# Sensor Data Fusion for Navigation of Mobile Robot With Collision Avoidance and Trap Recovery

Young-Su Jeon*, Byeong-Kyu Ahn**, and Tae-Yong Kuc***

* School of Information & Communication Engineering, Sungkyunkwan University, Suwon, Korea
(Tel : +82-31-290-7202; E-mail: youngsujeon@hanmail.net)

** School of Information & Communication Engineering, Sungkyunkwan University, Suwon, Korea
(Tel : +82-31-290-7202; E-mail: robo@allbod.com)

*** School of Information & Communication Engineering, Sungkyunkwan University, Suwon, Korea
(Tel : +82-31-290-7137; E-mail: tykuc@yurim.skku.ac.kr)

**Abstract :** This paper presents a simple sensor fusion algorithm using neural network for navigation of mobile robots with obstacle avoidance and trap recovery. The multiple sensors input sensor data to the input layer of neural network activating the input nodes. The multiple sensors used include optical encoders, ultrasonic sensors, infrared sensors, a magnetic compass sensor, and GPS sensors. The proposed sensor fusion algorithm is combined with the VFH(Vector Field Histogram) algorithm for obstacle avoidance and AGPM(Adaptive Goal Perturbation Method) which sets adaptive virtual goals to escape trap situations. The experiment results show that the proposed low-level fusion algorithm is effective for real-time navigation of mobile robot.

**Keywords :** Sensor Fusion, Neural Network, Mobile Robot, Obstacle Avoidance

## 1. INTRODUCTION

Since the complexity of environments and the large number of objectives make the mobile robot navigation difficult, two typical constraints which should be overcome in mobile robot navigation are obstacle avoidance and recovery of trap situation[3,4]. To overcome the difficulties, many mobile robots carry various multiple sensors used in such a way as complementary and/or redundant fashions. For example, many mobile robots gathers environmental data from optical encoders, ultrasonic sensors, magnetic sensors, laser scanners, active beacons, touch sensors, IR sensors, some kind of GPS sensors, and etc. In almost all cases, these multiple redundant sensors operate simultaneously and independently with different sensing principles and capabilities having different volumes of coverages. This necessitates some sensor fusion methods to gather and convert different sensor data to a unified and reliable logical data and environment model for mobile robot navigation. That is, the processed data are used in object detection, self-localization, world modeling, path planning, and etc., without mentioning of real-time obstacle avoidance and trap recovery[1,11,12].

This paper presents a neural network based sensor fusion algorithm for navigation of mobile robot with obstacle avoidance and trap recovery. The multiple sensors used in this experiment include optical encoders, ultrasonic sensors, infrared sensors, a magnetic compass sensor, and GPS sensors. The mobile robot is also equipped with VFH(Vector Field Histogram)[13] algorithm for obstacle avoidance and AGPM(Adaptive Goal Perturbation Method) which sets adaptive virtual goals to escape trap situations. The experiment results show that combined with the collision avoidance and trap recovery algorithms, the proposed low-level fusion algorithm by using a neural network is effective for real-time navigation of mobile robot.

## 2. MOBILE ROBOT AND INPUT DATA OF THE NEURAL NETWORK

### 2.1 Kinematic of The Mobile Robot

In this paper, we are considering 2-DOF mobile robot that has 4 assistance wheels and 2 drive wheels. [Fig 1] shows kinematic coordinate system of the mobile robot. Coordinate system $X_W$ - $Y_W$ refers to world coordinate system and $X_L$-$Y_L$ refers to the local coordinate system established for the mobile robot.

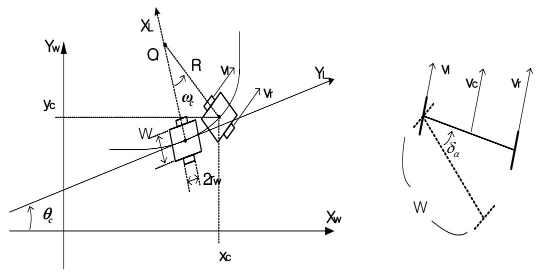


Fig 1 . Kinematics coordinate system of the mobile robot[3,4]

In this situation, we suppose that all the body parts of the mobile robot consists of pure iron and satisfies the condition of being non-slippery. In general, the movement of the mobile robot is decided towards the forward axis.

$$-\frac{dx}{dt}\sin\theta_c + \frac{dy}{dt}\cos\theta_c = 0 \tag{2.1}$$

Supposing that the center coordinate of the mobile robot is ( $x_{c,}$ $y_c$ ) and the present direction value is $\theta_c$ , then the position vector known as $\xi = [x_c \quad y_c \quad \theta_c]^T$ is derived. Meantime, supposing the strait linear velocity of the mobile robot is $v_l$ , $v_r$ and the angular velocity of the mobile robot is $\omega_l$ , $\omega_r$ , then the center velocity and angular velocity become $v_c$ , $\omega_c$ .

$$v_c = \frac{v_l + v_r}{2} = r_\omega(\frac{\omega_r + \omega_l}{2}) \tag{2.2}$$

$$\omega_c = \frac{v_r + v_l}{2R} = \frac{v_r - v_l}{W} \tag{2.3}$$

At this point, we will call the distance between wheels as "W" and the radius of gyration to the telegram center "$o_i$" as "R". From the formula above, the spinning velocity of both wheels and the central velocity of the mobile robot can be calculated.

$$\omega_r = \frac{2v_c + W\omega_c}{2r} \tag{2.4}$$

$$\omega_l = \frac{2v_c - W\omega_c}{2r} \tag{2.5}$$

$$\dot{x}_c = \frac{v_l + v_r}{2}\cos\theta_c = r_\omega(\frac{\omega_l + \omega_r}{2})\cos\theta_c \tag{2.6}$$

$$\dot{y}_c = \frac{v_l + v_r}{2}\sin\theta_c = r_\omega(\frac{\omega_l + \omega_r}{2})\sin\theta_c \tag{2.7}$$

As a result, the kinematics equation of the mobile robot is as following:

$$\dot{\xi} = \begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \dot{\omega}_c \end{bmatrix} = \begin{bmatrix} \cos\theta_c & 0 \\ \sin\theta_c & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{v_r + v_l}{2} \\ \frac{v_r - v_l}{W} \end{bmatrix} \tag{2.8}$$

**2.2 Dead Reckoning**

To know where the mobile robot moves from an early position to the present position, the mobile robot's carriage and position can be presumed through the rotation numbers of the motor inputted from the Encoder of the drive motor. Numerical formulas to calculate the above can be found from linear velocity and angular velocity from kinematics.

$$x_c(t) = x_c(t-1) + \dot{x}_c \times \nabla t = \int v_c \cos\theta_c dt \tag{2.9}$$

$$y_c(t) = y_c(t-1) + \dot{y}_c \times \nabla t = \int v_c \sin\theta_c dt \tag{2.10}$$

$$\theta_c(t) = \theta_c(t-1) + \dot{\theta}_c \times \nabla t = \int \dot{\theta}_c dt \tag{2.11}$$

However, since it is difficult to know the actual linear velocity and angular velocity of the mobile robot in reality, we will make the movement of the mobile robot to be approximately straight so that can make us to presume the posture of the mobile robot from the following formulas.

$$x_c = x_{old} + \frac{\delta_r + \delta_l}{2} cos(\theta_{old} + \frac{\delta_\alpha}{2}) \tag{2.12}$$

$$y_c = y_{old} + \frac{\delta_r + \delta_l}{2} sin(\theta_{old} + \frac{\delta_\alpha}{2}) \tag{2.13}$$

$$\theta_c = \theta_{old} + \delta_\alpha \tag{2.14}$$

$$v_c = \frac{\delta_r + \delta_l}{2T} \tag{2.15}$$

From the formula above, $\delta_\alpha$ is the direction of the mobile robot that has changed for T seconds from the sample time, $\delta_l$ and $\delta_r$ are the driving distances that each wheels moves for T seconds. If the experiment happens in fast enough samples, it can be said that we are able to get a value that is somewhat valid, although it is not totally accurate.

$$\delta_\alpha = \frac{\delta_r - \delta_l}{W} \tag{2.16}$$

$$2\pi R : \delta_r = PG : z_r \Rightarrow \delta_r = 2\pi R \frac{z_r}{PG} \tag{2.17}$$

$$2\pi R : \delta_l = PG : z_l \Rightarrow \delta_l = 2\pi R \frac{z_l}{PG} \tag{2.18}$$

$z_r$, $z_l$ : Amount of the Encoder which has changed for T seconds

P : Encoder's purse number per one rotation(purse/rev)

G : Gear ratio of motor

**2.3 Input Data of The Neural Network**

**2.3.1 Ultrasonic and Infrared Sensor**

**1) Structure of the ultrasonic and infrared sensor**

It is important how to compose the sensors to get environmental data, which is required for the mobile robot's navigation, when the ultrasonic and infrared sensors are used. [Fig 2] is about the ultrasonic sensor structure studied so far: It shows an ultrasonic ring structure that uses 24 ultrasonic sensors, the structure with DOF which can rotate with a pair of radio sensors and the structure which has DOF in each radio sensor. In case of a second and third structure, it can abstract the characteristic feature of the objects and when needed, it can compose the form conveniently.

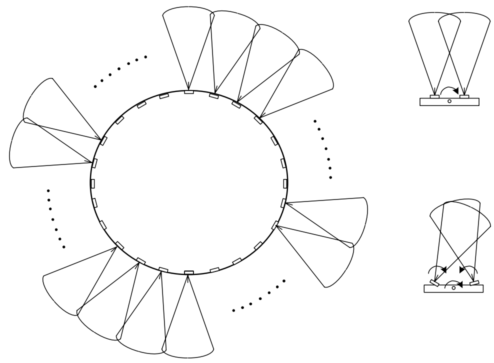

Fig 2 . Structure of the ultrasonic and infrared sensor [7]

For the purposes of reaching its destination promptly and creating a path, we use a sonar ring.

We arranged 8 sensors in front of the robot by making pairs of each ultrasonic and infrared sensor, and used the existence of objects about each sector using VFH (Vector Field Histogram) as the input value of neural network.

**2) VHF method**

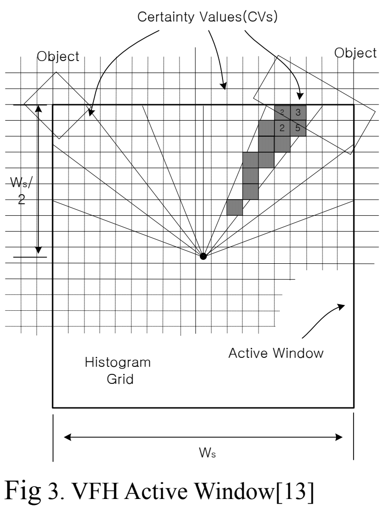


Fig 3. VFH Active Window[13]

VFH method decides the direction of study by calculating whether some objects exist or not by several stages.

At the first stage, it saves the sensor value that is input per each sampling in the two dimensional memory in terms of histogram grid. At this stage, each memory grid saves the certainty value of obstacles and calculates the polar histogram about all grids belonged to the active window. At the second stage, the each area that divides the whole active areas by standard angle ($\alpha^o$) is called as sector. The polar obstacle density ($h_k$) is described as the value that adds all rates of obstacle existence from the grids belonged to each sector. Finally, the average value of polar obstacle density (POD) is calculated, and this becomes the smoothed polar obstacle density (SPOD) and it is used to decide whether obstacles exist or not by thresholding this value. The course above could be described by numerical formula as follows. The definitions of the variables are as follows.

$a,b$ : Positive constants

$c^*_{ij}$ : Certainty value of active cell (i,j)

$d_{ij}$ : Distance between active cell (i,j) and the VCP

$m_{ij}$ : Magnitude of the obstacle vector at cell (i,j)

$x_c, y_c$ : Present coordinates of the VCP

$x_i, y_j$ : Coordinates of active cell (i,j)

$\beta_{ij}$ : Direction from active cell (i,j) to the VCP

First of all, the direction from VCP to grid could be calculated like this.

$$\beta_{ij} = a\tan 2(y_i - y_c, x_i - x_c) \tag{2.19}$$

The volume of obstacle vector for each grid is calculated like this.

$$m_{ij} = (c^{*2}_{ij}(a - bd_{ij})) \tag{2.20}$$

The total direction sectors are as follows.

$$k = INT\ (\beta_{ij} / \alpha) \tag{2.21}$$

The POD for each sector $k$ can be calculated as follows.

$$h_k = \Sigma_{ij}\, m_{ij} \tag{2.22}$$

In this stage, $m_{ij}$ is the value considering the object's distribution according to the distance from the VOD, so it should be treated with lower value about the grid locating at the end of the active window. Therefore, the invariable value $a$ and $b$ could be decided to the appropriate value to satisfy the following formula.

$$a - bd_{\max} = 0,$$

where $$d_{\max} = \sqrt{2}(\omega_s - 1)/2 \tag{2.23}$$

In the above formula, $d_{\max}$ is the maximum value of the distance from the VCP to the end. Therefore, $m_{ij}$ of the grid that the distance from the central is $d_{\max}$ is 0. And the nearer to the center point, the value of $m_{ij}$ would be bigger linearly. To describe the numerical formula of the course of calculating Smoothed Polar Obstacle Density ($\dot{h}_k$) is as follows.

$$\dot{h}_k = \frac{h_{k-l} + 2h_{k-l+1} + \cdots + lh_k + \cdots + 2h_{k+l-1}h_{k+1}}{2l+1} \tag{2.24}$$

Whether obstacle exists or not is decided by thresholding the calculated value of SPOD with appropriate critical value.

### 2.3.2 Motion Detector

For motion detectors, we use SH-906 from Shiho company to distinguish whether the obstacle is human or not. 2 motion detectors are arranged in front of the robot at 80-degrees each. Motion detectors sense the surroundings and tells whether a human is near, which is then used as the input data of the neural network.

## 3. COLLISION AVOIDANCE METHOD

When the mobile robot drifts towards the target position, it will encounter obstacles. Also, the obstacle might not be human and thus it could be number more than 1. Therefore, the collision avoidance algorithm will be applied differently depending on what kinds of obstacles or how many of them the mobile robot encounters. In the case where obstacles are close enough to bump into, then the mobile robot will avoid the collision using its neural network, and will move to its destination promptly by a linear path. Also, when there are more than 2 obstacles in its radius rage, the robot will avoid the nearest obstacle first since it has a greater chance to collide. Furthermore, if the obstacle is human, it gives more weight to that case and makes it react more sensitively. The collision radius we are discussing here is the maximum approximation to avoid a collision when the mobile robot encounters an obstacle in a nearby hypothetical radius.

### 3.1 Basic Ideas

Wall information expressions in the robot center coordination are separated from each sector by 22.5 degrees and compare each sector and analyze if the obstacle is human so each sector has values of two real numbers, either high (1) or low (0).

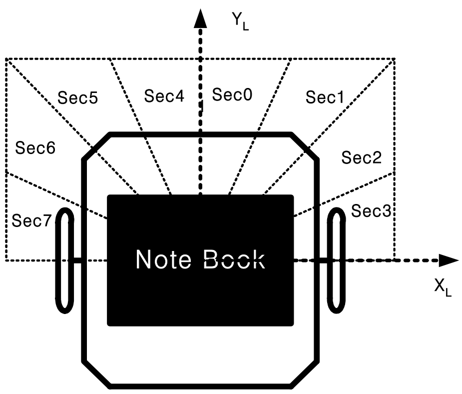


Fig 4. Wall_status of Direction on ego space coordinates

### 3.2 The Structure of The Neural Network

The neural network, dealt with in this paper, is a structure of the Multi layered backpropagation neural network.

[Fig 5] shows the structure of the neural network that is used in this paper. The input layers consist of 10 neurons, the hidden layers consist of 5 and the output layers consist of 8 neurons in this neural network.

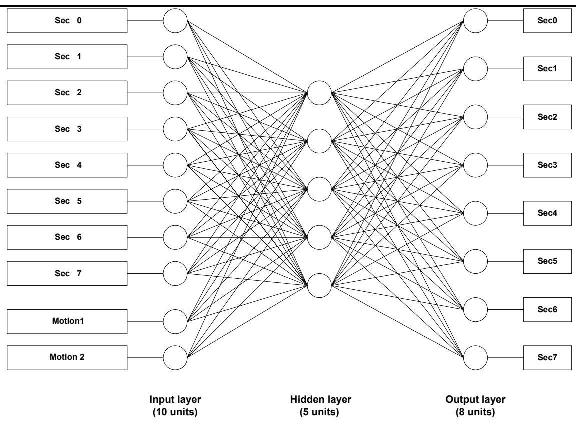


Fig 5. The Structure of the neural network

Input signals of the neural network make use of data from each sensor, which are applied in the mobile robot. And the output signals, comes from the results mentioned, apply fir robot's progress direction. [Table 1] shows the values of $\theta$ depends on the output data of neural network.

Table 1. $\theta$ depends on the output data of the neural network

| Sec0 | Sec1 | Sec2 | Sec3 | Sec4 | Sec5 | Sec6 | Sec7 | $\theta$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | -10 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | -20 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | -45 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | -90 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 10 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 20 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 45 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 90 |

### 3.3 Path Correct Signal

The mobile robot basically transfers to a short cut to get to its destination. However, it is inevitable that will correct its path when there are obstacles in its shortcut. When the mobile robot enter the collision radius, the angle of knowledge and conduct of the mobile robot is modified by the collision avoidance algorithm. In this research, we apply the output signals of the neural network as modifying signals to correct the progress degree of the path. Suggesting that the angle of the modified path called $\theta_{steer}$, the angle of the shortest path called $\theta_{Tar}$, and the angle of the path which passes the neural network $\theta$, then the angle of the modified progress direction path of the mobile robot as formula (3.1) may be determind.

$$\theta_{steer} = \theta_{Tar} + \theta \tag{3.1}$$

Following figure shows how to decide $\theta_{steer}$

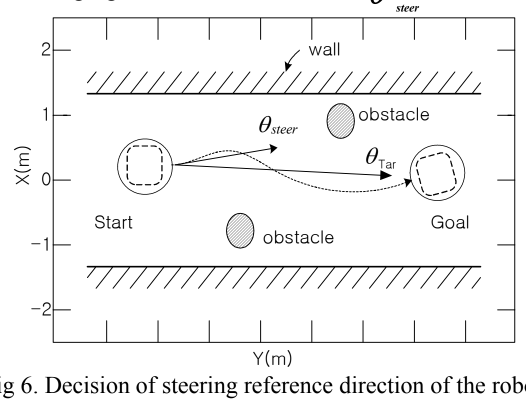


Fig 6. Decision of steering reference direction of the robot

### 3.4 The problem and solution of VFH method

VFH method can't always find the best suited way because it decides only the direction towards the destination avoiding objects using site-specific map. And all moving robot applied with VFH method don't always reach to the destined point. The reason is as follows: it is using site-specific map, so the case that robot can't escape could happen and there's no higher control function to respond to those cases. If it happens, as there's no way to recognize it and to escape from it, the robot would be turning around at one place always.

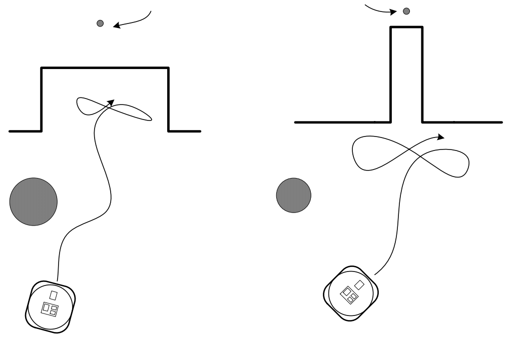

Fig 7. An example of Trap-situation

Therefore, Adaptive Goal Perturbation Method will be added in this paper, which is to travel preventing the case of falling in trap situation in the course of searching, and to escape from the trap situation by recognizing the situation on the basis of the self-centering coordinates of the robot. AGPM is consisted with the following three courses.

**1) Avoid-Trap-Mode**

This is the most basic traveling method. If a wall appear when seeing from the robot's coordinates system, in case that the gap between current direction and target direction is less than 90°, it moves by setting Virtual goal to go towards the nearest direction. If the gap between current direction and target is more than 90°, it gets out of Avoid-Trap-Mode.

**2) Escape-Trap-Mode**

This method is the method to escape from the trap situation that can't proceed any more with the basic traveling method that is described above. First of all, if it is decided to be in the trap situation now through the Check-Trap-Mode that will be described next, treat the courses as follows.

If the wall appears in front, firstly, the robot travels to escape the trap by creating virtual goal through Avoid-Trap-Mode. The robot would recognize the trap situation that returns to the repeated position through Trap Check Mode.

Whenever it checks the trap, the number of times of trap repetition is saved in the trap_count invariables. This number of times of trap repetition would decide the point to change the location of virtual goal. If the virtual goal is set to be further gradually, as the number of trap repetition is getting bigger, the robot in trap could escape from the trap.

**3) Check-Trap-Situation**

At this course, we would like to describe the method to recognize the trap situation. Firstly, robot would save the trap location through Avoid-Trap-Mode and set virtual goal. From this time, robot adds the direction error, if the added value of direction err is more than 150° and the distance from current location to the trap is less than $\varepsilon$, robot would recognize this situation as the trap situation.

# 4. POSITION AND ANGLE ACCUMULATION ERROR COMPENSATION METHOD

### 4.1 Position accumulation error compensation method

To compensate the accumulation error of the mobile robot indoors, when the transmitter's ultrasonic sensor sends ultrasonic waves, three triangular receiver's ultrasonic sensor on the ceiling detects the waves. Three receiver's ultrasonic sensors are located in triangular shape in random places.

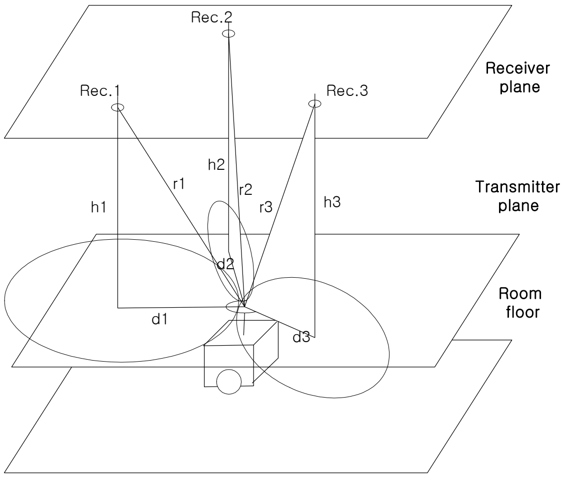


Fig 8. Planes of transmitter and receive[2]

The distance between the mobile robot and the receiver's ultrasonic sensor can be calculated as the following.

$$r_i = CS \times 34000 / CL \tag{4.1}$$

The Count clock number, 34000 comes from the ultrasonic wave's velocity 34000cm/s that originated from the processor while the ultrasonic waves transmitted as CountStep. CL means the Count clock frequency of the processor.

In [fig 8], Rec.1, Rec.2, Rec.3 mean the location of the receiver's ultrasonic sensors attached on the ceiling. Rec.i (i=1,2,3) means the total coordinate values and that it has a fixed position. h1, h2, h3 means the distance from the transmitter plane to the receiver plane.

When measured r1,r2,r3, it is possible to ascertain the distance of d1,d2,d3 by a trigonometrical function.

$$d = \sqrt{(r^2 - h^2)} \tag{4.2}$$

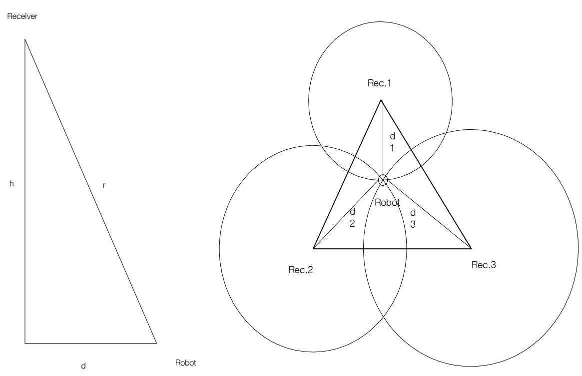


Fig 9. Receiver to Transmitter plane orthogonal projection

[Fig 9] is the picture of the projecting orthogonal to receiver plane of whole location of the mobile robot.

If the distance of d1, d2, d3 is known, then it is possible to find the absolute position of the mobile robot by the following formula.

$$(x_1 - x_r)^2 + (y_1 - y_r)^2 = d_1^2 \tag{4.3}$$

$$(x_2 - x_r)^2 + (y_2 - y_r)^2 = d_2^2 \tag{4.4}$$

$$(x_3 - x_r)^2 + (y_3 - y_r)^2 = d_3^2 \tag{4.5}$$

### 4.2 Angle accumulation error compensation method

MC (Magnetic Compass) used to compensate cumulative error about angle. Vector2X Electronic Compass Module is used. Atmega128 is used as the MC controller and data is transmitted by serial communication protocol of the Vector2X body itself. Vector2X uses 2 axis Compasses and Magnetic sensor modules, and it is relatively inexpensive, though the accuracy is low. MC is located right in the center of the robot's front body where it is less influenced by magnetic field caused by the motor.

# 5. SIMULATION AND EXPERIMENT

Collision avoidance algorithms, which were proposed in this paper, verify the possibility of application by simulation, and estimate its performance by applying it to a real system. Simulation is programmed using Visual C++.

### 5.1 Systems composition

We will explain the Robot system to be used in an experiment in this chapter. The robot platform is applied and its body has two drive wheels and two assistant wheels. It is necessary to read each sensor's cost through serial communication and to send the velocity and command to low rank controllers through FIFO communication in order to control this Robot platform. A Pentium 133MHz laptop computer is used to take charge of the real time controller. The environmental awareness department is composed of 8 ultrasonic and infrared sensors, and 2 motion detectors on each front. We built an indoor GPS by setting a transmitter ultrasonic sensor in the middle of the robot and setting 3 triangular-shaped receiver ultrasonic sensors on the ceiling. Visual C++6.0 is used as the compiler, sensor command and processed data is sent and received by 9600bps speed through a serial port. This kind of structure is often found on other robots. The figure of the system composition is as following.

Fig 10 . Robot system

### 5.2 Simulation

When we made the robot study the neural network 50,000 times, the error was 0.012, and the time elapsed was 15 seconds.

[Fig 11] shows the mobile robot avoiding a general obstacle and [Fig 12] shows it avoiding a human (obstacle 1) and a general obstacle (obstacle 2) at the same time. We can clearly see that the robot places more weight for a human and avoids more broadly than it would for a general obstacle when it encounters both human and general obstacle at the same time.

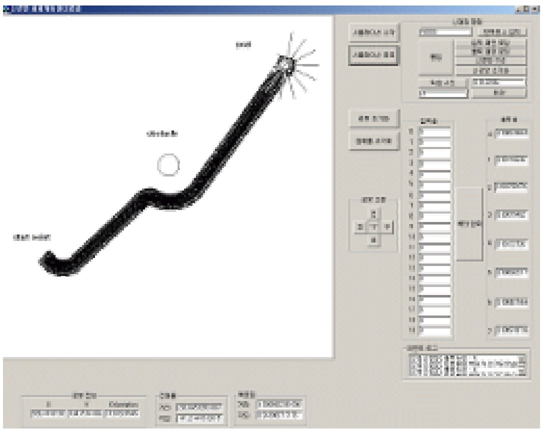

Fig 11. Collision avoidance result that applied the neural network

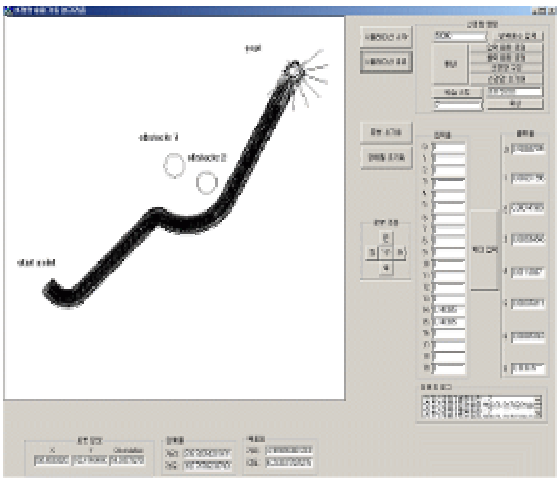

Fig 12. Collision avoidance result that applied the neural network

## 6. CONCLUSION

In this paper, we suggested an obstacle avoidance algorithm using the neural network for the mobile robot to transfer it to its destination safely in an unknown environment. In this paper, we used ultrasonic sensors, infrared sensors, motion detectors and a Magnetic Compass with input signals of the neural network. We used out signals of the neural network as the modifying signal for the mobile robot's progress direction in order to avoid collision. This paper shows that a collision avoidance algorithm by the neural network makes the mobile robot transfer to a shortcut as long as it doesn't meet an obstacle. However, when it comes to a possible collision with an obstacle, it modifies its path and arrives at its destination. We also found that the mobile robot can avoid an obstacle that it has not studied, by finding the closest output (answer). It was also verified by sample experiments applied in the real system that the mobile robot is able to reach its destination more correctly by compensating the cumulative error, which occurred from DR (Dead reckoning) by using indoor GPS.

## REFERENCES


[1] Moshe Kam, Xiaoxun Zhu and Paul Kalata, "Sensor Fusion for Mobile Robot Navigation" Proceedings of The IEEE, Vol. 85, NO. 1, pp. 108-119, January 1997

[2] Ghidary, S.S.; Tani, T.; Takamori, T.; Hattori, M., "A new home robot positioning system (HRPS) using IR switched multi ultrasonic sensors", Systems, Man, and Cybernetics, 1999. IEEE SMC '99 Conference Proceedings. 1999 IEEE International Conference on , Volume: 4 , pp. 737 -741, 1999

[3] Kyung-Soo Lim, "Obstacle Avoidance and Trap Recovery of Mobile Robot : Adaptive Goal Perturbation Method (AGPM)", Master's Thesis, 1999, Korea

[4] Kyung-Hun Hwang, "Dynamics Obstacle Avoidance using Fuzzy and AGPM for Trap Recovery of Mobile Robot", Master's Thesis, 2001, Korea

[5] Jin-Ho Choi , "A Study on Compensation Method of Cumulative Posture Errors for a Mobile Robot Using Multiple Sensors", Master's Thesis, 2002, Korea

[6] J.-H. Park, M.-C. Lee, "Moving Obstacle Collision Avoidance of a Mobile Robot Using Neural Network", Proceedings of The 12th KACC, pp. 1238-1241, October 1997, Korea

[7] E. G. Araujo and R.A. Grupen, "Feature Detection and Identification using a Sonar-Array", IEEE Proceding of International Conference on Robotics and Automation, pp. 1584-1589, May 1998.

[8] "Vector 2X and 2XG Electronic Compass Modules", Complete App. Notes

[9] Zhao Feng-Ji; Guo Hai-Jiao; Abe, K, "A mobile robot localization using ultrasonic sensors in indoor environment", Robot and Human Communication, 1997. RO-MAN '97. Proceedings., 6th IEEE International Workshop on , pp. 52 -57, 1997

[10] Ching-Chih Tsai, "A localization system of a mobile robot by fusing dead-reckoning and ultrasonic measurements" Instrumentation and Measurement Technology Conference, 1998. IMTC/98. Conference Proceedings. IEEE , Volume: 1 , pp. 144 -149, 1998

[11] http://moon.kjist.ac.kr/project/data_fusion/Data.htm

[12] Nakamura, T.; Morimoto, J.; Asada, R.; "Direct Coupling of Multisensor Information and Actions for Mobile Robot Behavior Acquisition", Multisensor Fusion and Integration for Intelligent Systems, 1996. IEEE/SICE/RSJ International Conference on, pp. 139-144, 8-11 Dec. 1996

[13] Johann Borenstein, Yoram Koren, "The Vector Field Histogram - Fast Obstacle Avoidance for Mobile Robots", IEEE Journal of Robotics and Automation, vol. 7, no. 3, June. 1991, pp 278-288.