A Distributed Mobile Robot Simulator and a Ball Passing Strategy.

Tzu-Chen Liang and Jing-Sin Liu

Institute of Information Science 20, Academia Sinica
Nankang, Taipei 115, Taiwan, R.O.C.
hartree@iis.sinica.edu.tw and liu@iis.sinica.edu.tw

Abstract

This report includes two parts. First we build a mobile robot motion simulator. The simulator provides a platform to develop strategies and control methods for robot soccer games. Second, a strategy for three mobile robots to pass a ball in turn with specific direction intention is suggested and realized by this simulator. In the simulator, the robot motion is simulated by dynamical model of type (2,0) unicycle mobile robot. Kick motion follows physical law, and a simplified collision model is utilized to avoid robot overlapping. Ball passing strategy includes a formation and role change scheme, a trajectory planning method and a tracking controller. It formulates the ball passing movement of robots as a problem of tracking and goal adjusting, just like human soccer players do. The results hint that a human-like behavior pattern is applicable in robot soccer games.

1 Introduction

In recent years, robot soccer game enforces research issues such as multi-agent systems, multi-robot cooperative teams, autonomous navigation, sensor fusion, fast pattern recognition and vision-based real time control. It also has been proposed as a benchmark problem in the field of artificial intelligence and robotic systems. Since soccer game is a teamwork game, multiple soccer robots should realize some motion with collaboration to score and defend. However, communication between robots is limited and with noise, so a strategy for the motion decision of each robot by insufficient information should be carefully considered. Some delicate soccer skills, like incessant ball passing movements, requiring not only fast and accurate information getting and transferring but also high performance of controls and trajectory planning, still seldom appear in physical robot soccer games[6][7]. The "Benchmark Test of Robot Soccer Ball Skills" proposed by FIRA (1999)[9] includes a millennium benchmark challenge, which is to control three robots in passing the ball around them in turn forming a circuit. There are only two teams completed this exercise in FIRA Robot World Cup, 2000. In this paper we build a mobile robot simulator and discuss an even more complicate ball-passing strategy for three robots. In this strategy, robots pass the ball with a direction intention and change their formation dynamically according to the ball position. We will also propose corresponded trajectory planning and control method, and then exemplify it by the simulator.

Both major robot soccer game leagues, Robocup and FIRA, hold simulation competition. The objective of simulation game is for joiners who are interested in the software designing to concentrate on the study of artificial intelligence or strategy development. Because of its comparably low cost in participation, teams joining this league are much more then other physical robot leagues. The simulator provided by Robocup, named Soccer Server is very different form which provided by FIRA, called Simurosot. Teams joining Robocup simulation game should develop robot soccer skills, strategies and robot intelligence like human playing soccer, while teams joining FIRA simulation game need to design controller, trajectory planning method and others “machine” soccer skills.

Our robot simulator is an imitation of Simurosot software of FIRA [1], except that the robot motion is simulated by dynamical model. We choose the unicycle mobile robot as soccer player because of its superior mobility and abundant research information. Ball motion is simplified as a pure slipping motion with friction. When the ball collides with the robot, a non-elastic collision model simulates the ball reaction. The simulator can serve as a platform to develop strategies and control methods for robot soccer games.

The objective of the proposed passing strategy is for three robots to carry the ball to a specific direction without holding. This kind of passing strategy is common in human soccer games and is also a basic skill in practicing. Every player does not hold the ball so that opponents cannot intercept it easily. Furthermore, three players working with cooperation make the passing movement more dexterous.

In the next section we will described the simulator and each model. Section 3 the passing strategy is proposed. Section 4 is a simulation to demonstrate its feasibility, and a conclusion will be addressed in section 5.

2 The Simulator

2.1 Model of Mobile Robot

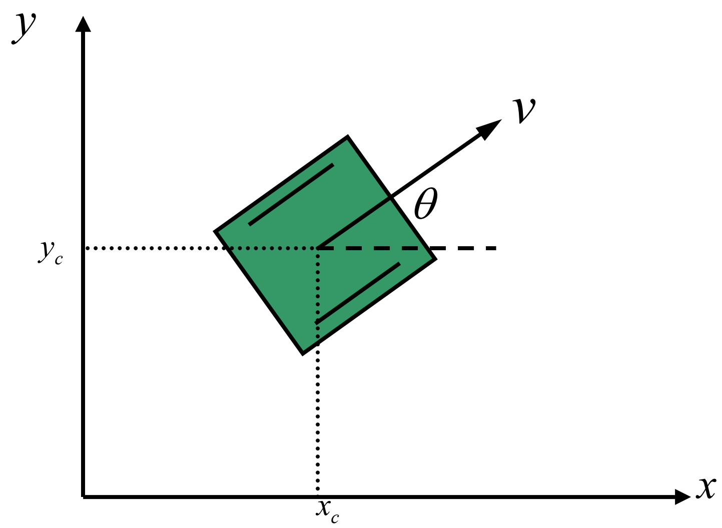


Fig.1. The schematic representation of the mobile robot.

The unicycle mobile robot is shown in figure 1, this kind of robot is mostly used in robot soccer games. Hence we choose it to perform the passing strategy and design controller. The vehicle position is described by the coordinate (x, y) of the mid point between the two driving wheels, and by the orientation angle θ with respect to a fixed frame. Under the hypothesis of "pure rolling" and "non slipping", the vehicle satisfies the nonholonomic constraint,

$$\dot{x}\sin(\theta) - \dot{y}\cos(\theta) = 0 \tag{1}$$

The motion of the robot can be described by the following kinematical model[4],

$$\begin{aligned} \dot{x} &= v\cos(\theta) \\ \dot{y} &= v\sin(\theta) \\ \dot{\theta} &= w \end{aligned} \tag{2}$$

where v is the linear velocity and w is the angular velocity of robot. Let $\dot{v}_L$ and $\dot{v}_R$ denote the velocities of the left driving wheel and the right driving wheel, respectively. Then v and w can be described as,

$$\begin{aligned} v &= (\dot{v}_R + \dot{v}_L)/2 \\ w &= (\dot{v}_R - \dot{v}_L)/l \end{aligned} \tag{3}$$

where l expresses the distance between two driving wheels.

The dynamical model of vehicle is described by the following equations [5],

$$\begin{aligned} \ddot{x} &= -\sin(\theta)\left[\dot{x}\cos(\theta)+\dot{y}\sin(\theta)\right]\dot{\theta}+\frac{\cos(\theta)}{mr}(\tau_R+\tau_L) \\ \ddot{y} &= \cos(\theta)\left[\dot{x}\cos(\theta)+\dot{y}\sin(\theta)\right]\dot{\theta}+\frac{\sin(\theta)}{mr}(\tau_R+\tau_L) \\ \ddot{\theta} &= \frac{l}{Ir}(\tau_R-\tau_L) \end{aligned} \tag{4}$$

Where τ_R and τ_L are driving torques of left and right wheels; m, I are the robot mass, moment of inertia, respectively; r is the wheel radius.

2.2 Discreet Time Simulation

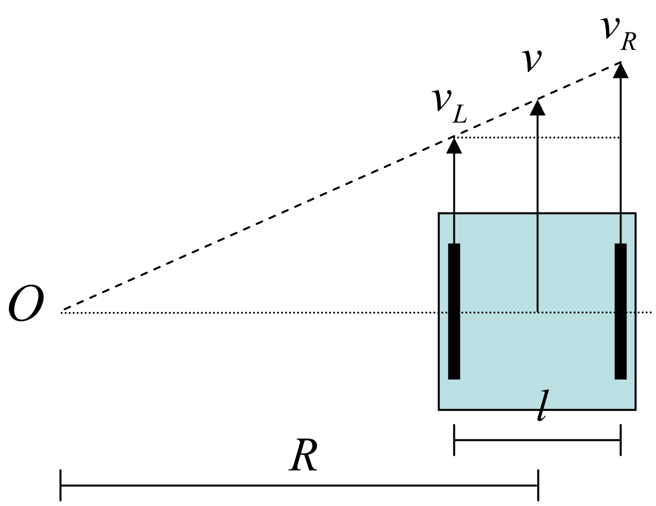


Fig.2. The relationship between the robot velocity, angular velocity and two wheel velocities.

The robot model described above is a set of differential equations, which depicts the continuous motion of mobile robot. However, in computer simulation, we need a discreet time model to fit the computation characteristic of digital computer. This section we will discuss the discreet time model of mobile robot.

Though a discreet version of equation (4) can simulate the robot motion successfully, we still need the kinematical model of equation (2). This is because in mobile robot control, velocity command is more general than torque command, but there is no wheel velocity appeared in equation (4) and the constraint equation is implicit. Therefore, a relationship between wheel torque and wheel velocity should be built and the kinematical model should be applied to assure the constraint condition.

Kinematically, the velocity of mobile robot can be computed by the average of two wheel velocities,

$$v = \frac{v_R + v_L}{2} \tag{5}$$

Differentiate the above equation,

$$\dot{v} = \frac{\dot{v}_R + \dot{v}_L}{2} \tag{6}$$

Because the robot velocity and wheel torques are always at the same direction, we also have,

$$\dot{v} = \frac{\tau_R + \tau_L}{mr} \tag{7}$$

From equation (6) and (7), we have,

$$\dot{v}_R + \dot{v}_L = \frac{2}{mr}(\tau_R + \tau_L) \tag{8}$$

Furthermore, from figure 2, assume that $v_R > v_L$, we can build the relationship of wheel velocity and the angular velocity of robot,

$$w = \dot{\theta} = \frac{v}{R} = \frac{v_R - v_L}{l} \tag{9}$$

Differentiate the above equation,

$$\ddot{\theta} = \frac{\dot{v}_R - \dot{v}_L}{l} \tag{10}$$

From the third equation of (4), we have,

$$\ddot{\theta} = \frac{l}{Ir}(\tau_R - \tau_L) \tag{11}$$

Then from above two equations,

$$\dot{v}_R - \dot{v}_L = \frac{l^2}{Ir}(\tau_R - \tau_L) \tag{12}$$

Solve equation (8) and (12), the relationship between wheel acceleration and wheel torque can be built as,

$$\begin{aligned} \dot{v}_R &= \frac{1}{mr}(\tau_R + \tau_L) + \frac{l^2}{2Ir}(\tau_R - \tau_L) \\ \dot{v}_L &= \frac{1}{mr}(\tau_R + \tau_L) - \frac{l^2}{2Ir}(\tau_R - \tau_L) \end{aligned} \tag{13}$$

This equation shows that one side wheel velocity is not solely dependent on its wheel torque. For a special case,

$$I = \frac{ml^2}{2} \tag{14}$$

We will have,

$$\begin{aligned} \dot{v}_R &= \frac{2}{mr}\tau_R \\ \dot{v}_L &= \frac{2}{mr}\tau_L \end{aligned} \tag{15}$$

The torque-acceleration relation is decoupled. This is beneficial for controller design. In general, we are not easy to build a mobile robot whose momentum of inertia is as described in (14). Therefore a torque-acceleration decouple method is suggested here. The objective of this method is to convert the output of controller, named τ_L^* and τ_R^*, to the nominal τ_L^+ and τ_R^+, which are proportional to accelerations of left wheel and right wheel separately, as the relationship shown in (15).

Torque-acceleration Decouple Method

Figure 5 is the conceptual diagram of the torque-acceleration decouple method. The robot model means equation (13), in which torques and accelerations of two wheels are coupled. The “Nominal Robot Model” includes the robot model and a decoupler pre-connected it. The result is that the relationship between input torques (τ_R^+, τ_L^+) and output accelerations can be described as,

$$\begin{aligned} \dot{v}_R &= \frac{2}{mr}\tau_R^+ \\ \dot{v}_L &= \frac{2}{mr}\tau_L^+ \end{aligned} \tag{16}$$

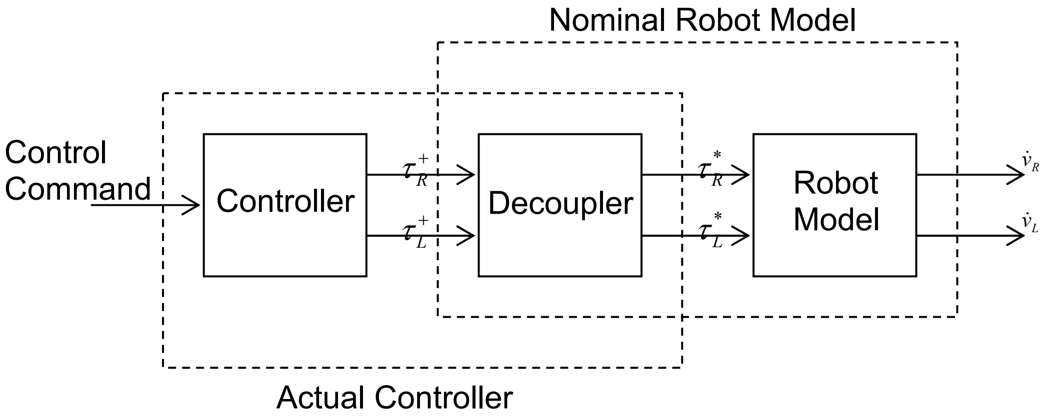


Fig.3. The conceptual diagram of the connection of decoupler, controller and robot model.

Just like (15), which is decoupled, and we can focus on control method design of the simpler model. It should be noted that we do not change the robot model. Therefore the “Actual Controller” is the designed controller plus the decoupler.

The decoupler, which defines the relationship between (τ_R^+, τ_L^+) and (τ_R^*, τ_L^*) can be derived as follows.

Substitute (τ_R, τ_L) in (13) to (τ_R^*, τ_L^*), and combine it with (16) to eliminate $(\dot{v}_R, \dot{v}_L)$,

$$\begin{aligned} \frac{2}{mr}\tau_R^+ &= \frac{1}{mr}(\tau_R^* + \tau_L^*) + \frac{l^2}{2Ir}(\tau_R^* - \tau_L^*) \\ \frac{2}{mr}\tau_L^+ &= \frac{1}{mr}(\tau_R^* + \tau_L^*) - \frac{l^2}{2Ir}(\tau_R^* - \tau_L^*) \end{aligned} \tag{17}$$

Solve (τ_R^*, τ_L^*), as

$$\tau_R^* = \frac{\lambda_2 + \lambda_1}{2\lambda_2}\tau_R^+ + \frac{\lambda_2 - \lambda_1}{2\lambda_2}\tau_L^+$$
$$\tau_L^* = \frac{\lambda_2 - \lambda_1}{2\lambda_2}\tau_R^+ + \frac{\lambda_2 + \lambda_1}{2\lambda_2}\tau_L^+ \tag{18}$$

where $\lambda_1 = \frac{1}{mr}, \lambda_2 = \frac{l^2}{2Ir}$.

From equation (16), for a specific input torque (τ_R, τ_L) and $t = kT$, the wheel acceleration pair $(\dot{v}_R, \dot{v}_L)$ can be computed. If the wheel velocity pair at $t = kT$ is $(\dot{v}_{Rk}, \dot{v}_{Lk})$,then wheel velocities at $t = k(T+1)$ are solved as,

$$\begin{aligned} v_{Rk+1} &= v_{Rk} + \dot{v}_{Rk}T \\ v_{Lk+1} &= v_{Lk} + \dot{v}_{Lk}T \end{aligned} \tag{16}$$

Let (x_k, y_k, θ_k) denotes the robot position and orientation at $t = kT$. Then form equation (2), its position and orientation at $t = k(T+1)$ is

$$\begin{aligned} x_{k+1} &= v_k \mathrm{T} \cos(\theta_k + \frac{\dot{\theta}_k T}{2}) + x_k \\ y_{k+1} &= v_k \mathrm{T} \cos(\theta_k + \frac{\dot{\theta}_k T}{2}) + y_k \\ \theta_{k+1} &= \theta_k + \dot{\theta}_k T \end{aligned} \tag{17}$$

where $v_k = \frac{v_{Rk} + v_{Lk}}{2}, \dot{\theta}_k = \frac{v_{Rk} - v_{Lk}}{l}$.

2.3 Ball Model

The ball locus is a straight line and the motion is pure slipping with friction force proportional to its velocity. Under this assumption, the discreet time ball motion can be modeled as,

$$
\begin{aligned}
v_{k+1} &= v_k + \alpha T \\
X_{k+1} &= X_k + v_k \cos(\varphi) T \\
Y_{k+1} &= Y_k + v_k \sin(\varphi) T
\end{aligned} \tag{18}
$$

where α is a negative constant that describes the deceleration caused by friction force. φ represents the ball velocity direction. φ does not change before any collision happens.

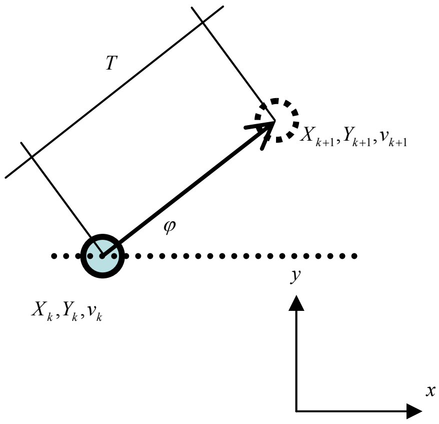


Fig.3. Ball Model

2.4 Kick Model

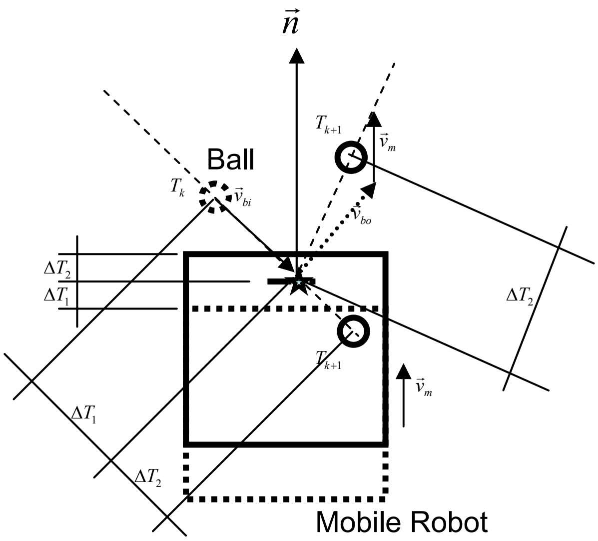


Fig.4. kick model

In the simulator, the robot soccer player does not have specific mechanism to kick ball. Instead, the robot kicks the ball by its front surface, which is orthogonal to the velocity direction. When a collision of ball and robot happens, the kick model is utilized to simulate the ball reaction. We assume that comparing with the ball mass, the robot is heavy and would not change its velocity when kicking. By dynamics, the kicked ball velocity vector can be computed by simply adding robot velocity vector and the zero-speed-collision reflection velocity vector $\vec{v}_{bo}$. But since the simulator is discreet in time, we shall first solve the real kick time and position for reflection computation. As shown in figure 4, at $t=(k+1)T$ the ball and robot are overlapping, which means the kick must happen between $t=kT$ and $(k+1)T$. Geometric computations tell us the real kick position and time, and the ball velocity after kick is computed again by the vector addition described above. Finally, the ball position at $(k+1)T$ is solved by kick position added by velocity vector multiply ΔT_2.

2.5 Collision Model

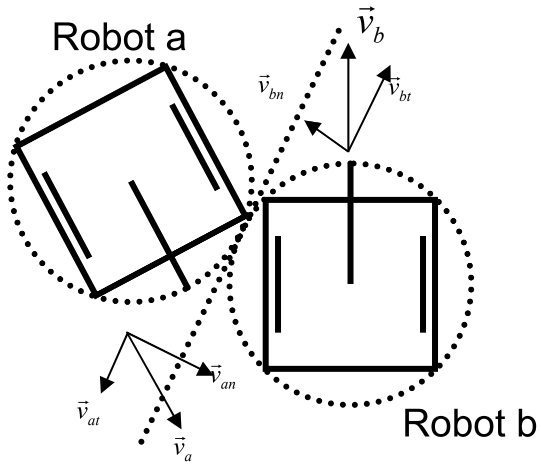


Fig.5. Collision Model

In physical robot games, collision between mobile robots is a complex phenomenon in dynamics and cannot be simulated by mathematical model easily. Since our simulator is developed for strategy and controller design, the complex and accurate collision model is not necessary. We propose a simplified collision model which can avoid the overlap of mobile robots. Though each robot has a square shape in the simulator, we formulate their collision as they are circle shapes for computation simply. As shown in figure 5, two robots collide if their enclosing circles collide, which can be judged by that the distance between two circle's centers is less then 2r. When a collision happens, to avoid overlapping, two robots can only move along the tangent direction of the collision surface, i.e., the normal component of velocities with respect to the collision surface are forced to be zero.

2.6 Simulation Process

The simulation process is shown in figure 6.

t=kT

Receive Control Commands

Compute Robots Motion

Compute Ball Motion

Check Collision and Kick

Compute Collision and Kick Reaction

Let k=k+1

Loop

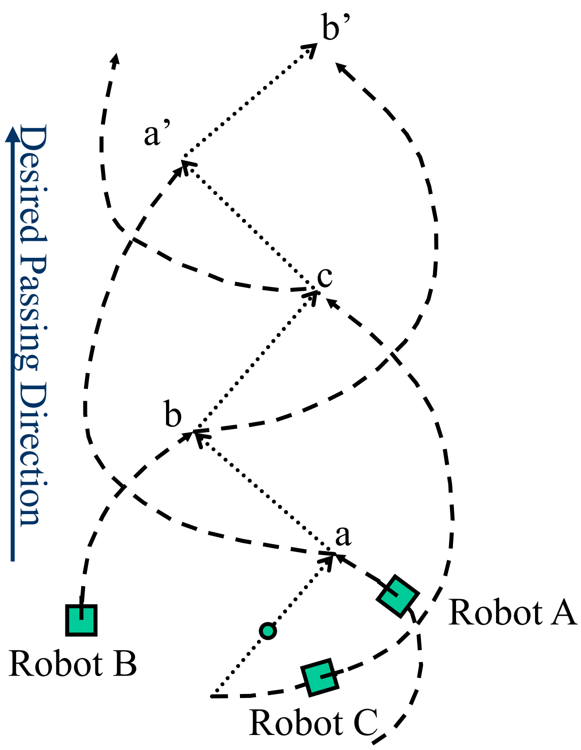
Fig.6. The simulation process.

2.7 Control Command and Strategic Computation

We design that the simulation tick time T=0.02 sec and is equal to the sampling time. The control command at time kT can be arbitrary computed by all motion data accessed at time $t=(k-1)T$. Moreover, all computation about strategy, including ball motion prediction and robot trajectory planning, should be completed in time duration less then T. This restricts the computation amounts so that we cannot make too many computations in strategy.

3. Ball Passing Strategy

3.1 Formation and Role Change Scheme



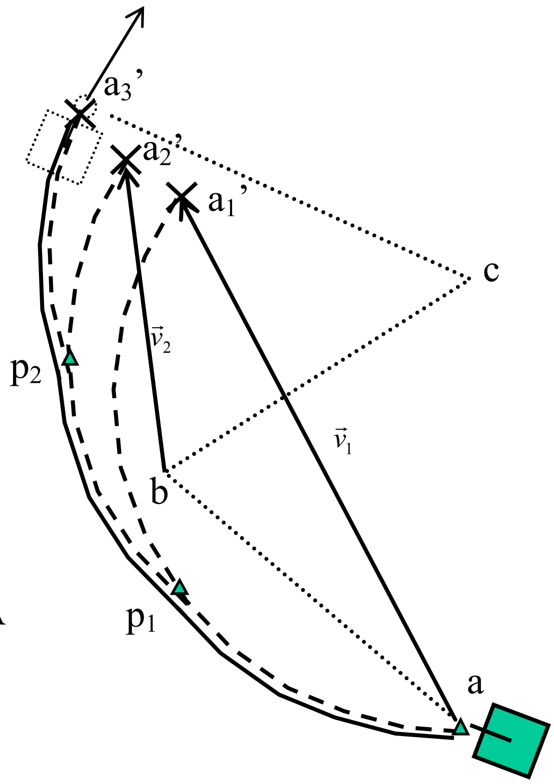


Fig.7.(LHS) The tactical plot of ball passing strategy. Dotted lines represent the ball locus. At each turning point (a, b, c, a' and b') a robot kicks the ball. Dashed lines are trajectories of robots. At this moment, Robot A is Passer, Robot B is Next Player and Robot C is Previous Player.

Fig.8.(RHS) This figure shows the three kick-position corrections of a robot. Dotted lines represent the ball locus; dashed lines are three trajectories generated by kick prediction, and thick solid line is the real trajectory that the robot finally performs. Vector $\vec{v}_1$ and $\vec{v}_2$ are applied to generate the trajectory a-a_1' and p_1-a_2' separately as robot's reference trajectory. Motion predictor tells the robot the exact kick position (a_3'), and trajectory p_2-a_3' is generated.

The ball locus and robot trajectories of our strategy are shown in figure 7. We design the zigzag type ball locus and on each direction change point, a robot kicks it. Three robots kick the ball in turn, as a result the relative positions of two kick points of the same robot are a to a' and b to b', etc. According to our strategy, the passing movement of each robot is a successive motion, i.e. after kicking, the robot runs to the next predicted kick position. Though the situation changes dynamically and the ball motion prediction has many uncertainties, the information is getting accurate while the next kick is impending. There are three times for a robot to correct its trajectory before arriving at the next kick position on time. All three robots do the same thing with different assigned roles, and then the passing strategy is achieved by single kind of controller and single trajectory planner.

For a passing movement, three roles are considered, named Passer, Previous Player and Next Player. Passer is the robot that prepares to kick the moving ball kicked by Previous Player, and Next Player is the robot that the Passer passes the ball to. Each robot acts as one of the three roles at a time and roles change after each kicking: Passer becomes the new Previous Player and Next Player becomes the new Passer. Besides, the Previous Player drives forward and becomes the new Next Player. Three robots do not have a fixed formation during the passing movements, but the formation dynamically changes according to the ball motion. We design the zigzag type ball locus so that the task of robot is to change ball direction and accelerate it to resist the natural deceleration caused by friction. For kick position prediction, we assume that the ball velocity after each kick can be observed by sensor and by prediction, the ball locus is exactly known. The Passer chooses a point on this locus and run there to kick the ball on the specific time point. As for Previous Player and Next Player, because they cannot know their next kick positions exactly, an approximate prediction is performed. As shown in figure 8, the vector $\vec{v}_1$ and $\vec{v}_2$ are dependent on the zigzag locus, which is a function of kick direction and kick velocity. When Passer kicks the ball, two things will happen. First, roles change as described above; second, the kick position is observed by sensor and two vectors are added to this position for setting the goal position of new Next Player and new Previous Player, who are original

Previous Player and Passer. Though vector $\vec{v}_1$ suggests the approximate next kick position for Previous Player and the robot is driven there, the trajectory will not be completed finally because as the robot's role changes to Next Player, a more accurate kick position prediction computed by $\vec{v}_2$ added by new kick position is applied to plan a new trajectory. The robot steers to track the new trajectory. Then when it becomes a Passer, the robot gets the exact ball information and tries to kick the ball.

To summarize, each robot switches its trajectory three times between its two subsequent kicks and the corrections base on other robots' kick position and two prediction vectors. If the vectors $\vec{v}_1$ and $\vec{v}_2$ are chosen appropriately, the change of trajectory will be slight, and the robot motion will be fluent.

3.2 Trajectory Planning

Because all three robots in the passing strategy do the same thing: move to the next kick position, we have only one kind of trajectory planning problem. For convenience, we also let the time duration between two kicks be fixed and separate the trajectory planning problem to a path planning problem and a velocity planning problem. The path planning problem is to find a curve to connect the start-position, which is the robot position, and the end-position, which is the kick position. The selection of this curve should consider the kick direction, i.e. the robot orientation at the kick position. The velocity planning problem is to decide the velocity profile of a robot on the path.

3.2.1 Circular Path Planning: Computation of Center and Radius

Path planning problem for wheeled mobile robot has been widely discussed in many studies [2][3]. They show that a path synthesized by several basic curves, for example: straight lines or part of circles, are practical because of generation fast and tracking easily. For our strategy, the selection of only the circle as the planned path is enough. Referring to figure 9, the current-point (P_c) and the end-point (P_e) are known. The orientation vector of robot at end-point ($\vec{v}_{eb}$), which is the desired kick direction, is decided by strategy. Then the radius (R) of the circular path can be solved by geometric computation, as

$$\begin{aligned} \beta &= 2\alpha \\ R &= \frac{L}{\sqrt{2(1-\cos(\beta))}} \end{aligned} \tag{19}$$

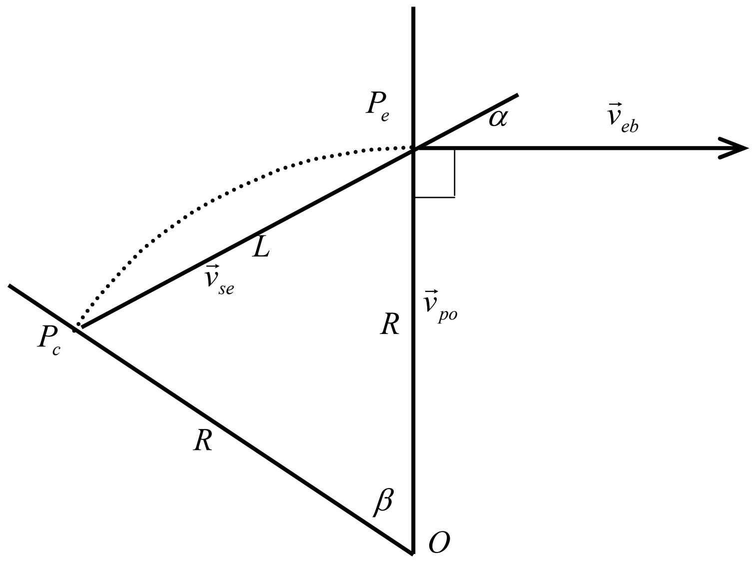


Fig.9. The geometric diagram for the computation of circular path. P_c is the current position of robot. P_e is the predicted kick position, which may be computed by $\vec{v}_{se}$ (either $\vec{v}_1$ or $\vec{v}_2$ in Fig. 8) or ball motion predictor. $\vec{v}_{eb}$ is the desired direction for robot to kick the ball. The dotted line shows the planned path, which is part of the circle centered at O with radius R.

Besides, geometric computation solves the center of this circle (O). Arc $\overset{\frown}{P_cP_e}$ is the proposed path. Note that the robot orientation at the start-point is not considered because in our strategy, when a robot switches to a new planned trajectory, the orientation error is slight and can be easily regulated by tracking control.

3.2.2 Velocity Planning

For a selected arc path s_r, along the s_r direction, the velocity profile of the robot is planned by meeting four boundary conditions,

$$
\begin{aligned}
&s_r(0) = 0 \\
&s_r(t_f) = R \cdot \beta \equiv s_f \\
&\dot{s}_r(0) = V_{robot} \equiv \dot{s}_0 \\
&\dot{s}_r(t_f) = V_d \equiv \dot{s}_f
\end{aligned}
\tag{20}
$$

where V_{robot} is the current velocity of robot; V_d is the desired kick velocity. The instant that the robot starts to execute the path is set to be 0, and the time duration allocated by strategy for the robot to traverse the path is assumed fixed as t_f. We formulate the time trajectory as a 3_{rd}-order polynomial,

$$
s_r(t) = q_1 t^3 + q_2 t^2 + q_3 t + q_4 \tag{21}
$$

Then the velocity profile $\dot{s}_r(t)$ is

$$
\dot{s}_r(t) = 3q_1 t^2 + 2q_2 t + q_3 \tag{22}
$$

Substitute the four B.C.s to above two equations, and the four coefficients of trajectory (21) can be solved as,

$$
\begin{aligned}
&q_1 = \frac{\dot{s}_f{}' t_f - 2 s_f{}'}{t_f^3} \\
&q_2 = \frac{3 s_f{}' - \dot{s}_f{}' t_f}{t_f^2} \\
&q_3 = \dot{s}_0 \\
&q_4 = 0
\end{aligned}
\tag{23}
$$

where $s_f{}' = s_f - \dot{s}_0 t_f$ and $\dot{s}_f{}' = \dot{s}_f - \dot{s}_0$.

We have found the reference of robot's mass center and its velocity profile along it now. Next section we realize the planed trajectory of soccer robot by designing suitable tracking controller.

3.3 Tracking Control

There are three errors in a tracking motion: e_s, e_d, and e_θ. As shown in figure 10, for some instant t', the signed distance between the robot and the reference path is e_d, and distance between the projection, $s(t')$, and $s_r(t')$ is e_s. e_θ is the orientation error of the robot to the tangent direction of the reference path at $s(t')$.

We define

$$\tilde{\tau}(t) = \ddot{s}_r(t) - k_2 \dot{e}_s(t) - k_1 e_s(t) \tag{24}$$

where $e_s(t) = s_r(t) - s(t)$ and $\dot{e}_s(t) = \dot{s}_r(t) - \dot{s}(t)$. Furthermore, let

$$\tau_\Delta(t) = \frac{R}{l} k_2 \dot{s}_r(t) + k_3 e_d(t) + k_4 e_\theta(t) \tag{25}$$

The first term in the RHS of equation (25) is the compensation of differential velocity derived from equation (3); the second and third terms are feedbacks to let the robot ride along the reference path. The suggested tracking controller is

$$\begin{aligned} \tau_R(t) &= \frac{1}{2}\left(\tilde{\tau}(t) - \tau_\Delta(t)\right) \\ \tau_L(t) &= \frac{1}{2}\left(\tilde{\tau}(t) + \tau_\Delta(t)\right) \end{aligned} \tag{26}$$

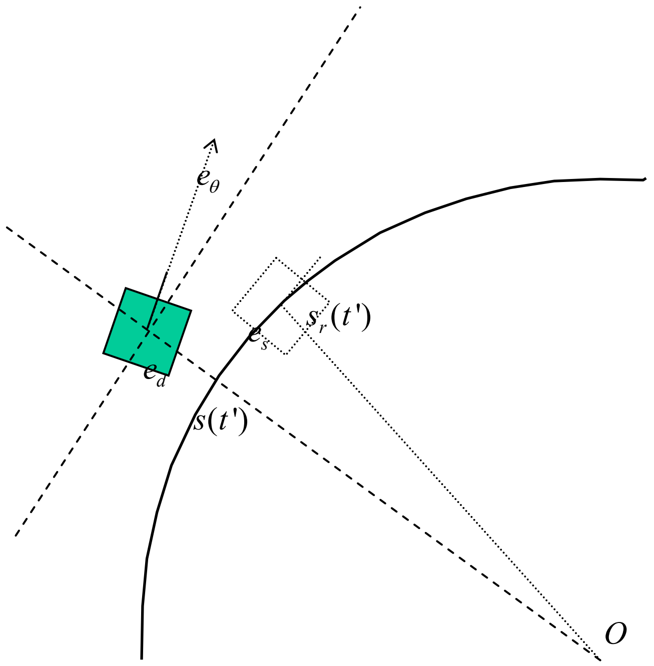


Fig.10. The diagram of three tracking errors at time t'. Solid line is the planned path. O is the center of the arc.

4 Simulations

In the simulation, three identical mobile robots perform the proposed ball passing strategy. The robot mass, moment of inertia and length are $m = 1kg$, $I = 0.02kg \cdot m^2$, and $l = 0.08m$, respectively. Wheel radius is $r = 0.04m$. Maximum linear velocity is $1.5m/\mathrm{s}$, and maximum torque is $0.4N \cdot m$. In each kick motion, the robot does not hold the ball. Instead, it kicks the ball by its front surface, which is orthogonal to the moving direction of robot. The desired kick velocity is $0.7m/\mathrm{s}$, and the kick direction, is set as a $+/-0.25\pi$ angle apart from the desired passing direction. The two vectors $\vec{v}_1$ and $\vec{v}_2$ shown in figure 8 are represented by polar coordinate (l,ϕ) as $(2.34,0)$ and $(3.66,+/-0.145)$ respect to the passing direction. Therefore, when the passing direction rotates, these two vectors and desired kick direction also change accordingly. As a result an intention of change ball passing direction is realized without utilizing any more commands.

The initial relative positions of ball and robots are shown in figure 11. Robot "John" is the first Passer, "Mary" is the Next Player, and "Rosa" (Nominally, the Previous Player) will kick the ball after Mary. Then the ball returns to John and another passing cycle starts. Time duration between subsequent two kicks is set to be 1.6 sec. In first simulation, after first 5 kicks, desired passing direction is designed to rotate clock-wisely by an angle $0.2rad.$ after each kick. In the second simulation, we design robots passing the ball to execute an S-shape course.

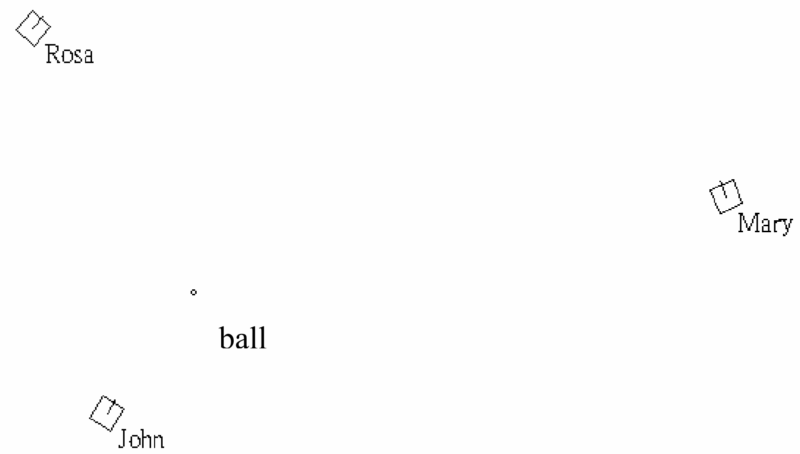
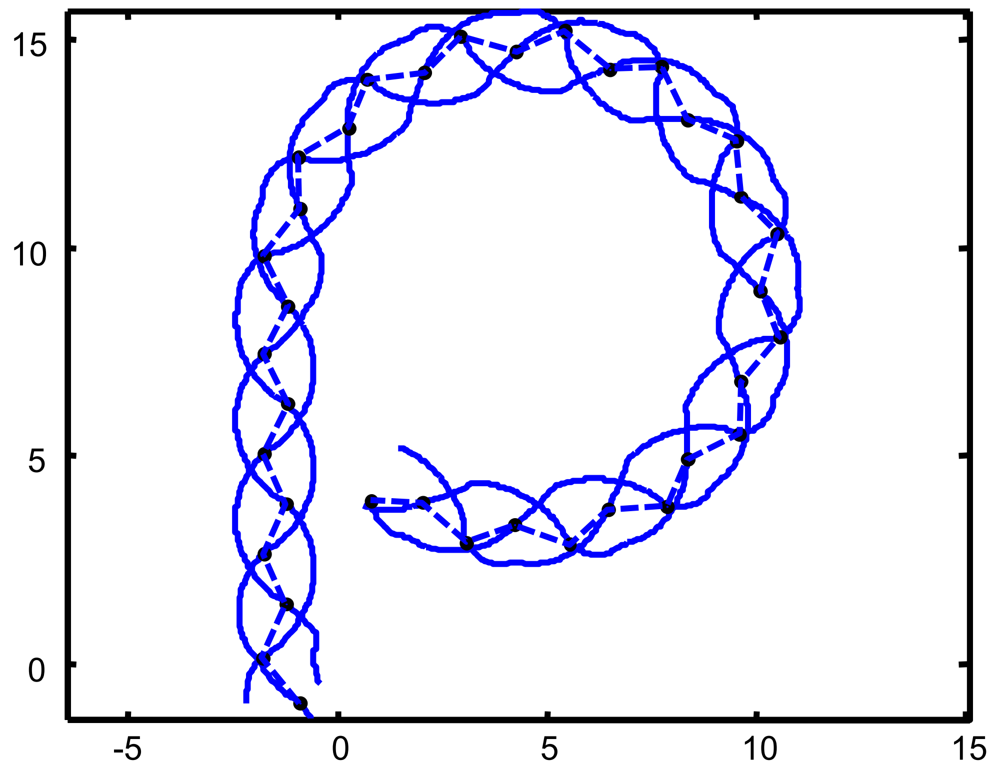


Fig.11. The initial positions of three robots and the ball.

Figure 12 and 13 are the simulation results. Three trajectories of robot motion and the locus of ball are plotted in the $x-y$ plane. Figure 14a-d show the velocity and position tracking error of robot "John" in the first simulation. Since the width of robot is $80mm$, small tracking errors do not cause ball missing. In figure 15, we select a part of robot John's trajectory in the second simulation and the planned trajectory to compare. Path planning method described in section 3.2.1 generates these circles, and part of each circle is a reference trajectory of robot. As the robot corrects the prediction three times between its two subsequent kicks, in figure 15 three circles form a group.



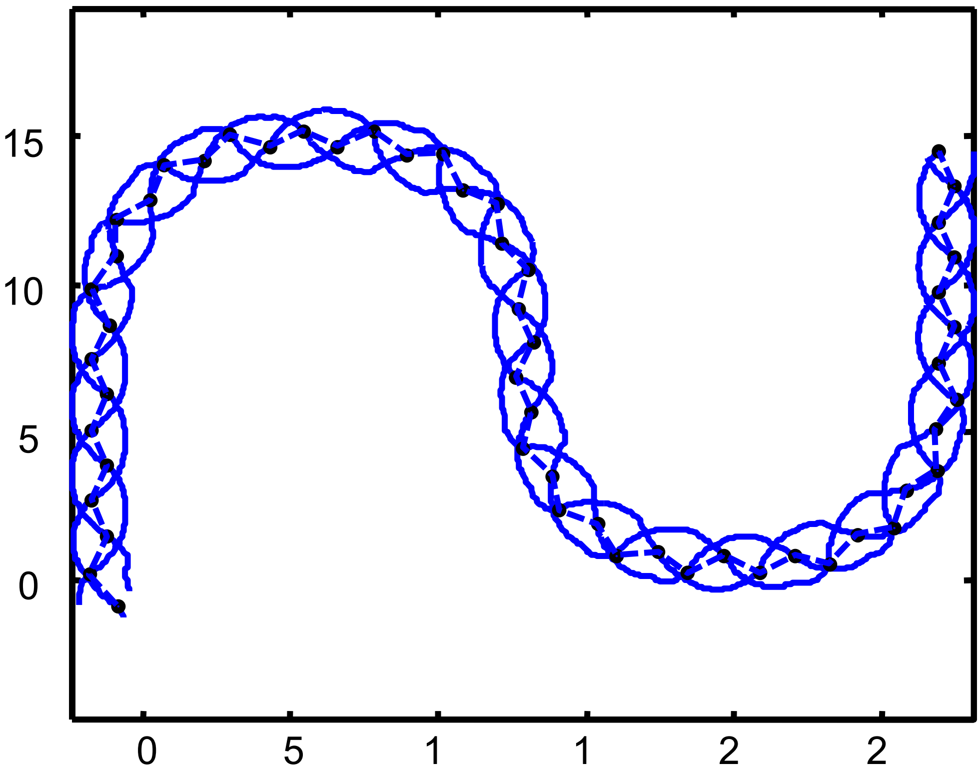


Fig.12&13. The simulation results of the ball passing strategy in first 36 kicks. Solid lines are trajectories of three robots; dashed line is the ball locus, and dots are the positions where a robot kicks the ball.

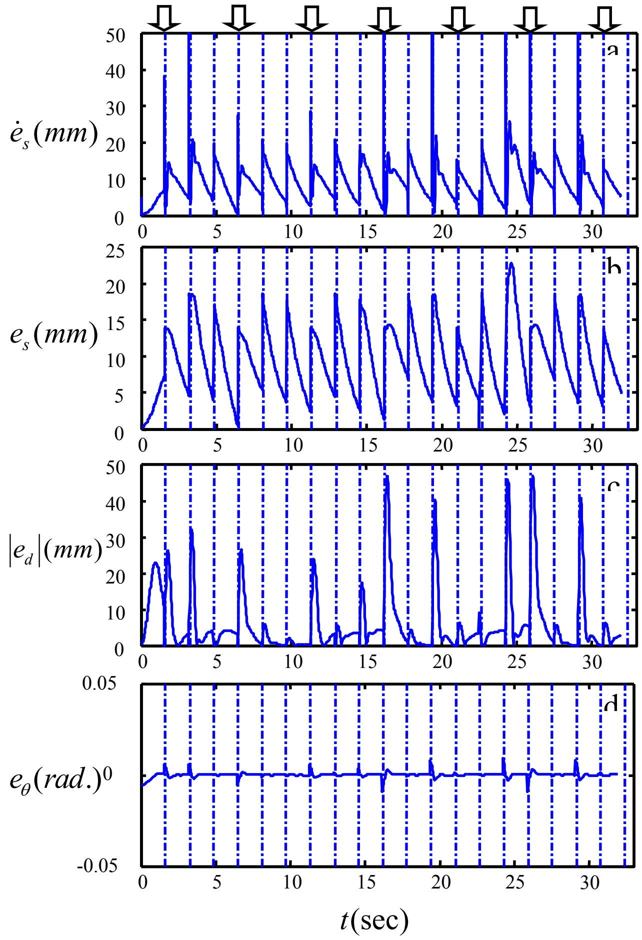


Fig.14. The error-time plots of selected robot “John”. These four figures show the evolution of $\dot{e}_s, e_s, e_d, e_\theta$ in first 33 sec of simulation. Dot-dashed lines mark the time instant that a kick happens. Arrow signs point out that these kicks are completed by John. e_d is plot by its absolute value.

5 Conclusion

In this report we built a computer simulator for multiple mobile robots and ball. We also proposed a ball passing strategy for robot soccer games, and verified it by computer simulation. The simulator serves as a platform for strategy development and controller design. Because it is an imitation of the FIRA’s simurosot, the developed strategies can be applied to simulation league robot soccer games. Furthermore, the simulator utilized dynamical model in robot motion. This is the necessary inclination when the strategy will practice in physical robot soccer games.

The ball passing strategy makes three robots work with cooperation to carry the ball at a specific direction without holding. Instead of using complex trajectory planning method or long-term prediction, it is accomplished by appropriate role assignments and goal change. The simulation results evidence the practicability. Moreover, the idea that a soccer robot adjusts its trajectory by getting accurate information while executing a specific task hints that a human-like behavior pattern can be successfully applied in robot soccer games. To convert more soccer skills by the novel idea is the future work.

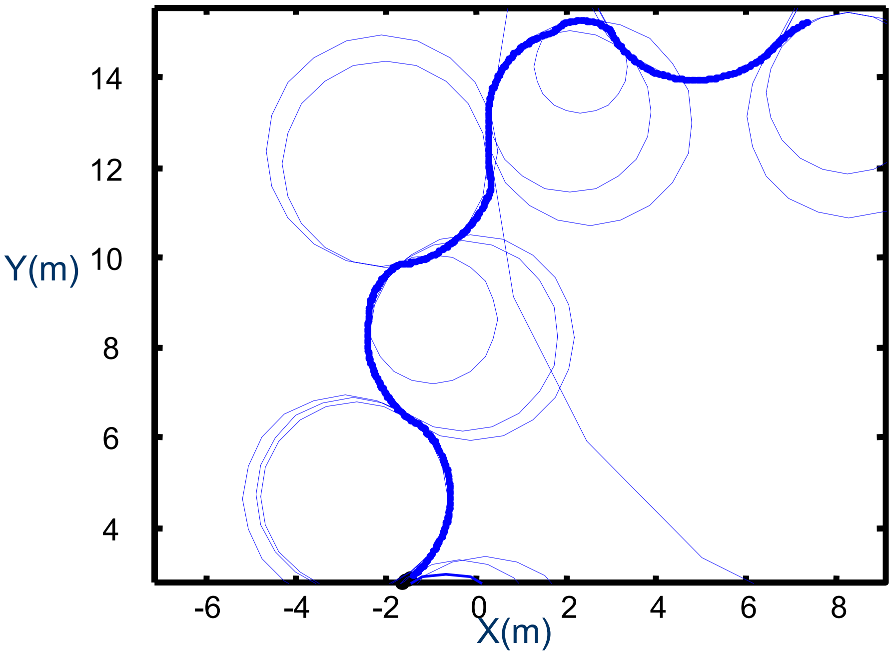


Fig.15. The comparison of planned path with the robot trajectory. The thick solid line is part of robot John's trajectory in simulation result. The circles are planned path generated by the method described in section 3.1.

References

1. Hong Bingrong, Gao Quansheng, et al. (2000), *Robot Soccer Simulation Competition Platform Base on Multi-agent*, FIRA-KAIST Cup Workshop.

2. P. Soueres and J.P. Laumond. (1996) *Shortest Paths Synthesis for a Car-Like Robot*", IEEE Trans. on Automatic Control, vol. 41, no. 5, 672-688.

3. Yutaka Kanayama and Shin'ichi Yuta. (1988) *"Vehicle Path Specification by a Sequence of Straight Lines"*, IEEE Journal of Robotics and Automation, vol. 4, no. 3, 265-276.

4. Carlos Canudas de Wit, Bruno Siciliano and Georges Bastin. (1997), *Theory of Robot Control*, Springer-Verlag, London.
5. M. L. Corradini and G. Orlando. (2001), *Robust Tracking Control of Mobile Robots in the Presence of Uncertainties in the Dynamical Model*, Journal of Robotic Systems, vol. 18, no. 6, 317-323.
6. Minoru Asada and Hiroaki Kitano. (eds.) (1999), *Robocup-98: Robot Soccer World Cup II*, Springer-Verlag, Berlin Heidelberg.
7. Peter Stone, tucker Balch, Gerhard Kraetzschmar. (eds.) (2001), *Robocup 2000: Robot Soccer World Cup IV*, Springer-Verlag, Berlin Heidelberg.
8. Byoung-Ju Lee and Gwi-Tae Park. (1999) *A Robot in Intelligent Environment: Soccer Robot*, Procceedings of the 1999 IEEE/ASME International Conference on Advanced Intelligent Mechatronics.
9. Jeffery Johnson, Josep Lluis de la Rosa, et al. (1999), *Benchmark Tests of Robot Soccer Ball Control Skills*, The Federation of International Robot-soccer Association (FIRA).