# Real-Time System Design and Point-to-Point Path Tracking for Real-Time Mobile Robot

F.H. Wang *
Department of Electrical & Computer Engineering
National University of Singapore,
Singapore 119260

June 8, 2003

**Abstract:** In this paper, a novel feasible real-time system was researched for a differential driven wheeled autonomous mobile robot so that the mobile robot can move in a smooth, safe and elegant way. Least Square Minimum Path Planning was well used for the system to generate a smooth executable path for the mobile robot, and the point-to-point tracking algorithm was presented as well as its application in arbitrary path tracking. In order to make sure the robot can run elegantly and safely, trapezoidal speed was integrated into the point-to-point path tracking algorithm. The application to guest following for the autonomous mobile robot shows its wide application of the algorithm. The novel design was successfully proved to be feasible by our experiments on our mobile robot Interactive Robot Usher (IRU) in National University of Singapore.

**Key words:** Path Tracking, Point-to-point tracking, Trajectory generating, Least square algorithm.

## 1 Introduction

In recent years, many researcher have focused in constructing a novel and flexible machines to perform some of the most risky, monotonous task instead of human beings. In these mobile robot, there are legged-based mobile and wheeled-based robots. Although, some researchers have succeeded in the former mobile robots, in our system design, the wheel-based mechanism was employed, because the wheel-based mechanism has strong adaptability to human-made flat surface, where our mobile robot IRU will work on. In addition, real-time system was employed to interact with environments swiftly. Our real-time system design takes the advantage of high-speed computer and DSP, the communication is solved by the parallel port.
For the wheel-based system, a lot of path generating algorithm and path tracking approach was used for the system. In [4], [5] and in [7], fuzzy logic path tracking algorithm is introduced. It sounds a good idea and has been proved to be an effective way to realize the path tracking. However, it is difficult to adjust the parameters in fuzzy logic rule, especially, when there are so many factors effect the system, the problem became more complicated. Therefore, the algorithm was greatly discounted. In our design, Least Square Minimum path generating algorithm is introduced to generate a smooth way for the robot to follow, and then, point-to-point tracking is developed to swiftly track the generated path. Further, trapezoidal velocity is introduced so as to make the robot run smoothly and safely. In order to show the wide application of the provided algorithm, guest following by point-to-point tracking is briefed. In our experiment, the trapezoidal point-to-point algorithm was proved to work efficiently and the tracking is smooth and elegant, which is uniform with our design.
In Section 2, a feasible real-time system design was introduced. Then, in Section 3, Least Square Minimum Path Planning was well developed for the system to provide the smooth executable path for the robot. And then, the point-to-point tracking algorithm was presented in detail together with its application in the path tracking. To improve the performance of the algorithm, trapezoidal speed was integrated into the algorithm. At last in this section, guest following by point-to-point algorithm illustrates its wide application of the algorithm in Section 4. Finally, a brief conclusion is given in the last Section.

## 2 Real-time system design

The differential driven robot consists of two independent driven wheels and two omni-directional non-driven castor wheels. The kinematic motion of the wheel is

*Corresponding author: Tel: (+65) 6874 5252; E-mail: engp2425@nus.edu.sg.

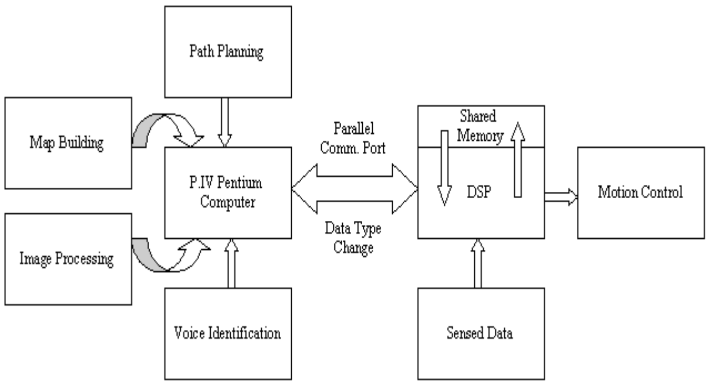


Figure 1: Real Time Computer-DSP Processor System Architecture

considered as pure rolling and non-slipping wheels. The two electrical motors of the driven wheels are controlled in a way that the left wheel velocity of the robot $v_l$ and the right wheel velocity of the robot $v_r$ can be considered as the control variable of the system. Then, by controlling the two independently velocities of the robot wheels, three configuration variables, namely two Cartesian coordinates characterizing the position of the mobile robot, and its orientation.

The system is a real-time system, self position estimation at any place depends on the self-position estimation. Because the model of the vehicle used in our project is based on differential-driven wheels. The dynamic model of the vehicle motion is described by the following equations.

$$\dot{x}(t) = v(t)\cos\theta(t) \quad (1)$$
$$\dot{y}(t) = v(t)\sin\theta(t) \quad (2)$$
$$\dot{\theta}(t) = \omega(t) \quad (3)$$

The key feature of our IRU operation system is its real-time operation system. This implies that the three levels including decision making level, sensor processing level and motion control level operate in parallel and are loosely coupled to each other. Naturally, a multi-processor computer architecture or a multi-computer architecture is required to process the different function levels since it is difficult for one processor to synchronously carry out all the robot's tasks, including image processing, map building, path planning and motion control.

An novel system structure, computer-DSP structure was developed for the real-time system. A high speed P.IV (1.6 GHz) computer works together with a digital signal processing (DSP) for the complex tasks and real-time operation of the interactive robot usher. The DSP processor is responsible for the fundamental motion control and some simple information processing, such as the information from the sonars and infrared. Also, the sensed data or feedback information will be sent to the DSP processor, so that the close-loop motion control system become a closed-loop system. The high level module are the most important part of the designed system. A large mount of information from image and voice system should be processed in this computer together with map building, path planning, decision making etc. After the high level processing, control command issued from the brain is transmitted down to DSP for executing. In the meanwhile, the motion information is transmitted up to the computer for the high level information processing such as map building.

The most difficult problem by employing this system is to overcome the communication problem between the computer and the DSP processor. In our design, a shared memory is accessible to both the computer and the DSP. The DSP can directly use the shared memory, while the computer can only access the shared memory by communicating with the DSP by the parallel port between the computer and the DSP processor. The structure of the real-time system can be shown in Fig. 1.

The advantage of our cooperation system is that the high speed computer have strong ability to handle the rich information from the high level module to map building and path planning etc., which is difficult task to the processors in the multi-processor system. While the DSP has the superiority of responding in a very swift and concise way, primarily concentrates on motion control, feedback information collection and obstacle avoidance. Based on the computer-DSP processor operation system, the IRU can efficiently handle the data in the high levels, and respond quickly in the low level.

## 3 Sequent Point Path Planning and Tracking

For a sequence of several points , generated by the brain of the robot where the decision is made and the directions are provided, the mobile robot should be able to generate smooth and feasible trajectory. In this paper, Least Square Minimum algorithm is introduced to draw a smooth way for the robot to follow. In order to let the mobile robot move along the pre-set path, trapezoidal point-to-point tracking algorithm is introduced , which functions as objective tracking from the original point to the objective point. The motion controller is to guide the robot to execute this nominal path and to track it with as minimum deviation as possible.

## 3.1 Least Square Minimum Path Generating

For a series of sequent points, a smooth path is expected to pass through all the given points. In order to get the corresponding path, a high order curve equation is introduced, whose coefficients are calculated by least square minimum algorithm. The order of the equation is decided by the number $n$ of the given points, and the order can be considered as $(n-1)$. For example, when the position of the robot is known, and another three sequent points are provided. Under the assumption that the sequence is along the forward direction, then, an equation with order three comes, represented by:

$$y(x) = a_3x^3 + a_2x^2 + a_1x_1 + a_0 \tag{4}$$

Here, $a_0, a_1, a_2, a_3$ are the pending coefficient. If there are $n$ forward sequent points, the coordinate of the original position and the coordinates of the corresponding objective points are known sequentially as $\{(x_0, y_0), (x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, considering $x$ as inputs, the corresponding $y$ as outputs, then, calculate the corresponding coefficients by least square minimum algorithm as:

$$\Phi = \begin{bmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ 1 & x_2 & x_2^2 & \dots & x_2^n \\ & \dots & & \dots & \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{bmatrix}^T \tag{5}$$

$$Y = [y_0,\ y_1,\ y_2,\ \dots,\ y_n]^T. \tag{6}$$

$$A = [a_0,\ a_1,\ a_2,\ \dots,\ a_n]^T. \tag{7}$$

Then, the corresponding coefficients can be calculated as:

$$A = (\Phi^T\Phi)^{-1}\Phi^T Y \tag{8}$$

For example, we know that the robot is at point $p_0(1,1)$ and the desired following three points are $p_1(2,0.5), p_2(3,2), p_3(4,1)$, which are the points the robot should follow in sequence. Then, according to the above analysis, a three order curve equation is needed to describe the corresponding curve.

$$\Phi = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 \\ 1 & 3 & 9 & 27 \\ 1 & 4 & 16 & 64 \end{bmatrix}^T \tag{9}$$

$$Y = [1,\ 0.5,\ 2,\ 1]^T. \tag{10}$$

$$A = [a_0,\ a_1,\ a_2,\ a_3]^T. \tag{11}$$

Then, according to the condition mentioned above, A is obtained:

$$A = (\Phi^T\Phi)^{-1}\Phi^T Y$$

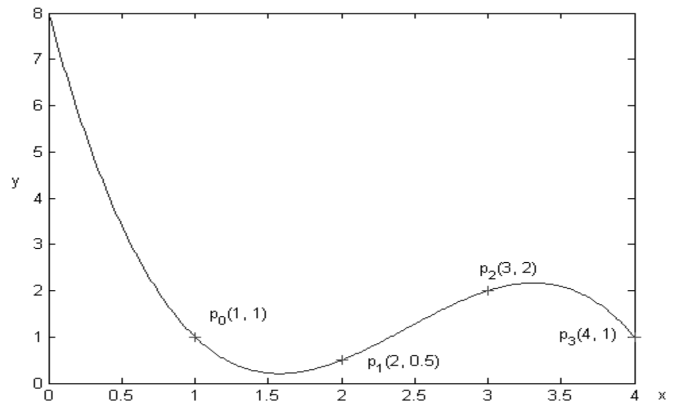


Figure 2: Least Square Minimum Path Scheduling to Follow Sequent Points

And therefore, $A$ is:

$$A = [8.0000 - 11.75005.5000 - 0.7500]^T$$

Then, the corresponding curve is obtained, which is the smooth path for the robot to follow.

$$Y = -0.75x^3 + 5.50x^2 - 11.75x + 8.00$$

Once the robot follows the path, it will go through the appointed three points smoothly. The generated path and the desired following points are shown in Fig. 2.

## 3.2 Point-to-Point Tracking

The condition to apply the Point-to-Point path tracking is that the current position of the robot and the objective point are known. Let the distance from the objective point to the current point be $s_{object}$, which should be a positive scalar, and the direction from the objective position to the current position be $\theta_{object}$, then the corresponding distance between the two points and the robot head directional angle relative to objective point can be calculated as follows:

$$s_{object} = \sqrt{(x_d - x)^2 + (y_d - y)^2} \tag{12}$$

$$\theta_{object} = \arctan\frac{(y_d - y)}{(x_d - x)} - \theta_0 \tag{13}$$

Here, $\theta_0$ is the head directional angle of the robot w.r.t the coordinate and the angle $\theta_{object}$ should take value from $(-\pi, \pi]$.

Once the forward direction of the robot is against the objective point, very similar to the human, the IRU is able to adjust his forward direction to align with the objective point direction so that it will go a short path to reach the objective point. This is different from other algorithms, which just run in a round way to reach the

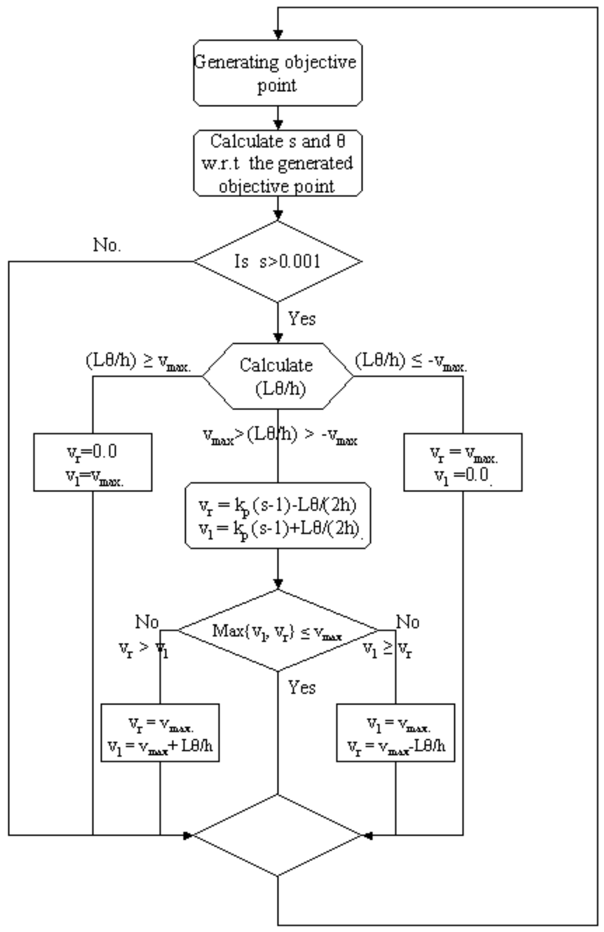

Generating objective point
Calculate s and θ w.r.t the generated objective point
Is s>0.001
No.
Yes
Calculate (Lθ/h)
(Lθ/h) ≥ v_max.
(Lθ/h) ≤ -v_max.
v_r=0.0 v_l=v_max.
v_r = v_max. v_l =0.0
v_max> (Lθ/h) > -v_max
v_r = k_p (s-1)-Lθ/(2h) v_l = k_p (s-1)+Lθ/(2h)
Max{v_l, v_r} ≤ v_max
No
v_r > v_l
No
v_l ≥ v_r
Yes
v_r = v_max. v_l = v_max+ Lθ/h
v_l = v_max. v_r = v_max-Lθ/h

Figure 3: point to point path tracking

appointed objective point. As the controllable variables in the real-time system are the two differential speeds, the command speed needed for the turn of the robot from its current direction to the desired direction is decided by the current angle deviation. Supposing that the sample space is $h$, $v_{max}$ is the maximum speed the wheel can move at and the minimum command speed is set at 0. The expected speed command can be calculated as:

$$v_d = L\theta_{object}/h \tag{14}$$

If the angle $\theta_{object}$ are too large, the maximum wheel speeds of the robot are unable to satisfy the expected speed for turn, then, it is obvious that the transformation speed of the robot is nougat. There are two cases:
(i)The deviation angle is positive, then set: $vd_r = 0.0$ and $vd_l = vd_{max}$
(ii)The deviation angle is negative, then set: $vd_r = vd_{max}$ and $vd_l = 0.0$
Once, the speed of the system is enough for turning, i.e. $-vd_{max} \leq L\theta_{object}/h \leq vd_{max}$ and the robot will move forward when turning. Under this condition, the command speed will be set as follows:

$$\begin{aligned} vd_r &= kps_{object} - L\theta_{object}/(2h) \qquad (15)\\ vd_l &= kps_{object} + L\theta_{object}/(2h) \qquad (16)\end{aligned}$$

Obviously, because of the distance between the robot and the objective point may be very large, the command speed is greater than the maximum command speed. Measure should be taken to prevent the over command speed while keep its turning command. Similar to the above, two cases may happens:
(i) The deviation angle is positive, then:

$$\begin{aligned} vd_l &= vd_{max} \qquad (17)\\ vd_r &= vd_{max} - L\theta_{object}/h \qquad (18)\end{aligned}$$

(ii) The deviation angle is negative, then:

$$\begin{aligned} vd_l &= vd_{max} + L\theta_{object}/h \qquad (19)\\ vd_r &= vd_{max} \qquad (20)\end{aligned}$$

From the formula above, it is obvious that once the direction of robot forward speed is aligned with the direction of the objective point, the robot will go straightly to the object. The Point-to-Point path tracking realization can be shown in Figure. 3.
As time goes, the robot will approach the objective point at the pre-set accuracy, i.e, $s \leq \delta$, $\delta$ is set as small distance error, the mobile robot will stop.

## 3.3 Point-to-Point Arbitrary Path Tracking

To track an pre-scheduled path by using the point-to-point tracking algorithm, the most critical problem is to set an appropriate point along the path as the objective point. The objective point will be generated by adding a fixed extra $\triangle$ to the current position coordinate either $x$ coordinate or $y$, which one to take depends on the future path change rate of the $x$ and $y$. If the $x$ change rate is larger, then take the $x$ as the independent axis and the $y$ can be calculated by the corresponding function of the appointed path, as result, the objective point on the path can be available for the robot to track. Otherwise, the $y$ coordinate will be set as the independent axis and $x$ can be obtained in the same way. The generated point is considered as "Objective Point" temporarily. Then, the path can be followed continuously step by step.
The value of the $\triangle$ is closely related with the smoothness of the path the mobile robot will run. The larger the value it takes, the smoother the path will be. As time goes, the sum of the current $x$ or $y$ coordinate and the extra $\triangle$ will be over the destination point. Then, the destination point will be considered the objective

point. In our experiments, the $\triangle$ takes the value $0.6m$, and the experiments proved the smooth motion of the mobile robot.

Once the objective point $(x_d, y_d)$ is generated, the point-to-point path tracking algorithm can be applied. The point-to-point algorithm makes sure that the objective point can be updated as time goes and the position of the mobile robot changes. Therefore, the smoothness of the tracking can be assured. Another important problem considered is the velocity saturation, in our point-to-point algorithm design, this problem is settled successfully by using the maximum possible speed instead of the excessive speed command when it needs more than the maximum speed it can provided. However, in our experiment on the robot, it is found the robot does not run elegantly, because of the large acceleration of the robot, to solve the problem, the Speed Trapezoidal was presented in the next subsection.

## 3.4 Trapezoidal Path Tracking

To find a universal trapezoidal velocity,consider limiting the maximum speed of tracking, which means that in order to smoothly increase the speed of the robot, the speed acceleration of the robot should be constrained to a limited value in the first several minutes. In our project, the maximum speed of robot is around $0.5m/s$, which is not so large speed, according to our experiment, 2 seconds is considered the most suitable acceleration period. After $2s$ acceleration, it is only possible for the robot to reach its maximum speed. As the time frame of the system is $1ms$, then it implies that after 2000 time space acceleration, it is possible for the robot to reach its maximum speed. The expression of the maximum speed within the $2s$ is as follows:

$$vd_{max} = vd + \frac{v_{max}}{2000} \quad \text{the initial value of } vd \text{ is set 0} \tag{21}$$

At the first run, the initial maximum speed is 0 and at the second run, the maximum speed is $\frac{v_{max}}{2000}$. And then, after 2 seconds, the maximum permitted speed becomes the maximum speed. At the moment, the maximum speed become reachable. When the robot goes to the end of the trajectory, the same as departure, it is expected that the robot does not stop suddenly. For this, actually, in the point-to-point tracking, it can be realized since the position error becomes smaller and smaller at the end of the trajectory and therefore according to the $P$ control algorithm, the velocity will drop down as it approaches the end objective point, the degree to drop is related to the coefficient $Kp$, the smaller $Kp$ takes, the more smooth it runs.In our project, the maximum speed of robot is about $0.5m/s$, and the corresponding speed is shown as in Fig.4.

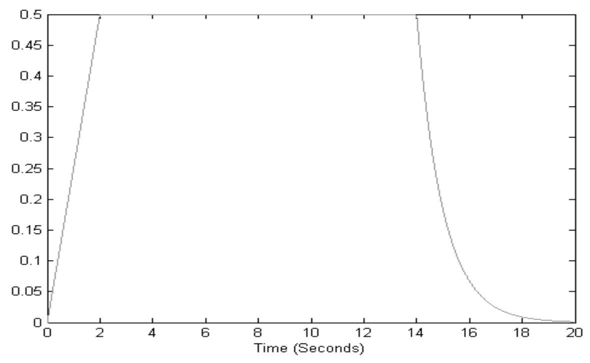
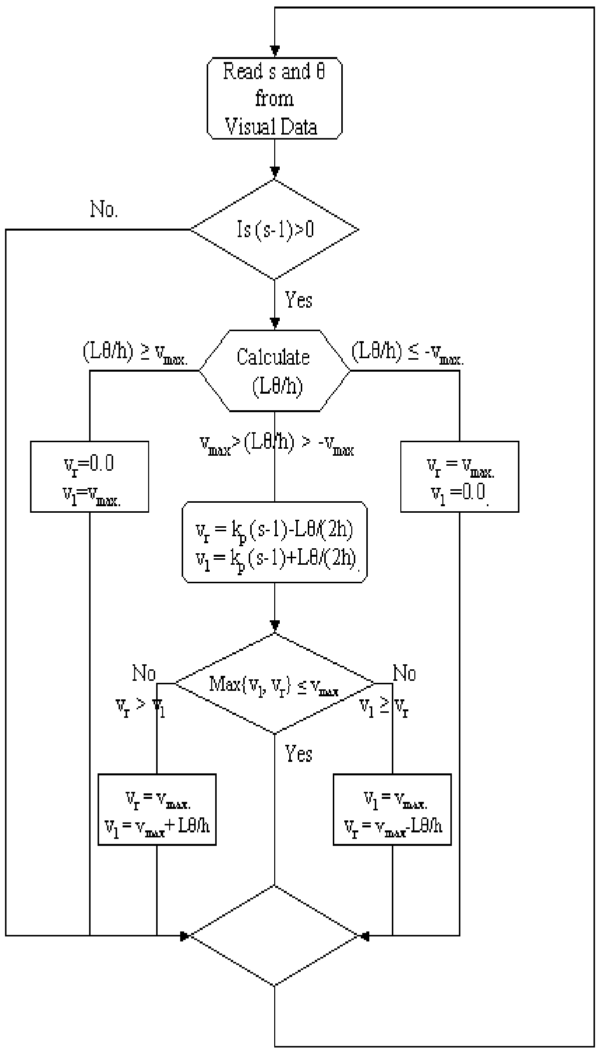


Figure 4: trapezoidal Velocity to Follow Sequent Points



Read s and θ from Visual Data
No.
Is (s-1)>0
Yes
(Lθ/h) ≥ $v_{max}$.
Calculate (Lθ/h)
(Lθ/h) ≤ -$v_{max}$.
$v_{max}$ > (Lθ/h) > -$v_{max}$
$v_r$=0.0 $v_l$=$v_{max}$.
$v_r$ = $v_{max}$. $v_l$ =0.0
$v_r$ = $k_p$(s-1)-Lθ/(2h) $v_l$ = $k_p$(s-1)+Lθ/(2h)
No
$v_r$ > $v_l$
Max{$v_l$, $v_r$} ≤ $v_{max}$
No
$v_l$ ≥ $v_r$
Yes
$v_r$ = $v_{max}$. $v_l$ = $v_{max}$+Lθ/h
$v_l$ = $v_{max}$. $v_r$ = $v_{max}$-Lθ/h

Figure 5: Guest Following

## 4 Guest Following with Certain Polite Distance by Point-to-Point Path Tracking

One of the distinct personality of IRU is that it can follow guest from here to there, keeping a certain distance away from the guest. In our project, we consider the polite distance is one meter away. When, it is farther than one meter, IRU can move near the guest until one meter from the guest. To realize this, the camera together with ultrasonic sensors and infrared sensors can provide two real-time discrete data, one is the distance $s$ from guest to IRU, which should be a positive scalar, the other is the orientation angle $\theta_{object}$ relative to the forward direction of IRU. Here the angle $\theta_{object}$ should take value from $(-\pi, \pi]$. We work out point-to-point tracking strategy to handle this problem. Once the guest is within one meter away, i.e, $(s-1) \leq 0$, no moving needed. If the guest is localized in a direction $\theta_{object}$ angle from the forward direction beyond one meter away. IRU is able to adjust his forward direction to align with the guest direction and move near the guest. Supposing that the sample space is $h$, and the $v_{max}$ is the maximum speed the wheel can move at. The software realization can be expressed as in Fig. 5.

## 5 Conclusion

In this paper, firstly, the computer-DSP architecture design of the real-time system has been introduced and the point-to-point path tracking algorithm together with the Least Square Minimum Path generating was well developed. After that, the application of the point-to-point algorithm was adopted in the guest following. In our experiment, the trapezoidal point-to-point algorithm was proved to work efficiently and the tracking is smooth and elegant, which is uniform with our design. However, there are some limitations to the application of the Point-to-point, for example, at least the objective point is given and the present point is known. In addition, the path should not be tortuous, otherwise, it will be separated into several segments and they will be handled separately.

## References

[1] S.S. Ge and Y.J. Cui. "Dynamic Motion Planning for Mobile Robots Using Potential Field Method" *Autonomous Robots*, 2002, Kluwer Academic Publishers. Manufactured in the The Netherlands. Vol.13, pp207-pp222.

[2] K.S. Fu (editor), R.C. Gonzalez (editor) and C.S.G. Lee (editor). *Robotics*, McGraw-Hill International Editions, Industrial Engineering Editions. 1987.

[3] A.T.De Almeida (Editor) M. Thoma (Editor) and O. Khatib (Editor). *Autonomous Robotic Systems*, Springer-Verlag New York, Incorporated. July 1998

[4] Annibal Ollera, Joaquin Freruz, Omar Sanchez, and Guillermo Heredia. "Mobile Robot Path Tracking and Visual Target Tracking Using Fuzzy Logic" *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, Springer-Verlag Company, 2001, pp51-pp72.

[5] G. Heredia, A.Ollero, F. Gordillo and J. Aracil. "Stability Analysis of Fuzzy Path Tracking Using A MIMO Frequency Response Technique" *IFAC Intelligent Components for Vehicles*, Seville, Spain, 1998.

[6] Phillip C-Y. Sheu and Q. Xue. *Intelligent Robotic Planning Systems*, World Scientific. World Scientific Series in Robotics and Automated systems Vol. 3.

[7] Jian Fei and Can Isik. "Adaptive Fuzzy Control Via Modification of Linguistic Variables" *IEEE*IEEE 1992, 0-7803-0236-2/92. pp399-pp406.

[8] G.M Van Der Molen. "Modeling and Control of a Wheeled Mobile Robot" *IFAC Intelligent Autonomous Vehicles*, Southampton, UK, 1993. pp287-pp292.

[9] Andre Kamga, and Emmauel Simeu. "A Straight Lines Path Following Strategy Applies to a Tricycle Vehicle" *IFAC intelligent Components and Instruments for Control Applications*, Annecy, France, 1997. pp303-pp308.

[10] Chase H. Kenyon, and Pushkin Kachroo (Editor). *Mobile Robots XI and Automated Vehicles Control Systems*, Proceedings of SPIE, Volume 2903, November, 1996.

[11] Gregiry Dudek and Michael Jenkin. *Computational Principles of Mobile Robots*, The Press of the University of Cambridge, United Kingdom, 2000.

[12] J. Sasiadek (editor). *Mobile Robot Technology*, IFAC, International Federation of Automatic Control, Pergamon, 1992.