한국공작기계학회지 Vol.18 No.6 2009. 12. pp. 664~674
Journal of the Korean Society of Machine Tool Engineers

Development of a New Navigation Technology for Mobile Robot Based on Sonar Sensors

Van-Quyet Nguyen*, Sung-Hyun Han+

(논문접수일 2009. 11. 02, 심사완료일 2009. 11. 29)

초음파센서 기반 이동로봇의 새로운 네비게이션 기술 개발

노연판쿠에트*, 한성현+

Abstract

This paper presents the theoretical development of a complete navigation problem of a non-holonomic mobile robot by using sonar sensors. To solve this problem, a new method to compute a fuzzy perception of the environment is presented, dealing with the uncertainties and imprecision from the sensory system and taking into account nonholonomic constraints of the robot. Fuzzy perception, fuzzy controller are applied, both in the design of each reactive behavior and solving the problem of behavior combination, to implement a fuzzy behavior-based control architecture. Different experiments in populated environments have proved to be very successful. Our method is able to guide the mobile robot named KUM-Robo safety and efficiently during long experimental time.

Key Words : Robot Navigation(로봇 항법), Ultrasonic Sensor(초음파센서), Fuzzy Controller(퍼지제어기), Non-holonomic Mobile Robot(논-홀로노믹 이동로봇)

1. Introduction

Navigation is the process of determining and maintaining a path or trajectory to a goal destination. We have seen that animals possess a variety of remarkable abilities to assist them in navigation. To provide robot with many of the same abilities, we use sensors such as: computer vision, a clock, a map of terrain, wheel encoder, range finder, gyroscopes, etc.(1). Autonomous mobile robots are required to navigate in more complex domains, where the environment is uncertain and dynamic. Autonomous navigation in these environments demands adaptation and

* Dept. of Mechanical System Engineering, Graduate School, Kyungnam University
+ Corresponding author, Division of Mechanical System and Automation Engineering, Kyungnam University (shhan@kyungnam.ac.kr)
Address: 449 Wolyoung-dong, Masan, Gyeongsangnam-do, Korea, 631-701

perception capabilities. This paper describes improvements in the perception functions used in these kinds of robots. It should be noted that this is a nonholonomic robot with significant limitations in the reactive capabilities due to kinematic and dynamic constraints, and to the few number of sensors and large blind sectors in between them, making autonomous navigation a nontrivial task. The methods presented in this paper have been conceived to deal with these limitations of conventional robots.

In addition, fuzzy perception can be used straightforward to perform the control of the mobile robot by means of fuzzy behavior-based scheme already presented in literature(2,4,7,8) The main differences of the proposed approach with respect to other behavior based methods are (1) the nonholonomic constraints are directly taken into account in the behaviors. (2) The fuzzy perception itself can be used both in the design of each reactive behavior and to solve the problem of blending behaviors.

Hence, the fuzzy behavior-based control scheme presented in this paper allows not only implement reactive behaviors but also teleoperation and planned behaviors, improving system capabilities and its practical application. Furthermore, in these behaviors, soft computing techniques play an important role to solve different problems.

2. Fuzzy Perception

2.1 Mobile Robot

The following considerations are based on a mobile robot with the three degrees of freedom of planar movement, x, y and ϕ (Fig. 1). It is equipped with a ring of 12

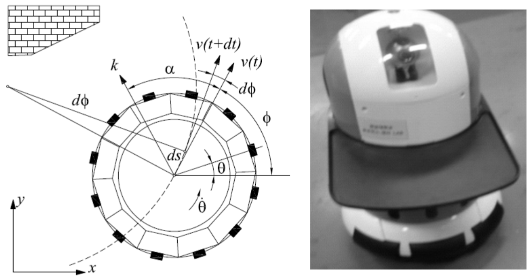


Fig. 1 Application of vector of general perception *k*

ultrasonic sensors which are able to perceive vertical or nearly vertical planes. The number of sensors is irrelevant as long as there are no blind sectors between them. θ refers to the orientation of this ring of sensors and not to the orientation of the robot itself, which is of no importance for the wall following algorithm. With ϕ indicating the direction of movement the kinematics model of such a robot is described as follows:

$$\begin{aligned} dx &= v\cos\phi dt \\ dy &= -v\sin\phi dt \\ d\theta &= \dot{\theta} dt \end{aligned} \tag{1}$$

Since there is no modeling of the environment the absolute position of the robot does not matter. So there is no world frame used here and the kinematics model can be expressed instead as:

$$\begin{aligned} ds &= vdt \\ d\phi &= \dot{\phi} dt \\ d\theta &= \dot{\theta} dt \end{aligned} \tag{2}$$

The speed v, the angular speeds $\dot{\phi}$ and $\dot{\theta}$ are used as control variables of the robot and generated by the fuzzy controller.

2.2 Concept of General Perception

It is a well known fact, that ultrasonic sensors have very poor directional resolution. Although these sensors very accurately determine the distance to the nearest object giving back an echo, this object can be anywhere under a certain angle to the sensor's axis. Moreover, this angle depends on the nature of the object's surface, the distance and the tilt of the surface with regard to the sensor's axis. That is why it would be difficult to try to first gain a representation of the immediate surroundings from the sensor data, i.e. to try to model objects or to determine the exact contours of a wall in order to control the robot accordingly. The concept of general perception avoids these difficulties because it does not undertake any kind of modeling of the environment.

For this purpose every ultrasonic sensor i of the mobile robot is assigned a perception vector k_i. Its direction equals the orientation of the sensor's axis and its length is a function of the distance d_i measured by this sensor:

$$k_i = \frac{d_{\max} - d_i}{d_{\max} - d_{\min}} \tag{3}$$

where $d_{\min}$ and $d_{\max}$ designate the shortest and longest distance respectively at which an object may be positioned to be reliably detected. k_i is limited to 0 and 1 respectively so that

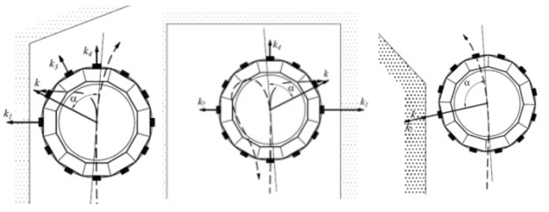
$$k_i = \begin{cases} 0 & \text{for } d_i > d_{\max} \\ 1 & \text{for } d_i < d_{\min} \end{cases} \tag{4}$$

The general perception vector k is composed of all individual perceptions k_i. Its direction equals the sum of the perceptions of all the sensors and its length equals the strongest individual perception:

$$k_i = k_{i,\max} \frac{\sum k_i}{\left|\sum k_i\right|} \tag{5}$$

The general perception's change in time is represented by $\dot{k}^*$ and expressed by a scalar. For this purpose the perception′s change in time of a sensor i is related to $\dot{k}_{\max} = v_{\max}/(d_{\max} - d_{\min})$ is the perception's change in time at head-on approach towards an obstacle at maximum speed $v_{\max}$. Moreover, only positive values are to be considered for $\dot{k}^*$, thus resulting in the relative perception's change in time of a sensor i as follows:

$$\dot{k}_i = \frac{dk_i}{dt} \approx -\frac{\Delta d_i}{\Delta t(d_{\max} - d_{\min})} \tag{6}$$



(a) (b) (c)

Fig. 2 Three typical situations of a wall following task of Mobile robot

$$\dot{k}_i = \begin{cases} \dfrac{\dot{k}_i}{\dot{k}_{\max}} = \dfrac{\Delta d_i}{\Delta t * v_{\max}} & \text{if } \Delta d_i < 0 \\ 0 & \text{for otherwise} \end{cases} \tag{7}$$

The maximum value of all sensors i thus arrived at, is the general perception's change in time $\dot{k}^*$:

$$\dot{k}^* = \dot{k}^*_{i,\max} \tag{8}$$

Figure 2 illustrates the concept of general perception. It shows a robot in three typical situations of a wall following mission. The general perception of the corner (Fig. 2a), the description of this situation in linguistic terms reads as follows: The general perception is very strong and relative to the path tangent it is to the left and somewhat ahead. The change of the perception is strongly positive if the robot moves at high speed. Another example of a standard situation is the dead end (Fig. 2b), which the robot enters moving along the left-hand wall. When the robot reaches the end of the dead end, the general perception starts moving further to the front. Its change in time can vary from small to very big depending on the robot's speed. In the case of the receding corner (Fig. 2c), for example, the general perception is strong and to the left back.

The concept of general perception is perfectly suited for such a linguistic description of a multitude of situations in which a mobile robot might find itself. A description of this kind is very simple, at the same time to the point, and is used in exactly this form as input for the rule base of the fuzzy logic wall following algorithm. In this context the orientation of the sensor ring is of subordinate importance. Neither is it necessary that several sensors perceive one and the same wall as was supposed in figure 2c for reasons of clarity. The concept of general perception enables a robot to follow a wall without difficulties, even if the wall is perceived by only one sensor.

One main aspect of the concept of general perception is the fact that it is composed of the perception of all sensors, hence the name general perception. Even those sensors, which from the point of view of the direction of movement are facing backwards and whose perception

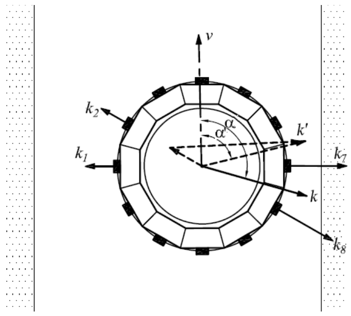


Fig. 3 Backward-direction of sensors' detection

might at first sight seem irrelevant, contribute to the general perception to the same degree. Using the example of a narrow corridor (Fig. 3) it can be shown that leaving out the backward-facing sensors would lead to a general perception characterizing the situation in a completely wrong way. The robot is moving down a very narrow corridor, the walls of which are perceived by two sensors 1 and 8.

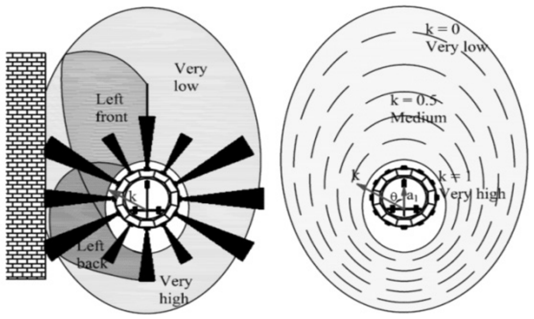
2.3 Fuzzy perception

Since a robot with nonholonomic constraints cannot move itself in any direction at every time instant, it is interesting to weight the different perceptions according with the direction where the obstacle was detected. In other words, an obstacle is less important if it is placed at a location that cannot be reached by the mobile robot, but it is more dangerous if it is on a reachable position. This task can be accomplished by considering the perception angle (θ_i) in the computation of the perception function

$$k_i = f(d_s, \theta_i) = sat_{0,1}\left(\frac{d_{\max}(\theta_i) - d_s}{d_{\max}(\theta_i) - d_{\min}(\theta_i)}\right) \tag{9}$$

where $sat_{0,1}(x)$ states for the saturation of x in the range [0, 1]. In this way, it is possible to assign different perceptions, i.e. different weights, to objects detected at the same distance relative to the mobile robot but at different directions. For example, perception function

$$k_i = f(d_s, \theta_i) = sat_{0,1}\left(\frac{nd_m(1-\varepsilon) - d_s(1-\varepsilon\cos\theta_i)}{(n-1)d_m(1-\varepsilon)}\right) \tag{10}$$



(a) (b)

Fig. 4 Ellipsoidal perception area

is obtained by using the nonlinear function $d_{\min}(\theta_i) = \frac{d_m(1-\varepsilon)}{(1-\varepsilon\cos\theta_i)}$, and $d_{\max} = nd_{\min}(\theta_i)$ (with n>1), in Eq. (9). Thus, an object detected in front of the robot, i.e. with a perception angle $\theta_i = 0$, at the minimum distance d_m, will get the highest perception length (k = 1), while one detected at the maximum distance (nd_m) will get the lowest perception. Therefore, an ellipsoidal perception area, as shown in Fig. 4b, is obtained, resulting in the curves of constant perception being ellipses of eccentricity ($0 < \varepsilon < 1$). Furthermore, it can be shown that perception function (10) results in a better reactive navigation performance and a more robust and stable perception-based control .

Perception vector can be considered by means of fuzzy logic using linguistic terms to describe perception angle and length as shown in Fig. 4a and b, respectively. Thus, a fuzzy description of the environment is obtained. It is important to recall that the perception vector does not intend to be a representation of the environment, rather than a fuzzy description of it.

Furthermore, it is interesting to stress that the perception vector implies a fuzzy high level description of the environment, being independent of the type of range sensor used. So, it is possible to use different perception functions from Eq. (9) for each kind of sensor (laser, ultrasonic, infrared). Thus, sensor data fusion can be reduced to compute different vectors from the sensor measurements and to combine them to obtain the perception vector.

However, if a full perception of the surroundings is not available, the method still fails because the robot does not recognize the situation in which it is. This is illustrated in Fig. 12a by a real experiment, performing left wall following with simple general perception vector, with the robot being unable to fulfill its mission. Therefore, a technique, called virtual perception memory, is presented in the next section (Fig. 12b).

2.4 Virtual sensors

The main idea behind virtual perception memory is to take into account the previous perception to build the new perception vector. The perceptions, are updated as the robot moves on, and used as virtual sensors (see Fig. 5).

The previous perception can be updated as follows: consider a robot of arbitrary shape equipped with proximity sensors. Any such sensor may be located at a position *U*, with its axis pointing to the direction *s* (see Fig. 6). A frame *r* represents the robots position and orientation, *x* and θ, respectively, with respect to the world reference system w. The velocity *v* of the reference point and the angular velocity $\omega_{r/w} = \dot{\theta}$ of the robot with respect to the fixed frame w, give the state of motion. Furthermore,

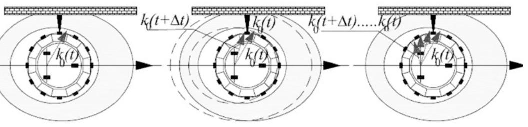


Fig. 5 Tracking by using virtual perception memory

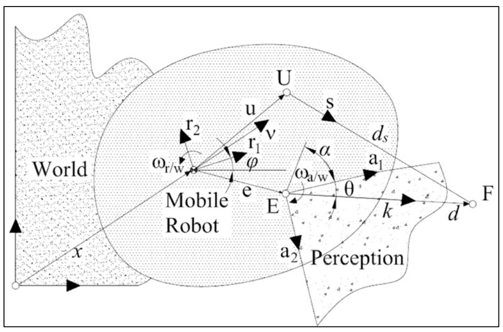


Fig. 6 Updated perception

the virtual perception coordinate system is assumed to be located at E, pointing to the direction of attention a₁. Then, an object detected by a proximity sensor at a distance d_s could be detected by a virtual sensor placed at E a distance *d*, and with an orientation θ with respect to the robot's direction of attention a₁.

Now the virtual perception will be updated taking into account the robots motion as follows: considering a perception function $k = f(d, \theta)$ and the corresponding inverse perception function, $d = g(k, \theta)$, and carrying out some calculations, it can be shown that the derivatives of angle and length of the perception vector are given by (assuming $g \neq 0$ and $\partial g/\partial k \neq 0$).

$$\dot{\theta} = \frac{1}{g}\{(\dot{x} + \omega_{r/w} \times e)[r_1 \cos(\alpha + \theta) - r_2 \cos(\alpha + \theta)]\} - \omega_{r/w} - \omega_{a/r} \tag{11}$$

$$\dot{k} = -\frac{1}{\partial g/\partial k}\left\{(\dot{x} + \omega_{r/w} \times e)[r_1 \cos(\alpha + \theta) - r_2 \cos(\alpha + \theta)] + \frac{\partial g}{\partial k}\dot{\theta}\right\} \tag{12}$$

where $\omega_{a/r} = \dot{\alpha}$ is the angular velocity of the virtual perception coordinate system relative to the robot. Integration of Eq. (12) yields the virtual perception. Thus, if the perception function is given by Eq. (11), then the inverse perception function is

$$g(k,\theta) = d_m \frac{1-\varepsilon}{1-\varepsilon\cos\theta}[n - k(n-1)] \tag{13}$$

So, looking in the direction of attention a₁ ($\theta = 0$); $k = 1$ corresponds to an object at a distance d_m to the perception reference system E, while $k = 0$ is mapped to a distance nd_m. Partial derivatives of $g(k, \theta)$, required to compute derivatives of angle and length of the perception vector in Eq. (12), are

$$\frac{\partial g}{\partial k} = d_m \frac{1-\varepsilon}{1-\varepsilon\cos\theta}(1-n) \tag{14}$$

$$\frac{\partial g}{\partial \theta} = d_m \frac{\varepsilon(1-\varepsilon)}{(1-\varepsilon\cos\theta)^2}[(n-1)k - n]\sin\theta \tag{15}$$

Inverse perception function (14) is a valid one since it fulfills the assumptions $g \neq 0$ and $\partial g / \partial k \neq 0$.

Fig. 12b shows the same real experiment as that of Fig. 12a, but now using the virtual perception memory. Thus, the robot also loses perception of the obstacle after starting to turn right. However, this time it tracks the perception it had before and the controller acts as if the robot continuously perceived the obstacle.

2.5 Obstacle Avoidance Behaviors

The wall following method described is not only useful to execute an explicit instruction such as “follow that wall”. It’s also used to avoid an unexpected obstacle in a predefined movement or mission. While the robot is moving, unexpected obstacles or walls can appear and avoiding them is desired and then continues executing the rest of the plan. Taking all that into account the problem of the obstacle avoidance could be reduced to three main aspects presented detail below.

2.5.1 Start to avoid an unexpected obstacle

This part has been simplified to the robot by the planner. The planner makes the calculations to obtain the minimum distance between each particular movement in the known environment. The avoidance begins when one sensor detects an object nearer than the distance given by the planner.

2.5.2 How to avoid the obstacle

The avoidance of the obstacle consists of following the contour of the obstacle in the same way that has been explained before. The maximum speed of the following process will be the speed of the element movement (EM) that was in execution when the obstacle has been detected. That speed has been calculated as the maximum safe speed in the region of the environment by the planner.

2.5.3 Finish the avoidance of the obstacle

That part of the avoidance is the most complex part because of the multiple possibilities of movements and reasons for the finishing.

The avoidance can finish:

a) When the robot gets back to one of the EMs of the plan. (Main case).

b) When a long time has elapsed from the beginning of the avoidance. (The obstacle covers all the rest of mission).

c) If the robot is very far from the point of the beginning of the avoidance. (The robot could go very far from its goal in the mission).

The cases (b) and (c) are easy to detect but the case (a) depends on the types of the movements of the robot in the mission. It’s important to know that all of the calculations to detect the end of the avoidance have to be made as fast as possible to get the maximum time free in the CPU for the rest of processes (Position control, radio communications, avoidance, etc.). Then all of the types of movements possible are reduced to segments of lines and circumference’s arcs.

3. Design Fuzzy Controller Based on Perception Vector

Perception vector can be considered by means of fuzzy logic yielding a fuzzy description of the environment. This description of the environment can be easily used as input to a fuzzy controller to perform reactive navigation. Furthermore, it is also possible to compute different perception vectors from the virtual perception memory (covering different areas around the mobile robot: left, right and front sides, for instance), and to use them to implement fuzzy controllers or behaviors which perform specific tasks taking into account nonholonomic constraints. The combination of the different behaviors, in a cooperative scheme, can be also easily done by means of fuzzy logic. In the following, a detailed description of the perception based fuzzy control system is performed, including implementation and combination of behaviors.

3.1 Application of the perception vector in designing control scheme’s behavior

Each behavior is defined by a set of fuzzy rules supplying control commands to the mobile robot. The rules have a fuzzy description of the environment obtained from the virtual memory perception as antecedents, and the control commands to be applied to the mobile robot, such as steering angle and linear velocity, as consequents. The implemented reactive behaviors are the following ones: left wall following, right wall following, obstacle avoidance, turn around and corridor following.

- In case of **left wall following**, for instance, the necessary input is the perception to the left side of the mobile robot. Then, a left hand perception vector is built based on the perceptions stored in the virtual perception memory, yielding a left perception angle (θ_l), together with a left perception length (k_l). The goal of this behavior is to keep the left perception vector at a medium value and pointing to left center. The left wall following fuzzy logic controller is a conventional Mamdani one composed of fuzzy rules such as:

  If (θ_l is LF) and (k_l is H) then (steer is HR)

Table 1 shows the whole rule base controlling the robots direction of motion by means of the steering angle, where θ_l, for example, is described by the linguistic terms LF (Left Front), LC (Left Center), and LB (Left Back). Fig. 8 shows the membership functions for the linguistic terms.

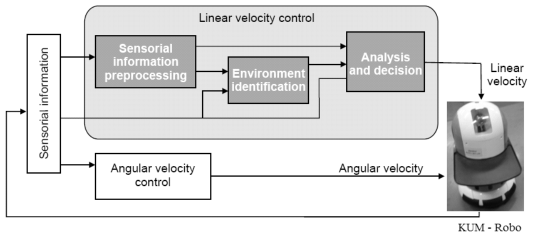


Fig. 7 Block diagram of control system

Table 1 Rule base controlling the robot direction of motion

θ_l / k_l	VL	L	M	H	VH
LF	C	C	R	HR	HR
LC	HL	L	C	R	HR
LR	HL	L	C	C	R

In the same way, the velocity command is obtained using rules like

If (θ_l is LF) and (k_l is L) then (velocity Medium);

- The other behaviors are defined in a similar way. For example, right wall following behavior is equivalent to the left one, but requires the computation of a right hand perception vector (θ_r and k_r). Corridor tracking behavior is straightforward obtained from the combination of left and right wall following behaviors (this also shows how primitive behaviors can be used as building blocks to produce other, even more complex, behavior by means of a hierarchical combination). In order to implement the obstacle avoidance behavior in the forward motion of the robot, the left, right and frontal perception vectors are used. The frontal perception (θ_f and k_f) covers a particular sector in front of the mobile robot (yellow

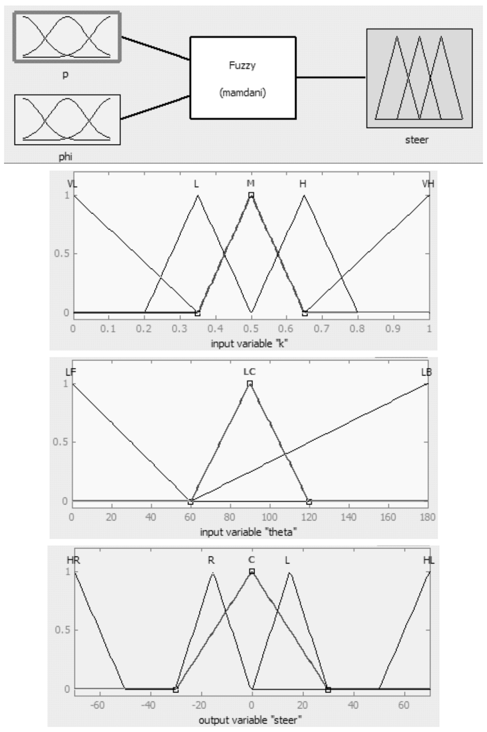


Fig. 8 θ_l, k_l and steer − membership functions

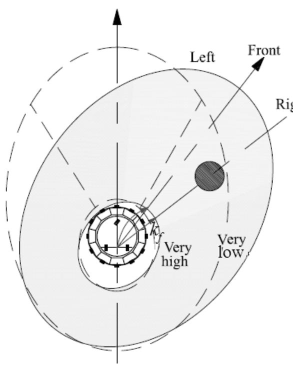


Fig. 9 Improving obstacle avoidance through direction of attention

area in Fig. 9), and there are similar fuzzy rules used like:

if (θ_f is R) and (k_f is VH) and (k_l is NOT VH) then (steer HL)

At this point, it is convenient to remark the effect of moving the direction of attention in the obstacle avoidance performance, making possible a faster and safer navigation. Consider, for example, the situation shown in Fig. 9, where the nonholonomic mobile robot approaches the obstacle by turning right, following the given trajectory due to its kinematics constraints. If the direction of attention is not considered, then the obstacle avoidance behavior will not practically react to the obstacle, because it is considered non-dangerous since it lays out of the frontal sector (see dashed ellipse), resulting in the sudden appearance of the obstacle in front of the robot after the turn. However, if the nonholonomic constraints are taken into account, i.e. direction of attention is kept parallel to the front heading wheel, then the obstacle avoidance behavior could react early, giving a smooth trajectory avoiding the collision.

- Turn around behavior is used to perform a turn of 180° when there is no feasible way in front of the robot (see Fig. 10). This may also imply a corridor that is too narrow. The aim of this behavior is to avoid guiding the robot into a situation from which it could not recover without reverse motion due to its kinematic constraints.

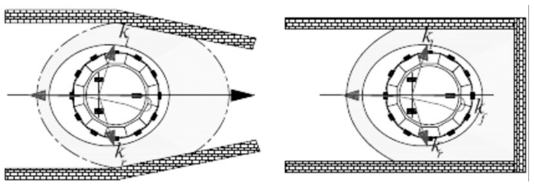


Fig. 10 An application of turn-around behavior

3.2 Behavior weight function

The combination of the previous behaviors is also made by means of fuzzy logic. Thus, working in parallel, each behavior produces its own control command vector, named c_i (steering and velocity). These vectors are fused to obtain a single command vector c by weighting the output of each behavior with a behavior weight (W_i) as

$$c = \frac{\sum_i (W_i c_i)}{\sum_i W_i}, \qquad i = lwf, rwf, oa, trn... \tag{16}$$

Behavior weights determine the relevance of each behavior at every time instant and they are calculated dynamically taking into account the situation the mobile robot is in. For example, the obstacle avoidance behavior weight (W_{oa}) increases as the obstacle comes closer. Therefore, the influence of a behavior on the mobile robots action is situation dependent. Several approaches have been proposed in the literature which could be used to compute the behavior weights. Some of these methods require computing a set of specific variables, or context rules, describing situations where the corresponding behavior should be activated. Nevertheless, in this paper, the behavior weights can be obtained directly from the information provided by the different perception vectors.

For example, in the case of left wall following behavior, the behavior weight has to be high if a wall is detected at the left side and at a medium distance (i.e., when the left perception vector is pointing at *LeftCenter* with a perception length around 0.5). On the contrary, if a wall is detected at the left front and very near, it would not be a good solution to perform wall following but avoiding collision. The left wall following behavior weight (W_{lwf}) must be low in this case. This can be computed as follows

$$W_{lwf} = sat_{0,1}\left(1-\left|\frac{\theta_l - LC}{2}\right|\right)sat_{0,1}(2k_l) \tag{17}$$

where *LeftCenter* corresponds to a perception angle θ_l with the highest membership degree to the linguistic term Left Center (θ_l: $LC(\theta_l)$ = 1). The first factor in Eq. (17) takes into account the relative orientation error of the robot with respect to the wall. This factor takes 1 when the wall is detected with a perception angle of LeftCenter, decreasing progressively as the error grows. The second factor computes how far away the robot is from the wall. It is 1 if the wall is detected with a perception length of 0.5 or closer. When this happens, the final W_{lwf} depends fully on the relative orientation error.

On the other hand, the weight of the obstacle avoidance must be higher as the obstacle is closer. Therefore, it follows $W_{oa} = sat_{0,1}(1-|\theta_f/2|)sat_{0,1}(2k_f)$. Furthermore, it is possible to define priorities between behaviors. This can be done by taking into account the weight of one behavior to compute the weight of the remaining. For example, if W_{lwf} and W_{oa} get a high value at the same time, the priority of the left wall following behavior will decrease as $W_{lwf} = W_{lwf}(1-W_{oa})$.

4. Experimental results

This section presents some experimental results of the proposed methods to the non-holonomic mobile robot KUM-Robo. The robot carries on-board a heterogeneous configuration of ultrasonic sensors. It is presented two kinds of experiment including general perception and application of fuzzy perception. All the experiments have been implemented in the KUM-Robo with on-board 486 DX2 66MHz, PC computer with a real time Windows XP professionals operative system. All real experiments are recorded as below figures.

4.1 General perception Conception

Figure 11 shows a tour along walls and around corner so that the robot simultaneously perceives the two walls on either side the robot encounters an obstacle. If the distance between the wall and the obstacle is too small for the robot to pass through, the obstacle is seen as part of the wall and circumvented (Situation A). If the distance is big enough, the robot passes the obstacle with a slight deviation from its path (Situation C). The obstacle in position B leaves a relatively large gap which, nevertheless, is too small for passing through. In this situation the robot loses orientation.

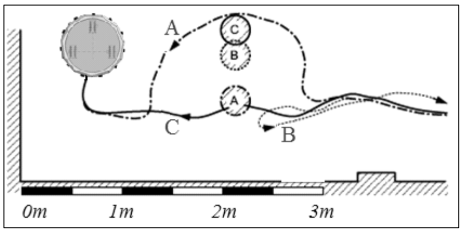
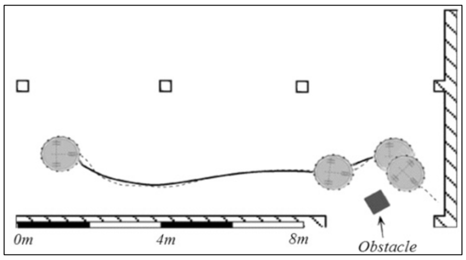
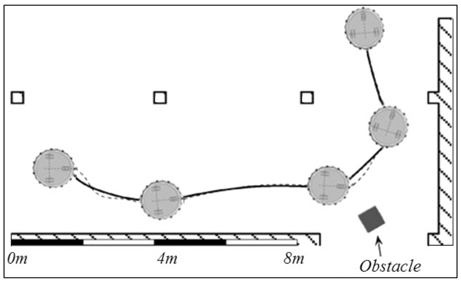


Fig. 11 An obstacle at three different distances from the wall



(a) Loss of perception while turning right



(b) With perception memory

Fig. 12 The differences result of perception's type

Figure 12 shows an experiment with the same conditions but it was applied with differential perceptions.

4.2 Fuzzy perception

In this Instead of a typical ring of identical sonars, there are 12 sonars of three different types, placed at different locations (see Fig. 4a). Six of them are large-range sensors (0.6–3.0 m), two are mid-range (0.2–1.0 m), and the other four are of short-range (0.06–0.3 m). Furthermore, these ultrasonic sensors use a higher frequency and have a narrower sonar beam than the commonly used sonars in these kinds of applications. The sensors are arranged in a way that three of them cover the front part of the robot and the other six cover its lateral sides, and the other three cover its back part.

In these experiments, the virtual perception system was placed at the center of the robot's rear axle and the direction of attention a_l was kept parallel to the front wheel. The perception function used is given by Eq. (10), with $n = 2$; $d_m = 1.4$, and $\varepsilon = 0.6$.

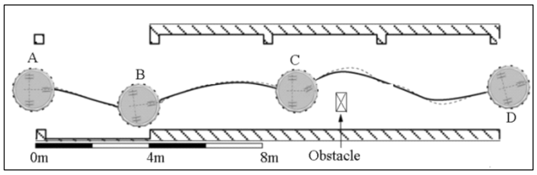


Fig. 13 Reactive behavior navigation of KUM-Robo

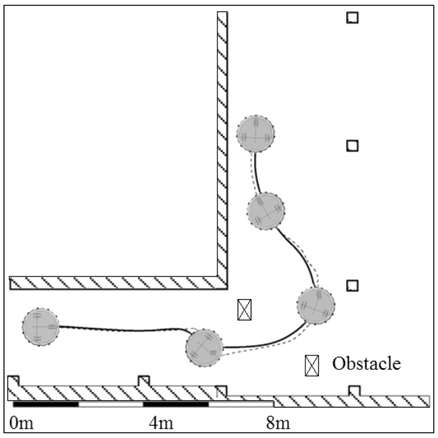


Fig. 14 2 – Obstacle avoidance behaviors

Experiments result is shown in Fig. 13 where the robot has to navigate through a corridor which is partially obstructed by an obstacle. The robot starts at point A with corridor tracking behavior, since it has equal perception at both sides. As the robot moves on it detects free space to its left and changes its behavior smoothly to follow right wall. When entering the corridor it tries again to center itself in the corridor B until it encounters the obstacle at C and the obstacle avoidance behavior becomes dominant (note that the obstacle is detected just in the front of the robot, and the collision risk is higher due to the non-holonomic constraints). The corridor is wide enough, i.e. the perception at both sides is sufficiently low that the turn- around behavior is not activated, so the robot tries to round the obstacle and, indeed, detects the passage between the obstacle and the wall. From this point on the robot is again guided mainly by the corridor tracking behavior until D.

Finally, figure 14 shows the mobile robot in an autonomous behavior experiment avoiding two obstacles doing a right turn, to later on keep following a right wall. Note again the difficulties performing this experiment due to

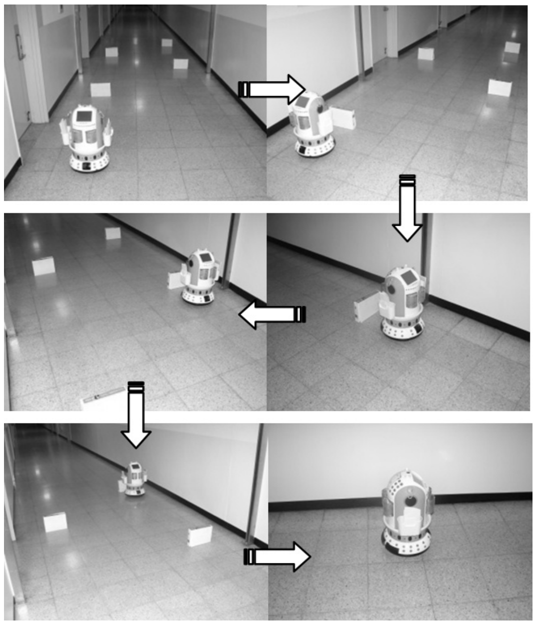


Fig. 15 Real experimental environment

the features of the robot and the environment involved, making navigation a nontrivial task.

Fig. 15 presents the real experimental environment of proposal. Our mobile robot was performed good navigation in real environment with wall following behavior and obstacle avoidance etc.

5. Conclusions

This work describes the design and real implementation of wall following and fuzzy perception concept with a non-holonomic mobile robot named KUM-Robo. The techniques to obtain a fuzzy perception of the environment in the design of each reactive behavior and solving the problem of behavior combination, to implement a fuzzy behavior based control architecture. It should be remarked that, at difference with other behavior based approaches, in the proposed technique the nonholonomic constraints are considered in the design of each behavior.

Furthermore, in order to improve the capabilities of the intelligent control system and its practical applicability, teleoperation and planned behaviors, together with their combination with reactive ones, have been considered. Experimental results, of an application to control the KUM-Robo autonomous robot, demonstrate the robustness of the proposed method.

References

(1) Bekey, G. A., 2005, *Localization, Navigation, and Mapping*, *Autonomous Robots: From Biological Inspiration to Implementation and Control*, The MIT press, Cambridge, Massachusetts, London, England, pp. 473~507.

(2) Baturone, I., Moreno-Velo, F. J., Sanchez-Solano, S., Ollero, A., 2004, "Automatic design of fuzzy controllers for car-like autonomous robots", *Fuzzy Systems, IEEE Trans.*, Vol. 12, pp. 447~465.

(3) Luo, R. C. and Chen, T.M., 2000, "Autonomous mobile target tracking system based on grey-fuzzy control algorithm," *IEEE Trans. Ind. Electron.,* Vol. 47, pp. 920~931.

(4) GasSos, J. and Rosetti, A., 1999, "Uncertainty representation for mobile robots: perception, modeling and navigation in unknown environments," *J. of Fuzzy Sets and Systems.,* Vol. 107, pp. 1~24.

(5) Braunstingl, R., 1996, "Improving reactive navigation using perception tracking", *Robotics and Manufacturing, Recent Trends in Research and Applications*, NewYork, Vol. 6, pp. 25~30

(6) Leonard J. and Durrant-Whyte, H. F., 1992, *Directed Sonar Sensing For Mobile Robot Navigation*, Kluwer Academic Publishers, Boston.

(7) GarcSTa-Cerezo, Ollero. A., MartSTnez, J. L., and Mandow, A., 1997, *Fuzzy tracking methods for mobile robots, in: Applications of Fuzzy Logic: Towards High Machine Intelligence Quotient*, Prentice-Hall, NJ, Chap. 17.

(8) SaGotti, 1997, "Fuzzy logic in autonomous robot navigation: a case study," *Soft Comput.,* Vol. 1, No. 4, pp. 180~197.

(9) Crowley, J. L., 1989, "*World Modeling and Position Estimation for a Mobile Robot Using Ultrasonic Ranging*," Proceedings of the 1989 IEEE International Conference of Robotics and Automation, Scottsdale, Arizona, pp. 674~680.

(10) Borenstein, J., Wehe, D., Feng, L.. and Koren, Y., 1995, "*Mobile Robot Navigation in Narrow Aisles with Ultrasonic Sensors*," ANS 6th Topical Meeting on Robotics and Remote Systems, California., pp. 1~9.