Proceedings of the 2001 IEEE
International Conference on Robotics & Automation
Seoul, Korea - May 21-26, 2001

A Fast Path Planning-and-Tracking Control for Wheeled Mobile Robots

T. H. Lee
The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
thlee@eie.polyu.edu.hk

H. K. Lam
The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
hkl@eie.polyu.edu.hk

F. H. F. Leung
The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
enfrank@hkpucc.polyu.edu.hk

P. K. S. Tam
The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
enptam@hkpucc.polyu.edu.hk

Abstract

This paper presents a fast method to plan the path, and control the position of wheeled mobile robots (WMRs) in a robot soccer game. In a WMR system with position control, when the target position is given, the controller will generate the reference path and then the WMR will move to the target position following the reference path. To tackle the difficulties of the nonlinear control problem, the motion of the WMR is realized via changing the linear displacement and angular displacement in a separate manner such that the WMR is either rotating or moving in a straight line at any one time. This method takes advantage of a proposed fast path-planning algorithm. The complexity of design is thus significantly reduced.

I. Introduction

This paper presents the position control of wheeled mobile robots, which takes advantage of a proposed simple path-planning algorithm [1]. The micro-robot soccer tournament (MIROSOT) is used as a testing paradigm. In MIROSOT, a camera captures the playground images that are sent to and processed by a host computer. The host computer extracts the locations of the robots and the ball. Based on this information and the soccer game strategy, we can decide the target positions for our team robots. For each robot, the host computer will generate a reference path that avoids any obstacles. Then, it computes the corresponding control signals applying to each wheel. The robot will have both linear and angular displacements until it arrives at the target position. Due to the nonlinear dynamic characteristics of the WMRs, the controller design will be a difficult problem.

In this paper, a simple path-planning algorithm generates line segments as the reference path, and a two-phase control method is used to realize the position and path-tracking controller. There are two advantages: 1) The control method is simple for implementation, and the trajectory of the WMRs can be easily monitored. 2) The boundary conditions can be considered during the path planning so that the WMRs are ensured to move inside the stadium.

Many path-planning methods were reported in the past. The roadmap method [2] involves a pre-processing of the known environment, making it suitable for static environment navigations. The cell decomposition [3] method is based on the idea of a sorting operation defined on the cells in a decomposition of a two-dimensional free space. It also works on a static environment, and the complexity of the path-planning depends on the decomposition method, the number of cells and the algorithm of sorting the cells. The potential field [4] method is commonly used in a robot navigation problem. This approach can work on a dynamic environment, but a large computation demand of the host computer is present.

Thanks to the proposed path planning algorithm's characteristics, the position control of the WMR can be separated into two phases. The first phase controls the angular displacement (θ) of the WMR about its center. The second phase controls the linear displacement (l) of the WMR. Thus, the WMR is running in a piecewise straight-line trajectory. When the robot is near the boundary of the stadium, the robot can stop the linear motion and rotate about its center before moving to any other positions inside the stadium. During the control process, the computer program only needs to detect the obstacle(s) across the line segment(s). More importantly, the controller design problem will become less complex. This is because on separating the control into two phases, the robot in each phase is sufficiently represented by a linear model. We can know exactly the path of the WMR from the current position to the target position within a very short time, and linear control theory can be applied to design the controller for acceptable performance. In a paradigm that requires quick responses, this control method offers distinguished advantages.

This paper is organized as follows. In section II, we focus on the proposed simple path-planning algorithm. It includes the obstacle detection phase and the path generation phase. Section III discusses the nonholonomic wheeled mobile robot system [5]. The kinematic, dynamic equations of the WMR and the controller design methodology will be presented. Simulation results will be given in section IV to show the merits of the proposed method. A conclusion will be drawn in section V.

II. Path planning algorithm for WMR

The camera has an image size fixed at 320×240 pixels per frame, which also governs the size of the coordinate system. With reference to Fig. 1, the following variables are defined.

0-7803-6475-9/01/$10.00© 2001 IEEE

$(x_c y_c)$- Coordinates of the robot center at present (source)
$(x_d y_d)$- Coordinates of the robot center targeted (destination)
$(x_o y_o)$- Coordinates of the obstacle center
w - radius of the effective boundary of the robot, which should be at least longer than $\sqrt{2} \times$ diagonal length of the robot
c_1, c_2, c_3 and c_4 – corner points of the effective obstacle. Their positions (with the y-coordinates denoted by y_{c1}, y_{c2}, y_{c3} and y_{c4} respectively) are at the up-, left- low-, and right-side of the obstacle center respectively. The line joining the source and the destination is called the shortest path. We consider the three vertical line passing through the corner points c_2, c_3, (or c_1) and c_4. The distances between the intersecting points of these lines with the shortest path, and the horizontal axis are denoted by h_{c2}, h_o and h_{c4} respectively. If the shortest path contains obstacle(s), a planned path will be formed by connecting some corner points of the obstacle(s). These specific corner points are called connection points. The angle between the shortest path and the horizontal axis is denoted by θ_r.

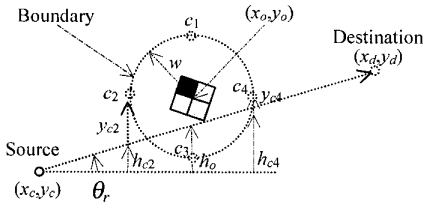

Fig. 1. Graphical representation of various notations.

The path-planning algorithm involves 7 steps in 2 phases: the obstacle detection phase and the path generation phase.

2.1. Obstacle detection phase

Step 1. The position coordinates of the source and the destination are determined based on the image data and the game strategy respectively.

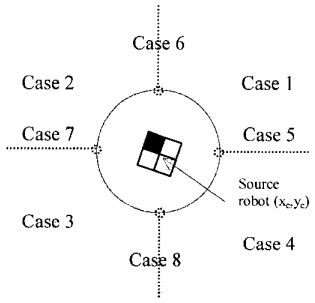

Fig. 2. Different possible areas of the obstacle position.

Step 2. We set the source coordinates as (0,0). It can help defining a case number according to the position of the obstacle. There are totally eight different cases corresponding to the 8 different possible areas containing the obstacle with respect to the source robot as shown in Fig. 2: upper right (1), upper left (2), lower left (3), lower right (4), horizontal right (5), vertical up (6), horizontal left (7) and vertical low (8). For instance, if an obstacle is in the upper right area with respect to the source robot, it is Case 1.

Step 3. We generate the shortest path equation between the source position and the destination position. This shortest path influences the finding of the case number for generating the path.

Step 4. Based on the shortest path and the obstacle center(s) (x_o, y_o), calculate the perpendicular distance as shown in Fig. 3. If the perpendicular distance is less than w, an obstacle is detected and the robot's moving along the shortest path will have a chance to collide with the obstacle. A counter will be used to record the number of obstacles. The obstacles will be sorted in ascending order of the distance from the source robot. If no obstacle is detected, the robot will follow the shortest path to the destination point based on the angle θ_r and the length of the shortest path l_r. The path-planning algorithm then ends.

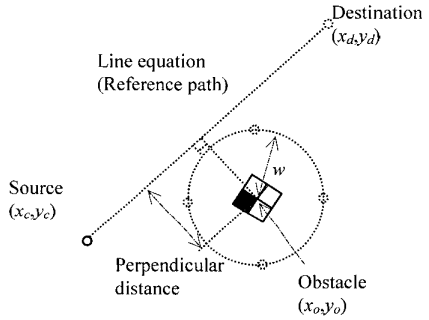

Fig. 3. Obstacle detection method.

2.2. Path generation phase

Step 5. When an obstacle is detected in the shortest path, we find the horizontal distance x_o, which is the x-coordinate of the obstacle center. This distance x_o is substituted into the shortest path equation to find h_o (see Fig. 1). Based on the case number, y_o and h_o, the direction of the planned path can be found. For instance, under case 1 as shown in Fig. 4a, if h_o is greater than y_o (i.e. the y-coordinate of the obstacle center), the path will be generated to the side of the obstacle farther away from the horizontal axis.

Step 6. From the direction of the path determined in step 5 and the values of (x_o, y_o), the path is generated by connecting connection points (if obstacle is detected) in straight lines between the source and the destination. The connection points must be corner points of the effective obstacle (i.e., c_1 to c_4). The conditions governing the generation of connection points are summarized in Table 1. For instance, we consider the situation as shown in Fig.

4b, which belongs to case 3, $h_o < y_o$. Because the test conditions ($h_{c4} > y_{c4} = y_o$) and ($h_o > y_{c3}$) are both satisfied, from Table 1, c_4 and c_3 are connection points. The path is generated as line segments starting from the source passing through c_4 and c_3 to the destination point.

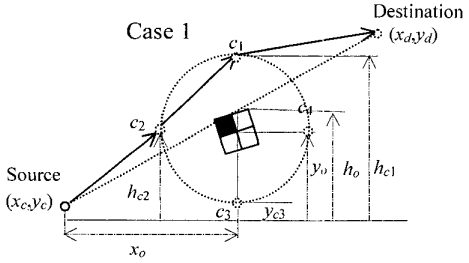

Fig. 4a. Path planning under an obstacle robot (Case 1)

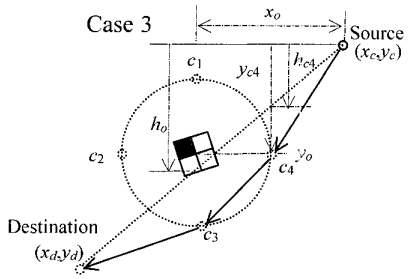

Fig. 4b. Path planning under an obstacle robot (Case 3)

Upon implementing the proposed algorithm, (x_o, y_o) is derived from the image captured. The angle θ_r between the horizontal axis and the shortest path is obtained by $\theta_r = \tan^{-1}\left(\frac{y_d}{x_d}\right)$. The length of the reference path l_r is equal to $\sqrt{x_d^2 + y_d^2}$. Case 5 to 8 correspond to the special cases of θ_r = 0°, 90°, 180° and 270° respectively. In these 4 cases, the program will jump to a special subroutine to generate the path. The new path can be on either side of the effective obstacle. Fig. 5a and 5b illustrate the path generation under case 5 and case 6, respectively.

Case 1	If $h_o > y_o$ then (if $h_{c2} < y_{c2}$ then generate connection point in c_2 and if $h_o < y_{c1}$ then generate connection point in c_1)
	If $h_o < y_o$ then (if $h_o > y_{c3}$ then generate connection point in c_3 and if $h_{c4} > y_{c4}$ then generate connection point in c_4)
Case 2	If $h_o > y_o$ then (if $h_{c4} < y_{c4}$ then generate connection point in c_4 and if $h_o < y_{c1}$ then generate connection point in c_1)
	If $h_o < y_o$ then (if $h_o > y_{c3}$ then generate connection point in c_3 and if $h_{c2} > y_{c2}$ then generate connection point in c_2)
Case 3	If $h_o > y_o$ then (if $h_o < y_{c1}$ then generate connection point in c_1 and if $h_{c2} < y_{c2}$ then generate connection point in c_2)
	If $h_o < y_o$ then (if $h_{c4} > y_{c4}$ then generate connection point in c_4 and if $h_o > y_{c3}$ then generate connection point in c_3)
Case 4	If $h_o > y_o$ then (if $h_o < y_{c1}$ then generate connection point in c_1 and if $h_{c4} < y_{c4}$ then generate connection point in Y_4)
	If $h_o < y_o$ then (if $h_{c2} > y_{c2}$ then generate connection point in c_2 and if $h_o > y_{c3}$ then generate connection point in c_3)
Case 5	If $\theta_r = 0$, then set connection point in c_1 or c_3, the setup point depends on the designer
Case 6	If $\theta_r = 90$, then set connection point in c_2 or c_4, the setup point depends on the designer
Case 7	If $\theta_r = 180$, then set connection point in c_1 or c_3, the setup point depends on the designer
Case 8	If $\theta_r = 270$, then set connection point in c_2 or c_4, the setup point depends on the designer

Table 1. The testing conditions for path generation

Step 7 The next obstacle in the path is considered by repeating steps 4 to 6, until all obstacles have been considered. The final path will then start from the source point, pass through all connection points in straight-line segments, and end in the destination point.

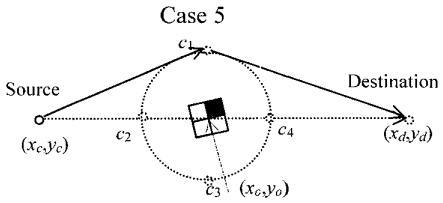

Fig. 5a. Obstacle in horizontal axis.

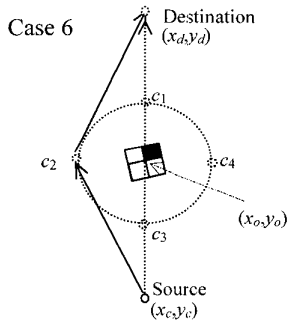

Fig. 5b. Obstacle in vertical axis.

III. Nonholonomic Wheeled Mobile Robots systems

The WMR dynamic equations are obtained by the well-known Lagrange's equations [6],

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\mathbf{q}}}\right) - \frac{\partial L}{\partial \mathbf{q}} = \tau \qquad (1)$$

where τ represents the torque. The Lagrangian variable L is equal to the difference between the kinetic energy k and the potential energy p. The dynamic equation of the mobile robot can be expressed as

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{V}_m(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \mathbf{B}(\mathbf{q})\tau - \mathbf{A}^{\mathrm{T}}(\mathbf{q})\lambda \qquad (2)$$

where $\mathbf{M}(\mathbf{q})$ represents the 3×3 inertia matrix (which is symmetric), $\mathbf{V}_m(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}$ represents the 3×1 vector of centrifugal and Coriolis torques, $\mathbf{G}(\mathbf{q})$ represents the gravity torques, $\mathbf{A}(\mathbf{q})$ is given by the nonholonomic constraints, λ is a Lagrange multiplier associated with the constraints, $\mathbf{B}(\mathbf{q})$ is a 3×2 matrix, and $\boldsymbol{\tau}$ represents the external torques applying to the system. In the present case, the variables are defined as follows:

$$\mathbf{M}(\mathbf{q})=\begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & I \end{bmatrix}, \quad \mathbf{V}_m(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}=\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{G}(\mathbf{q}) = \mathbf{0},$$

$$\mathbf{A}^{\mathrm{T}}(\mathbf{q})=\begin{bmatrix} -\sin\theta \\ \cos\theta \\ 0 \end{bmatrix}, \quad \boldsymbol{\tau}=\begin{bmatrix} \tau_r \\ \tau_l \end{bmatrix}, \quad \mathbf{B}(\mathbf{q})=\frac{1}{r}\begin{bmatrix} \cos\theta & \cos\theta \\ \sin\theta & \sin\theta \\ R & R \end{bmatrix},$$

$\lambda = -m(\dot{x}_c\cos\theta + \dot{y}_c\sin\theta)\dot{\theta}$, I is the moment of inertia of the WMR about its center, m is the mass of the robot, τ_r and τ_l are torque control inputs generated by the right and left motors respectively. R and r are the distance between two wheels and the radius of the wheel respectively. Substituting the above variables into equation (2) and simplifing the equation, we have

$$\begin{bmatrix} \ddot{x}_c \\ \ddot{y}_c \\ \ddot{\theta} \end{bmatrix}=\begin{bmatrix} -\dot{\theta}\sin\theta\cos\theta & -\dot{\theta}\sin^2\theta & 0 \\ \dot{\theta}\cos^2\theta & \dot{\theta}\sin\theta\cos\theta & 0 \\ 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \dot{\theta} \end{bmatrix}+\begin{bmatrix} \frac{\cos\theta}{mr} & \frac{\cos\theta}{mr} \\ \frac{\sin\theta}{mr} & \frac{\sin\theta}{mr} \\ \frac{R}{Ir} & \frac{-R}{Ir} \end{bmatrix}\begin{bmatrix} \tau_r \\ \tau_l \end{bmatrix} \tag{3}$$

This plant model is the basis for designing the position controller of the WMR. More details about the modeling of wheeled mobile robot can be found in [6].

3.1. Controller design methodology

It can be seen, from (3), that the WMR is a nonlinear plant. This nonlinearity makes the controller design a difficult problem. Not only may the controller structure be complicated, but also the computational demand will increase. In a paradigm with only a limit computing power like the MIROSOT, the computational demand has to be kept as low as possible in order to ensure a fast response. In addition, many currently-used position and path tracking controllers for WMRs suffer from the following disadvantages: 1) Some points or regions in the playground cannot be reached, as the boundaries of the playground have not been considered during the design. 2) During the path tracking, the robot may not stay in the desired path during the transient state. This increases the chance the robot crashes into obstacles around the desired path.

In order to have a position and path tracking controller which can overcome the aforementioned disadvantages and yet has a simple enough structure, a two-phase control algorithm is proposed in this paper. The rotational motion and translational motion of the WMR are realized in a separate manner. First, during the rotational phase, the robot will rotate to the desired heading angle. Then the translational phase follows, and the robot will move to the target position in a straight-line. Thus, we can see that a position control problem is solved. To handle a path-tracking problem, the desired path has to be segmented into a number of straight lines. This can be done by employing the simple path-planning algorithm discussed in section 2. In the following, we shall consider the design of the proposed two-phase controller. By considering (3), two models, namely a rotational model and a translational model, can be obtained. The rotational model is obtained from the third equation in (3) and by applying the control signals of the left wheel and right wheel with the same magnitude but opposite sign. It can be written as follows,

$$\dot{\boldsymbol{\theta}}(t) = \mathbf{A}_R\boldsymbol{\theta}(t) + \mathbf{B}_R u_R(t) \tag{4}$$

where $\boldsymbol{\theta}(t)=[\theta_1(t) \quad \theta_2(t)]$, $\theta_1(t)=\theta(t)$, $\theta_2(t)=\dot{\theta}(t)$, $\mathbf{A}_R=\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$ and $\mathbf{B}_R=\begin{bmatrix} 0 \\ \frac{2R}{Ir} \end{bmatrix}$ are the system and the input matrices of the rotational model respectively, $u_R(t)=\tau_r(t)=-\tau_l(t)$ is the control input. For the translational phase, we consider the first and the second equations in (3). As $\dot{\theta}=0$ after the rotating phase has ended, we can apply the control signal with the same magnitude and sign as the inputs. The first and the second equations can then be written as,

$$\dot{\mathbf{x}}(t) = \mathbf{A}_x\mathbf{x}(t) + \mathbf{B}_x u_M(t) \tag{5}$$

$$\dot{\mathbf{y}}(t) = \mathbf{A}_y\mathbf{y}(t) + \mathbf{B}_y u_M(t) \tag{6}$$

where $\mathbf{x}(t)=[x_1(t) \quad x_2(t)]$, $x_1(t)=x(t)$, $x_2(t)=\dot{x}(t)$, $\mathbf{y}(t)=[y_1(t) \quad y_2(t)]$, $y_1(t)=y(t)$, $y_2(t)=\dot{y}(t)$; $\mathbf{A}_x=\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$ and $\mathbf{B}_x=\begin{bmatrix} 0 \\ \frac{2\cos\theta(t)}{mr} \end{bmatrix}$ are the system and input matrices of (5) respectively, $\mathbf{A}_y=\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$ and $\mathbf{B}_y=\begin{bmatrix} 0 \\ \frac{2\sin\theta(t)}{mr} \end{bmatrix}$ are the system and input matrices of (6) respectively, $u_M(t)=\tau_r(t)=\tau_l(t)$ is the control input. It can be seen that (4) to (6) are all linear dynamic equations. Simple linear controllers can be designed for the rotational and translational models by employing linear control techniques. It should be noted that in the translational phase, there are two state-space models. We can use either one of these two state-space models to design a linear controller for this phase under a normal condition. It is because the robot is in the desired heading angle towards

the target position after the rotational phase. On moving in a straight line, when one of the states ($x(t)$ or $y(t)$) equals the corresponding coordinate of the target point, the other state will automatically be in the correct position. In some special cases, we can only use either (5) or (6) to design the controller in the translational phase. This happens when the robot has to move horizontally or vertically. For instance, we may use (5) to design a translational phase controller, and use this controller to control the robot in the translational phase under normal case. When $\cos\theta(t) = 0$, however, we have to use a controller designed based on (6) as $\mathbf{B}_x = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$. In the following section, state-feedback controllers with integral control will be designed for the rotational and translational phases of a WMR. Simulation results will then be given.

IV. Application Example

State-feedback controllers with integral control for the rotational and translational phases are to be designed based on the models of (4) to (6). The parameters of the WMR are as follows: m=0.45kg, r=0.02m, R=0.07m, I=0.01kg-m^2. The maximum linear velocity V of the WMR is 1.48m/s. The maximum torque is 2.1N-m. First, for the rotational phase, we consider (4) and augment it as follows, $\dot{\mathbf{z}}_\theta(t) = \mathbf{A}_\theta \mathbf{z}_\theta(t) + \mathbf{B}_\theta u_\theta(t) + \mathbf{E}_\theta$ (7)

where, $\mathbf{z}_\theta(t) = \begin{bmatrix} \theta(t) \\ \theta_e(t) \end{bmatrix}$, $\theta_e(t) = \int (\theta_r(t) - \theta_1(t))dt$, $\theta_r(t)$ is the reference angle, $\mathbf{A}_\theta = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}$, $\mathbf{B}_\theta = \begin{bmatrix} 0 \\ \frac{2R}{Ir} \\ 0 \end{bmatrix}$, $\mathbf{E}_\theta = \begin{bmatrix} 0 \\ 0 \\ \theta_r(t) \end{bmatrix}$.

$u_\theta(t) = \mathbf{G}_\theta \mathbf{z}_\theta(t)$ (8)

where $\mathbf{G}_\theta \in \Re^{3\times 1}$ is the feedback gain to be designed. In this example, we choose $\mathbf{G}_\theta = [-271.4286 \quad -1.2143 \quad 10714.2857]$ such that the eigenvalues of the closed-loop system are −50, −300 and −500. Then we have $\tau_r(t) = -\tau_l(t) = u_\theta(t)$.

For the translational phase, we consider (6) and augment it as follows, $\dot{\mathbf{z}}_x(t) = \mathbf{A}_x \mathbf{z}_x(t) + \mathbf{B}_x u_x(t) + \mathbf{E}_x$

where $\mathbf{z}_x(t) = \begin{bmatrix} \mathbf{x}(t) \\ x_e(t) \end{bmatrix}$, $x_e(t) = \int (x_r(t) - x_1(t))dt$, $x_r(t)$ is the reference position with respect to the x-axis,

$\mathbf{A}_x = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}$, $\mathbf{B}_x = \begin{bmatrix} 0 \\ \frac{2\cos\theta(t)}{mr} \\ 0 \end{bmatrix}$, $\mathbf{E}_x = \begin{bmatrix} 0 \\ 0 \\ x_r(t) \end{bmatrix}$.

$u_x(t) = \mathbf{G}_x \mathbf{z}_x(t)$ (9)

where $\mathbf{G}_x \in \Re^{3\times 1}$ is the feedback gain to be designed. In this example, we choose $\mathbf{G}_x = [-8599.4000 \quad -183.3300 \quad 26640.0000]\frac{mr}{2\cos\theta(t)}$ such that the eigenvalues of the closed-loop system are $[-3.33 \quad -80 \quad -100]$. Then we have $\tau_r(t) = \tau_l(t) = u_x(t)$. It should be noted that the controller is an adaptive one with respect to θ, which is a known constant updated before each straight-line movement. (9) is employed to control the robot in the translational phase under the case that $\cos\theta(t) \neq 0$. When $\cos\theta(t) = 0$, we have to employ a controller designed based on (6). Augment (6) as follows, we have $\dot{\mathbf{z}}_y(t) = \mathbf{A}_y \mathbf{z}_y(t) + \mathbf{B}_y u_y(t) + \mathbf{E}_y$

where $\mathbf{z}_y(t) = \begin{bmatrix} \mathbf{y}(t) \\ y_e(t) \end{bmatrix}$, $y_e(t) = \int (y_r(t) - y_1(t))dt$, $y_r(t)$ is the reference position with respect to the y-axis

$\mathbf{A}_y = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}$, $\mathbf{B}_y = \begin{bmatrix} 0 \\ \frac{2\sin\theta(t)}{mr} \\ 0 \end{bmatrix}$, $\mathbf{E}_y = \begin{bmatrix} 0 \\ 0 \\ y_r(t) \end{bmatrix}$.

$u_y(t) = \mathbf{G}_y \mathbf{z}_y(t)$ (10)

where $\mathbf{G}_y \in \Re^{3\times 1}$ is the feedback gain to be designed. In this example, we choose $\mathbf{G}_y = [-8599.4000 \quad -183.3300 \quad 26640.0000]\frac{mr}{2\sin\theta(t)}$ =[−38.6973 −0.825 −1198.81] such that the eigenvalues of the closed-loop system are $[-3.33 \quad -80 \quad -100]$. Then we have $\tau_r(t) = \tau_l(t) = u_y(t)$. Simulations have been done to test the path planning and trajectory tracking. The stadium diamension is 1.5m × 1.3m. The radius of the effective boundary (w) is 0.15m. We set the robot initial position and target position to (0,0) and (0.6,0.4) respectively. An obstacle is detected in (0.3,0.3). The path is set under case 1, the connection point c_3 is generated at the position (0.3,0.15). So the actual path is generated as two segments; the first segement starts from (0,0) to (0.3,0.15) and the heading angle (θ) is 0.46 rad. The second segment starts from (0.3,0.15) to (0.6,0.4) and the heading angle (θ) is 0.69 rad. The change of θ with respect to time is shown in Fig. 6a. The x-displacment change with respect to time is shown in Fig. 6b. The y-displacment change with respect to time is shown Fig. 6c.

The actual trajectory is shown in Fig. 6d. The destination is reached in 0.8s.

V. Conclusion

We have proposed a path-planning algorithm for WMRs working in the MIROSOT paradigm. The algorithm is very simple and easy to implement. We only need to calculate the perpendicular distance between the line equation of the shortest path and the obstacle center to detect the obstacle(s). If an obstacle is detected, the connection point(s) of the actual path will be generated. This method reduces the computational demand on the host computer, and the path can be generated within a short time. As a result, the reaction of the soccer robot is quick, and the chance of winning the game is enhanced.

A two-phase control algorithm for WMR has been presented in this paper. Two linear models, namely the rotational and translational models, have been proposed to describe the dynamic behavior of the WMR. The proposed two-phase controller have a simple structure and can be designed easily. A simulation example testing the path planning and trajectory tracking of the WMR has been given to illustrate the merits of the proposed method. During the simulation, we assume that the obstacle image is static. In practice, the obstacle is moving and its position is automatically refreshed every 33ms. In each refresh, we have to update the information, regenerate the path and issue new control commands. Thanks to the simplicity of the proposed algorithm, these steps can be finished within the sampling period.

ACKNOWLEDGEMENT

The work described in this paper is substantially supported by a Research Grant from the Hong Kong Polytechnic University (Project No. G-V764).

References

[1] T.H.Lee, F.H.F. Leung and P.K.S. Tam, "A simple path planning algoritm for a robot soccer game," *in Proc. International Symposium on Signal Processing and Intelligent System*, Guangzhou, China, 1999, pp. 607-611.

[2] L E. Kavraki, M. N. Kolountzakis and J. Latombe, "Analysis of probabilistic roadmap for path planning," *IEEE J. Robotics and automation*, vol. 14 no.1, pp. 166-171, Feb., 1998.

[3] K. Kedem and M. Sharir, Tech. Rep. 253: An efficient motion planning algorithm for a convex polygonal object in 2-dimensional polygonal space, *Courant Inst. Math. Sci.*, New York Univ., New York, NY, 1986.

[4] Y. K. Hwang, N Ahuja, "A potential field approach to path planning," *IEEE Transactions on Robotics and Automation*, Vol. 8, No. 1, pp. 23 –32, Feb., 1992.

[5] B. d'Andrea-Novel, G. Bastin and G. Campion, "Modelling and control of nonholonomic wheeled Mobile Robot," *in Proc. of the IEEE International Conference on Robotics and Automation*, Sacramento, California, April 1991, pp. 1130-1135.

[6] F.L. Lewis, C.T. Abdallah and D.M. Dawson, *Control of Robot Manipulators*. New York: Macmillan, 1993.

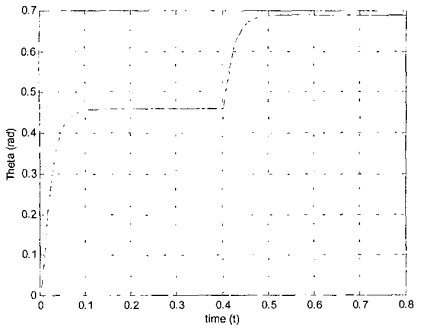

Fig. 6a. Heading angle of the WMR

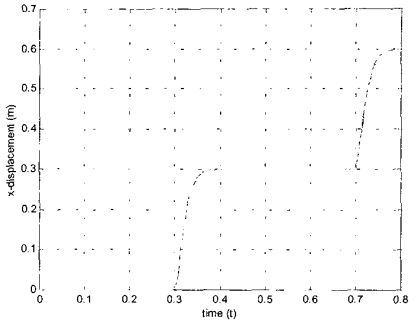

Fig. 6b. *x*-coordinate change with respect to time

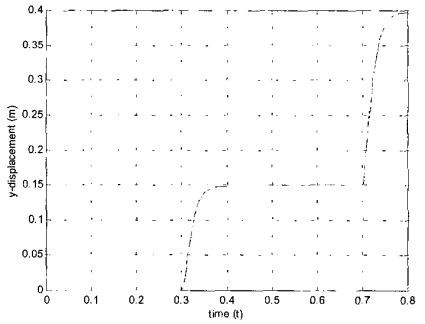

Fig. 6c. *y*-coordinate change with respect to time.

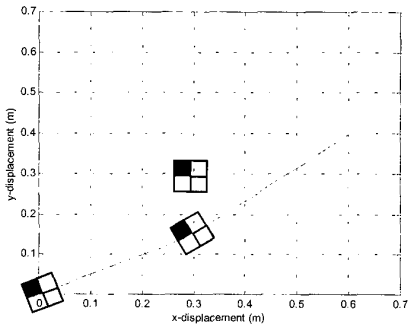

Fig. 6d. The WMR trajectory in the x-y plan.