systems, pp.8 - 13, 1999.

[42] Fichera, O., Pellegretti, P., Roli, F., Serpico, S.B, "Automatic Acquisition of Visual Models for Image Recognition", ICPR'92, vol. I, pp.95-98, 1992.

[43] L.M. Wong, C. Dumont and M.A. Abidi, "A Next-Best-View System in a 3D Object Modeling Task," in Proc. IEEE Int. Symp. Comput. Intell. Robot. Autom., Monterey, CA, pp.306-311, 1999.

[44] H.K. Alfred and K.C. Wong, "Robust scene reconstruction from lines and points in three uncalibrated color image," in Proc. IEEE Int. conf. on Intelligent Robots and Systems, Maui, Hawai, USA, pp.557-562, 2001.

[45] H. Kobayashi, M. Yanagida, "Moving object. detection by an autonomous guard robot" Proc. the 4th IEEE Int. Workshop on Robot and Human Communication, TOKYO, pp.323-326, 1995.

[46] S. Hirata, Y. Shirai and M. Asada, "Scene interpretation using 3D information extracted from monocular colour images," Proc. IEEE/RSJ International Conference on Intelligent Robots and. Systems, pp. 1603-1610, 1992.

[47] Guilherme N. DeSouza , Avinash C. Kak, "Vision for Mobile Robot Navigation: A Survey" IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24 No. 2, pp.237-267, 2002.

[48] Siripun Thongchai, "Behavior based learning fuzzy rules for mobile robots," in Proc. of the American Control Conference, pp.995-1000, 2002.

[49] Choset, H., and Burdick, J. W., "Sensor based planning and nonsmooth

analysis," Proc. IEEE Int. Conf. on Robotics and Automation, San Diego, CA, pp.3034-3041, 1994.

[50] D.VAN ZWYNSVOORDE, T.SIMEON, R.ALAMI, "Building topological models for navigation in large scale environments," Proc. IEEE Int. Conf. on Robotics and Automation, Seoul, Korea, pp.4256-4261, 2001.

Appendix

A. Stepping motor control source code

```
#include <io.h>
#include <ina90.h>
#include <sig-avr.h>
#include <interrupt.h>
#include <iom128.h>

#define SYSTEM_CLOCK 16000000

void putch(unsigned char ch);
unsigned char getch(void);

void Delay_ms(unsigned int time_ms);
void Delay_us(unsigned char time_us);

void RightWheelOp(unsigned char pd);
void LeftWheelOp(unsigned char pd);
void TransOfData(void);
unsigned char PeriodSetting(unsigned char cP, unsigned char pP);

volatile unsigned char Data[4];
volatile unsigned char rightCnt;
volatile unsigned char leftCnt;
volatile unsigned char cnt = 0;

volatile unsigned char cur_rPd = 0;
volatile unsigned char cur_lPd = 0;
volatile unsigned char pre_rPd = 0;
volatile unsigned char pre_lPd = 0;

volatile unsigned int TransFlag = 0;
```

```
SIGNAL(SIG_UART0_RECV)
{
  if(cnt<4)
  {
    Data[cnt] = getch();
    cnt = cnt+1;
  }
  else
  {
    if((Data[0]==0xff)&&(Data[3]==0xf0))
    {
      cur_rPd = Data[1];
      cur_lPd = Data[2];
      TransFlag = 1;

      if((cur_rPd==0)&&(cur_lPd==0))
      {
        TIMSK = 0x00;    // Right wheel disable
        ETIMSK = 0x00;  // Left wheel disable

        pre_rPd = 0;
        pre_lPd = 0;
      }
      else
      {
        TIMSK = 0x10;    // Right wheel enable
        ETIMSK = 0x10;    // Left wheel enable
      }
    }
    cnt = 0;
  }
}
```

```
SIGNAL(SIG_OUTPUT_COMPARE1A)
{
  if((PORTB&0x20)==0)
  {
    pre_rPd = PeriodSetting(cur_rPd, pre_rPd);
    RightWheelOp(pre_rPd);
  }
  PORTB = PORTB^0x20; // Right wheel
  rightCnt = rightCnt + 1;
}

SIGNAL(SIG_OUTPUT_COMPARE3A)
{
  if((PORTB&0x40)==0)
  {
    pre_lPd = PeriodSetting(cur_lPd, pre_lPd);
    LeftWheelOp(pre_lPd);
  }
  PORTB = PORTB^0x40; // Left wheel
  leftCnt = leftCnt + 1;
}

void UART_Init(unsigned long BaudRate)
{
  outp(0x00, UCSR0A);

  outp(0x98, UCSR0B);
  outp(0x06, UCSR0C);

  // Setting BaudRate
  outp(0x00, UBRR0H);
  outp((SYSTEM_CLOCK/BaudRate/16 - 1),UBRR0L);
}
```

```
void Port_Init(void)
{
  //Port_A setting up output : Miscellaneous output setting
  outp(0xFF, DDRA);

  //Port_B setting up output : Timer 1 setting
  outp(0xFF, DDRB);
  //Port_E setting up output : Timer 3 setting
  outp(0xFF, DDRE);

  // wait for system stabilization
  Delay_ms(50);
}

unsigned char getch(void)
{
  while(!(UCSR0A & (1 <<RXC)));
  return UDR0;
}

void putch(unsigned char ch)
{
  while( !(UCSR0A  & (1<<UDRE)) );
  outp(ch, UDR0);
}

void puts(unsigned char *str)
{
  while(*str != ’\0’)
  {
    putch(*str++);
  }
}
```

```
//------------------- DELAY ------
void Delay_us(unsigned char time_us)
{
  register unsigned char i;

  for(i = 0; i < time_us; i++)
  {
    asm volatile(" PUSH R0");
    asm volatile(" POP  R0");
    asm volatile(" PUSH R0");
    asm volatile(" POP  R0");
    asm volatile(" PUSH R0");
    asm volatile(" POP  R0");
  }
}

void Delay_ms(unsigned int time_ms)
{
  register unsigned int i;

  for(i = 0; i < time_ms; i++)
  {
    Delay_us(250);
    Delay_us(250);
    Delay_us(250);
    Delay_us(250);
  }
}
```

```
void RightWheelOp(unsigned char p)
{
  unsigned int pd, dt;

  pd = (200-p)*10+200;
  dt = pd/4;
  OCR1AH = (pd>>8);
  OCR1AL = pd&0xFF;
  OCR1BH = (dt>>8);
  OCR1BL = dt&0xFF;
}

void LeftWheelOp(unsigned char p)
{
  unsigned int pd, dt;

  pd = (200-p)*10+200;
  dt = pd/4;
  OCR3AH = (pd>>8);
  OCR3AL = pd&0xFF;
  OCR3BH = (dt>>8);
  OCR3BL = dt&0xFF;
}
unsigned char PeriodSetting(unsigned char cP, unsigned char pP)
{
  if(cP>pP)
    pP = pP+1; // save pre-period
  else if(cP<pP)
    pP = pP-1; // save pre-period

  return cP;
}
```

```
void TransOfData(void)
{
  if(TransFlag==1)
  {
    putch(0xff);
    putch(rightCnt);
    rightCnt = 0; // initialize
    putch(leftCnt);
    leftCnt = 0; // initialize
    putch(0xf0);
    TransFlag = 0;
  }
}

int main(void)
{
  unsigned char RESET, ENABLE, DIR_L, DIR_R;

  RESET = 0x00;
  ENABLE = 0x01;
  DIR_R = 0x02;
  DIR_L = 0x03;

  outp(0x00, MCUCR);

  sbi(PORTA, RESET);  // reset
  sbi(PORTA, ENABLE); // motor enable
  cbi(PORTA, DIR_R);
  sbi(PORTA, DIR_L);
  Port_Init();                // Port initialize
  UART_Init(19200);             // Serial port setting
```

```
 // Timer 1 setting : Left wheel
 TCCR1A = 0x00;
 TCCR1B = 0x0C;
 TCCR1C = 0x00;
 // Timer 3 setting : Right wheel
 TCCR3A = 0x00;
 TCCR3B = 0x0C;
 TCCR3C = 0x00;
 ////////////////////////////////
 TIMSK = 0x00;  // OCIE1A disable
 ETIMSK = 0x00; // OCIE3A disable
 TIFR = 0x00;
 ETIFR = 0x00;

 // initialization of variables
 rightCnt = 0;
 leftCnt = 0;

 Data[0] = 0;
 Data[1] = 0;
 Data[2] = 0;
 Data[3] = 0;

 sei();

 while(1)
 {
   if(TransFlag==1)
     TransOfData(); // data transmission
 }

 return 0;
}
```

B. Ultrasonic sensor source code

```
#include <io.h>
#include <ina90.h>
#include <sig-avr.h>
#include <interrupt.h>
#include <iom8.h>


#define SYSTEM_CLOCK 16000000
#define DATA_LENGTH 100

volatile unsigned char cmd = '\0';
void TriggerPulse(void);

void DigitToChar(unsigned char val);

volatile unsigned char dist = 0;
volatile unsigned int INTCnt = 0;

void putch(unsigned char ch);
void puts(unsigned char *str);
unsigned char getch(void);

void Delay_ms(unsigned int time_ms);
void Delay_us(unsigned char time_us);


SIGNAL(SIG_UART_RECV)
{
  cmd = getch();
}
```

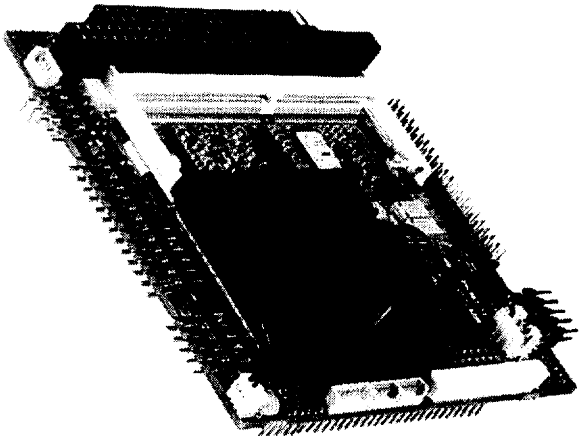
```
unsigned char getch(void)
{
  while(!(UCSRA & (1 <<RXC)));
  return UDR;
}

SIGNAL(SIG_OVERFLOW0)
{
  INTCnt++;
}

SIGNAL(SIG_INTERRUPT0)
{
  INTCnt = 0; // reinitialize
  TCNT0 = 0;

  TIMSK=0X01; // Timer 0 overflow enable
}

SIGNAL(SIG_INTERRUPT1)
{
  TIMSK=0x00;  // Timer 0 overflow disable

  // calibration of range
  dist = (INTCnt*16*34000)/2000000; // unit : cm
  if(dist>=0xc8)
    dist = 0xc8;
  putch(0xff);  // head
  putch(dist);
  putch(0xf0);  // tail
}
```

```
void UART_Init(unsigned long BaudRate)
{
  outp(0x00, UCSRA);

  outp(0x98, UCSRB); //98
  outp(0x06, UCSRC);
  outp(0x00, UBRRH);  // Setting BaudRate
  outp((SYSTEM_CLOCK/BaudRate/16 - 1),UBRRL);
}

void Port_Init(void)
{
  cbi(DDRD,2);  // Ex-INT 0 : input
  cbi(DDRD,3);  // Ex-INT 1 : input
  sbi(DDRD,4);  // trigger pulse output

  Delay_ms(50);  // wait for system stabilization
}

void putch(unsigned char ch)
{
  while( !(UCSRA  & (1<<UDRE)) );
  outp(ch, UDR);
}

void puts(unsigned char *str)
{
  while(*str != '\0')
  {
    putch(*str++);
  }
}
```

```
void Delay_us(unsigned char time_us)
{
  register unsigned char i;
  for(i = 0; i < time_us; i++)
  {
    asm volatile(" PUSH R0");
    asm volatile(" POP  R0");
    asm volatile(" PUSH R0");
    asm volatile(" POP  R0");
    asm volatile(" PUSH R0");
    asm volatile(" POP  R0");
  }
}

void Delay_ms(unsigned int time_ms)
{
  register unsigned int i;
  for(i = 0; i < time_ms; i++)
  {
    Delay_us(250);
    Delay_us(250);
    Delay_us(250);
    Delay_us(250);
  }
}

void TriggerPulse(void)
{
  sbi(PORTD,4);
  Delay_us(10);
  cbi(PORTD,4);
  Delay_ms(40);
}
```

```
int main(void)
{
  //outp(0x00, MCUCR);
  cli();

  // Timer 0 overflow enaable
  TIMSK=0x01;
  TCCR0=0X01; // no prescaling

  // External interrupt setting
  GIMSK = 0XC0;     // INT 0:rising, INT 1:down
  MCUCR = 0X0B;     // INT 0,1

  Port_Init();                    // Port initialize
  UART_Init(19200);                 // Serial port setting

  sei();

  while(1)
  {
    if(cmd=='g')
    {
      // generation of trigger pulse
      TriggerPulse();
      cmd = '\0';  // null command
    }
  }

  return 0;
}
```

C. MPOSlcd7



● Functional Specifications

◎ Processors
- Mobile PentiumⅢ processor - 500MHz or 700MHz
- Ultra Low Power(ULP) Celeron processor - 300MHz

◎ Cache (on die)
- PentiumⅢ - 256KB
- celeron - 128KB

◎ Bus
- 100MHz CPU bus (PentiumⅢ or Celeron)
- 100MHz or 133MHz memory bus (PentiumⅢ or Celeron)

◎ VIA VT8606/VT82C686B Super I/O (north bridge / south bridge)

◎ Memory
- One 144-pin SODIMM
- 3.3V PC-133 or PC-100 unBuffered SDRAM, up to 512MB

◎ Two serial ports
- COM1 and COM2 are fully RS232C compatible

◎ One Parallel Port
- LPT1
- Enhanced Parallel Port (EPP) and Extended Capabilities Port (ECP) with bidirectional capability

◎ Floppy drive interface

◎ Enhanced Intelligent Drive Electronics (EIDE) : One Peripheral Component Interconnect (PCI) Bus Master, IDE ports (up to two devices) support :
- Ultra Direct Memory Access 33 (U-DMA33) mode
- Programmed Input / Output (PIO) modes up to Mode 4 timing
- Multiword DMA Mode 0,1,2 with independent timing

◎ Two Universal Serial Bus (USB) ports
- Two USB 1.1 ports (UHCI)
- USB legacy keyboard and PS/2 mouse support
- USB floppy-boot and CDROM-boot support

◎ Onboard Ethernet : Davicom 9102A PCI, 10/100BASE-TX LAN

◎ ProSavage4 AGP4x graphics controller integrated in VIA VT 8606
- Up to 32MB Graphics RAM (UMA)
- Cathode ray tube (CRT) and low voltage differential signaling (LVDS) liquidcrystal Display (LCD) interfaces

◎ BIOS : Phoenix, 512KB Flash-BIOS

◎ NV-EEPROM for CMOS-setup retention without battery
- PS/2 keybaord controller
- PS/2 mouse interface
- Watchdog timer (WDT)
- Real-time colck (requires external battery)

- System RAM
- System ROM (BIOS)
- Direct Memory Access (DMA)
- Counters
- Interrupt controllers

● Mechanical Specifications

◎ Dimensions : 95.0 x 90.0 mm (3.7" x 3.7")

● Electrical Specifications

◎ Supply Voltage : 5V DC ±5%
◎ Supply Voltage Ripple : 100mV peak to peak 0-20MHz

D. AVR ATmega 128

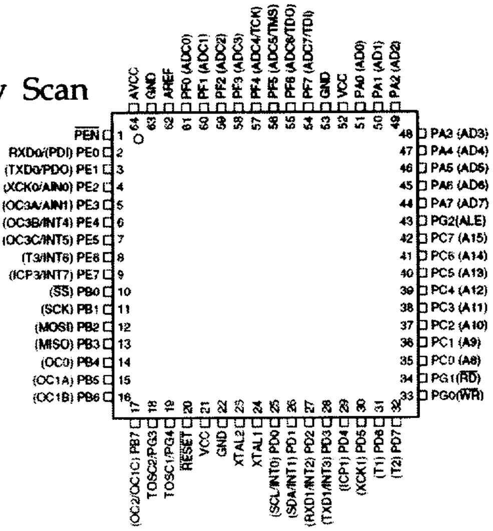
◎ Timers / Counters
- Timer/Counters (8-bit): 2
- Pulse Width Modulator: 6+2 ch(s)
- Timer/Counters (16-bit): 2
- Real Time Counter
- Watchdog Timer with On-chip Oscillator

◎ Serial I/O
- 2-wire Serial Interface (I2C compatible)
- Full Duplex Serial Peripheral Interface (SPI)
- Full Duplex USART: 2

◎ Programming Modes
- In-System Programming via JTAG port
- High Voltage Parallel Programming (12V)
- Self-Programming via on-chip Boot Program
- In-System Programming via SPI Port

◎ Memory
- External data memory interface (64kB)
- General Purpose Registers (Accumulators): 32

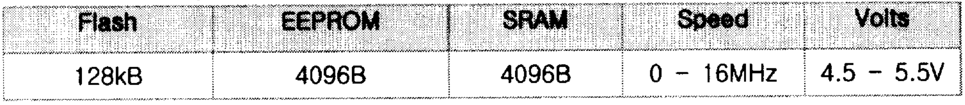
◎ MCU Specific
- Brown-out Detection
- IEEE 1149.1 (JTAG) Boundary Scan
- Sleep Modes: 6
- Interrupts, External pins: 8
- On-Chip Debug support via JTAG port
- Interrupts: 29
- On Chip Oscillator
- I/O Pins: 53
- Hardware Multiplier
- Fully Static Operation
- Power-on Reset



◎ Analog I/O

- Analog Comparator
- Analog-to-Digital Converter (10-bit): 8 ch(s)
- Analog Gain Stage: 2 ch(s)

◎ Availability



Flash	EEPROM	SRAM	Speed	Volts
128kB	4096B	4096B	0 - 16MHz	4.5 - 5.5V

E. AVR ATmega8

◎ Nonvolatile Program and Data Memories

- 8K Bytes of In-System Self-Programmable Flash Endurance : 10,000 Write/Erase Cycles
- Optional Boot Code Section with Independent Lock Bits In-System Programming by On-chip Boot Program True Read-While-Write Operation
- 512 Bytes EEPROM Endurance: 100,000 Write/Erase Cycles
- 1K Byte Internal SRAM
- Programming Lock for Software Security

◎ Peripheral Features

- Two 8-bit Timer/Counters with Separate Prescaler, one Compare Mode
- One 16-bit Timer/Counter with Separate Prescaler, Compare Mode and Capture Mode
- Real Time Counter with Separate Oscillator
- Three PWM Channels
- 8-channel ADC in TQFP and MLF package Six Channels 10-bit Accuracy Two Channels 8-bit Accuracy
- 6-channel ADC in PDIP package Four Channels 10-bit Accuracy Two Channels 8-bit Accuracy
- Byte-oriented Two-wire Serial Interface
- Programmable Serial USART
- Master/Slave SPI Serial Interface
- Programmable Watchdog Timer with Separate On-chip Oscillator
- On-chip Analog Comparator

◎ Special Microcontroller Features

- Power-on Reset and Programmable Brown-out Detection
- Internal Calibrated RC Oscillator

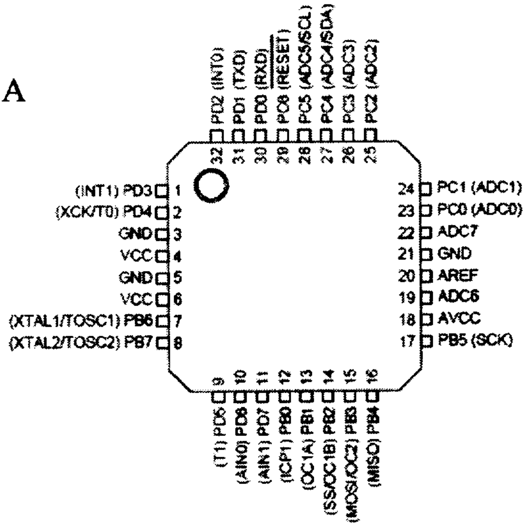
- External and Internal Interrupt Sources
- Five Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, and Standby

◎ I/O and Packages
- 23 Programmable I/O Lines
- 28-lead PDIP, 32-lead TQFP, and 32-pad MLF

◎ Operating Voltages
- 4.5 - 5.5V (ATmega8)

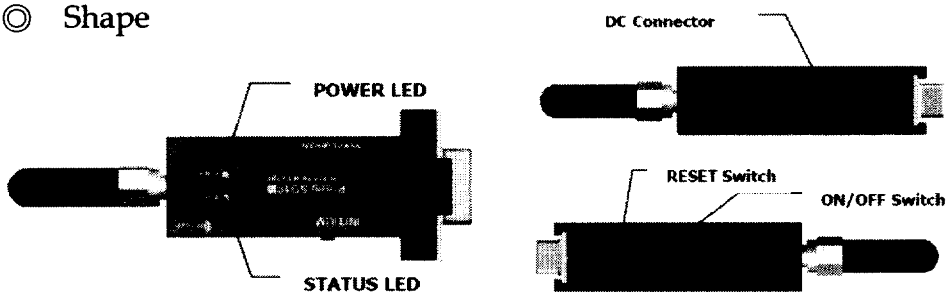
◎ Speed Grades
- 0 - 16 MHz (ATmega8)

◎ Power Consumption at 4 Mhz, 3V, 25°C
- Active: 3.6 mA
- Idle Mode: 1.0 mA
- Power-down Mode: 0.5 μA

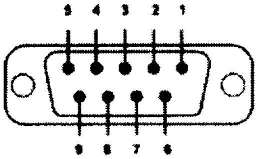


F. Bluetooth (Promi-SD202)

◎ Shape



◎ Connector



Pin	Signal	Direction
1	CD	Output
2	TxD	Output
3	RxD	Input
4	DSR	Input
5	GND	-
6	DTR	Output
7	CTS	Input
8	RTS	Output
9	Vcc	Input

◎ Bluetooth Interface

Bluetooth specification	Promi-SD202
Level	18 dBm
Range	~100m
Bluetooth protocols	RFCOMM, L2CAP, SDP
supported Profiles	General Access Profile Serial Port Profile

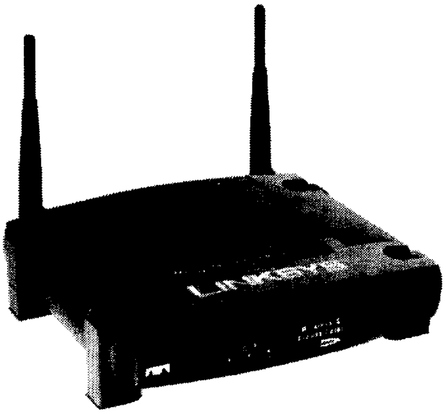
◎ Power Supply , Consumption Current

Power Supply		4V ~ 12V, 150mA minimum
condition of Baud Rate	9600 bps	40 mA
	115200 bps	72 mA

G. Wireless LAN

● Wireless-G Access Point

◎ Set up a high-speed Wireless-G network
◎ Data rates up to 54Mbps
◎ Advanced wireless security with 128-bit WPA encryption and wireless MAC address filtering



Wireless-G
Access Point

● Wireless-G USB Network Adapter

◎ A wireless network at up to 54Mbps when used with a USB2.0 port

◎ Wireless communications are protected with 128-bit Wi-Fi Protected Access(WPA) encryption

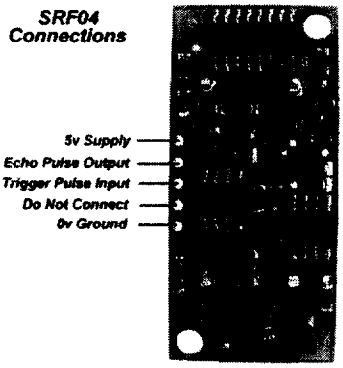
Wireless-G
USB Network Adapter

H. SRF04-Ultra-Sonic Ranger

◎ Voltage
- 5V only required.

◎ Current
- 30mA Typ. 50mA Max.

◎ Frequency
- 40KHz



◎ Max Range
- 3m

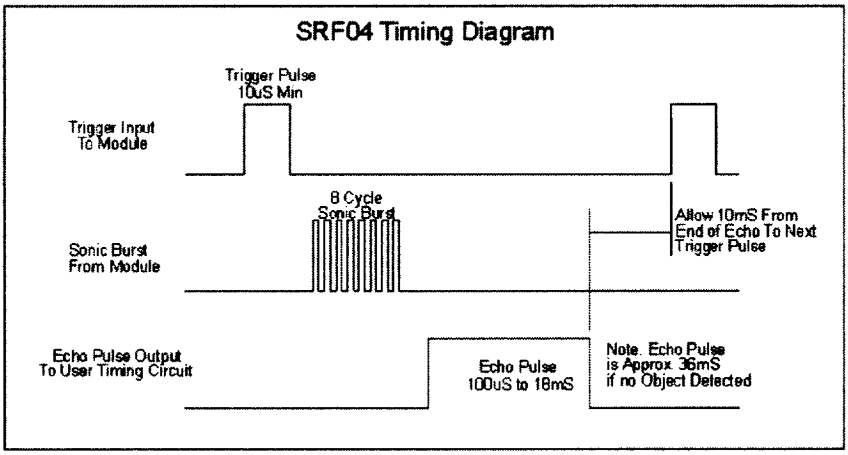
◎ Min Range
- 3Cm

◎ Sensitivity
- Detect 3Cm diameter broom handle at > 2m

◎ Input Trigger
- 10uS Min. TTL level pulse

◎ Echo Pulse
- Positive TTL level signal, width proportional to range

◎ Small Size
- 43mm*20mm*17mm height



◎ Calculating the Distance

The SRF04 provides an echo pulse proportional to distance. If the width of the pulse is measured in uS, then dividing by 58 will give you the distance in cm, or dividing by 148 will give the distance in inches. uS/58=cm or uS/148=inches

I. Battery

◎ 2600mAh 18.5V(5S1P) Lithium Polymer Battery.

◎ Nominal Voltage : 18.5V(3.7V * 5)

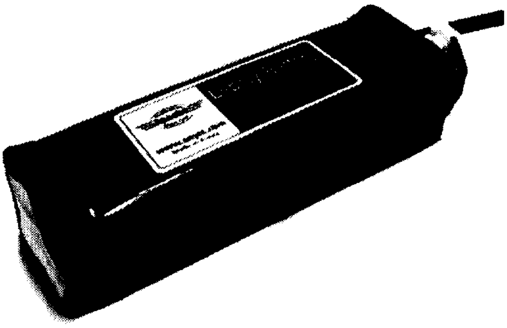
◎ Capacity (mAh) : 2600mA

◎ Temperature : Charge 0 ~ +45℃, Discahrge -20 ~ +65℃

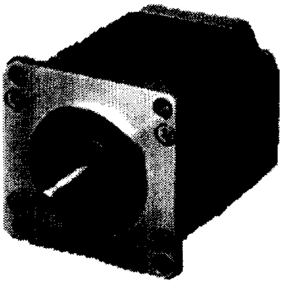
◎ Charge rate : 1C

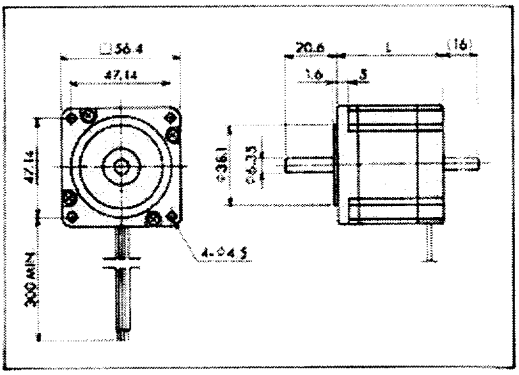
◎ Dimension (mm) : 33.5 mmT x 41.0mmW x126.0mmL

◎ Weight : 310g



J. Stepping Motor (SST58D5200)





item	specification	unit
Step Angle	1.8	DEG
Voltage	4.8	V
Current	2	A / Φ
Resistance	2.4	Ω / Φ
Inductance	5.1	mH / Φ
Holding Torque	12	kgf-cm MIN
Rotor Inertia	430	g · ㎠
Number of leads	6	LEAD
Weight	1.1	kg
Dimension	77	L

Acknowledgements

I would like to express my sincere appreciation to my advisor, Professor Doo-Sung Ahn for introducing me to intelligent control theory and supporting me throughout graduate course. Without his guidance and continuous encouragement, this thesis would never have been completed. I would like thank to my dissertation committee for valuable comments : Professor Won-Chang Lee, Professor Jin-Young Kim, Professor Sang-Woon Shin and Professor Young-Seok Jung.

감사의 글

너무나 멀어 보이고 막연하게만 느껴지던 목표를 이룬 지금, 뒤돌아보지 않고 달려온 세월이 아득하기만 합니다. 앞으로 쉬지 않고 또 얼마를 더 달려야 할지 모르기에 논문이 완성된 이 시점에서 잠시 여유를 가지고 고마운 분들에게 감사의 마음을 전하고자 합니다.

짧지 않은 대학원 생활을 하는 동안 진정한 학문의 의미를 깨닫게 해주시고 어버이처럼 보살펴 주신 안두성 지도교수님께 감사드립니다. 선비의 자세로 학문에 열중하시는 진정한 학자의 모습을 교수님을 통해 배웠으며, 항상 교수님의 가르침대로 세상을 살아가겠습니다. 그리고 바쁘신 와중에도 저의 논문 심사를 맡아 부족한 부분을 채워주신 이원창 교수님, 김진영 교수님, 신상운 교수님, 정영석 교수님께 깊은 감사의 말씀을 드립니다.

지난 10년 동안의 연구실 생활을 하면서 언제나 힘이 되어준 선후배들의 모습들 또한 눈에 선합니다. 언제나 바쁘다는 핑계로 연락 한 번 제대로 하지 못한 연구실 선배님들, 남일 형, 주형 형, 성택 형, 재경 형, 원희 형께 감사드립니다. 그리고 항상 친근한 맘으로 선배들을 위했던 사랑스런 후배님들, 용탁, 규상, 영원, 광수, 성우, 경수, 재호, 창우, 지영, 재욱, 정훈, 웅태에게 모두 고마운 맘을 전하며, 석사과정 김태형씨께도 감사드립니다. 특히 연구실의 정신적인 지주이시며, 개인적으로 많은 도움을 주신 효정 형께는 각별한 감사를 드립니다.

대학 생활을 잘 할 수 있도록 이끌어주신 동아리 선배님들, 형철 형, 진승 형, 태윤 형께도 감사드리며, 전국 각지에 흩어져 있지만 언

제나 힘든 일들을 같이 한 동아리 동기들, 길원, 진안, 수남, 영전, 대식, 은주, 은영, 영실, 희성, 유경에게 감사하며, 은하를 포함한 동아리 후배들 모두에게도 감사드립니다.

특히 직장생활을 하면서도 논문이 완성될 수 있게끔 각별한 신경을 써주신 SR-KOREA 신인승 사장님께 깊은 감사를 드립니다. 또한 SR-KOREA 가족들, 김재은 차장, 신종근 차장, 김형규 과장, 김형석 대리, 정봉익 주임, 권기옥씨, 이민성씨에게 감사드립니다.

공부를 한다는 핑계로 소홀했음에도 불구하고 많은 이해와 관심을 주신 가족들에게 진심으로 감사드립니다. 늘 자식 걱정이 앞서시는 아버님, 어머님께 조그마한 기쁨과 자랑이 되길 바라는 맘으로 이 논문을 바칩니다. 힘들면서도 언제나 동생이 원하는 것은 다 들어주신 큰 형님과 작은 형님, 그리고 큰형수님, 작은 형수님께 감사의 마음을 전합니다. 특히 항상 친근하고 다정다감하게 대해주시는 큰 누님, 작은 누님과 자형께도 감사드리며, 사랑스런 조카들에게도 고마운 맘을 전합니다. 아무것도 가진 것 없는 학생 신분임에도 불구하고 결혼을 허락해주시고, 부족한 사위지만 언제나 격려해주시고 염려해주신 장인어른과 장모님께 깊은 감사의 마음을 드리며, 처남에게도 감사드립니다.

마지막으로 10년간의 연애기간 동안 언제나 한결같이 변치 않고 나만 바라봐준 나의 사랑 나의 신부 스무살 소녀같은 혜진이에게 이 논문을 바칩니다.