International Journal of Automation and Computing 7(1), February 2010, 55-63
DOI: 10.1007/s11633-010-0055-1

Action Control of Soccer Robots Based on Simulated Human Intelligence

Tie-Jun Li[1] Gui-Qiang Chen[2] Gui-Fang Shao[3]

[1]Department of Information Engineering, Jimei University, Xiamen 361021, PRC
[2]Institute of Command Automation, Chongqing Communication College, Chongqing 400035, PRC
[3]Department of Automation, Xiamen University, Xiamen 361005, PRC

Abstract: A multi-modal action control approach is proposed for an autonomous soccer robot when the bottom hardware is unchangeable. Different from existing methods, the proposed control approach defines actions with the principle of "perception-planning-action" inspired by human intelligence. Character extraction is used to divide the perception input into different modes. Different control modes are built by combining different control methods for the linear velocity and angular velocity. Based on production rules, the motion control is realized by connecting different perceptions to the corresponding control mode. Simulation and real experiments are conducted with the middle-sized robot Frontier-I, and the proposed method is compared with a proportional-integral-derivative (PID) control method to display its feasibility and performance. The results show that the multi-modal action control method can make robots react rapidly in a dynamic environment.

Keywords: Intelligent control, artificial intelligence, PID control, perception, action control, robots.

1 Introduction

Industrial robots play an important role in the field of manufacturing. Because of improvements on hardware and software, their performance has greatly enhanced, and their application cases have significantly increased. Many types of robots, such as fish robot[1], mobile robot[2], snake robot[3], and humanoid robot[4], have been developed so far. The control problem for wheeled mobile robots, especially for the autonomous soccer robot, has gained increasing attention during the past few decades.

In the previous studies, most researchers paid attention to decision-making on how to select one behavior to meet the need of the robot′s role or environment[5−7]. Meanwhile, the bottom hardware control system was also addressed, like the structure design of the control system and the choice of the hardware unit[8, 9]. Some of them might be focused on the realization of certain behavior, for example obstacle avoidance[10, 11] and shooting[12]. These studies prompted the development of robots.

However, no researcher has yet designed a unified method for bottom action planning and control, that is to say, a control strategy that should be taken to realize the action selected by high level decision-making. Bottom action means that which can realize certain functions and cannot be divided into other actions, for instance, going to a fixed point, shooting or something else[13]. In terms of the car-like autonomous soccer robot, any kind of bottom action consists of a series of basic motions, i.e., rotation and movement in a direction (such as front, back, left, or right)[1]. A proportional-integral-derivative (PID) algorithm for bottom motion control has been currently adopted in most car-like robot systems. Once the PID controller has been designed, its parameters are invariable, so these parameters cannot be adjusted according to the change of actual load. In order to acquire a fast response rate and a better adaptability of the soccer robot, some intelligent control methods are introduced into the traditional PID controller, such as neural networks[14, 15], linear quadratic (LQ)[16], and fuzzy control[17].

Human beings are able to accommodate external environmental disturbances and effectively regulate their behaviors. They have the ability to adjust their skills or behaviors in various environments. Therefore, particular attention has been paid to the study of how to apply the strong adaptive natural mechanisms of human beings to man-made systems, such as humanoid robots and unmanned air vehicles. Artificial intelligence is just a branch of computer science that aims to simulate the nature of the human mind and create the intelligence of machines.

For the purpose of improving the dynamic performance and precision of bottom action, a multi-modal control algorithm based on human simulated intelligence is proposed in this paper. That is, given different control strategies, the bottom action will yield various responses according to the mapping rule. The advantage of the multi-modal method is that the robot can perform at a fast response rate and with a better adaptability. The behavior-based approach, proposed by Brooks[18], is a methodology for designing autonomous robots. Most behavior-based systems are reactive, i.e., the robot control strategy is embedded into a collection of condition-action pairs. In contrast, the planning part is considered in the multi-modal control algorithm, and genetic algorithm is used to compute the parameters.

This paper is organized as follows. In Section 2, the kinematic model of the wheeled robot, Frontier-I, is introduced. The proposed action control algorithm is described in detail and a parameter confirming the method is established in Section 3. The simulation results and actual game records are presented to demonstrate the effectiveness of the

Manuscript received March 3, 2009; revised May 13, 2009
This work was supported by National Natural Science Foundation of China (No. 60443004) and Science and Technology Project of CQ Education Committee (No. KJ080621).

proposed method in Section 4. Finally, conclusions and future work are given in Section 5.

2 Kinematic model

The Frontier-I, which is an autonomous robot designed by Shanghai Jiao Tong University[19], is introduced as the experiment object to test the effectiveness of human-simulated intelligence control (HSIC) method. It is a car-like robot with two wheels (left and right) on one axle, as shown in Fig. 1.

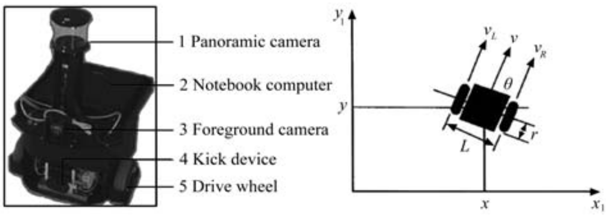


Fig. 1 A car-like robot and its generalized coordinates

For simplicity, it is assumed that the two wheels collapse into a single wheel located at the midpoint of the axle. The generalized coordinates are $q = (x, y, \theta)$, where x and y are the Cartesian coordinates, and θ measures the orientation of the car body with respect to the x axis.

In the actual control process, the wheel velocity should be changed into linear velocity v and angular velocity ϖ by the following transformation

$$\begin{cases} V_L = r\varpi_L \\ V_R = r\varpi_R \end{cases} \Rightarrow \quad \begin{aligned} v &= \frac{V_L + V_R}{2} \\ \varpi &= \frac{V_R - V_L}{L}. \end{aligned}$$

It can be also expressed as

$$\begin{bmatrix} v \\ \varpi \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{L} & -\frac{1}{L} \end{bmatrix} \begin{bmatrix} V_L \\ V_R \end{bmatrix} \tag{1}$$

where V_L and V_R are the left wheel velocity and the right wheel velocity, respectively; R is the radius of the wheel; L is the distance between the wheels.

According to (1), the angular velocity of the centroid is zero when V_L is equal to V_R, so the robot will move in a straight line. If $V_L = -V_R$ then the linear velocity is zero, so the robot will rotate where it is. Therefore, the kinematic model can be derived as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} \nu \\ \omega \end{bmatrix}. \tag{2}$$

From (2), it can be seen that we only need to control the left and right wheel velocity, then the robot position and pose will be changed.

3 Action control algorithm

A robot soccer game is basically composed of robots, a vision system, a host computer, and a communication system. In addition, decision-making is most important for the robots to think like human beings. For example, the robot may make a choice whether it should play as a vanguard or a backfield according to the current environment, self state, and other information. In addition, the behavior selection will affect the decision-making. Most teams have divided the behavior selection into several levels according to their intelligence degree[20, 21]. For instance, basic action, skilled action, combinatorial action, and tactical action, where the intelligence degree increases from low to high, while the control precision decreases from high to low. During the behavior selection, these actions are decomposed from top to bottom.

In this paper, how to make a decision or how to select a behavior is not our issue. We only focus on how to execute the selected behavior, i.e., how to make the robot move as we expect.

3.1 Structure designing

In the present soccer robot systems, most adopt a PID algorithm to realize motion control. Because of various disturbances, the non-linearity, and uncertainty of robot motion, the traditional PID control algorithm cannot meet the need of actual control. It is essential to propose a new method for bottom action control, so that the robot can adapt to a variety of environment and state information at any moment. In an actual match, some factors must be taken into account when the bottom action control algorithm is designed:

1) It requires the robot to react timely, i.e., real-time reaction;

2) There are multiple outputs, such as the linear velocity and the angular velocity, which affect each other;

3) There are many control targets to be balanced, for instance, the rapid control and the track optimizing.

Human beings can adjust their behaviors when they perceive the change of the environment state. Similarly, inspired by human intelligence, we design a new method, where we first divide the robot motion into different phases based on perception information, then select different control modes to realize motion control. Fig. 2 illustrates the process of action control.

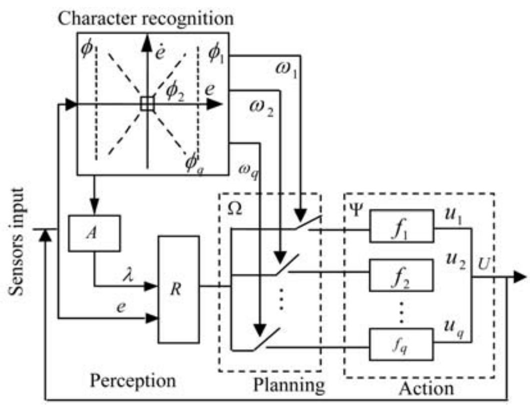


Fig. 2 Sketch diagram of the multi-modal method

The new controller consists of three parts: perception that obtains the states related with environment, action for motion control, and planning which switches actions according to perception.

1) Perception module

Frontier-I has four sensors, i.e., foreground camera, panoramic camera, odometer, and photoelectric cell (only placed on a vanguard robot). All these sensors can provide lots of information, but only part of them is useful for action control. Therefore, we can define the perception module as $P = \{\ R \quad Q \quad K \quad \otimes \quad \Phi\ \}$.

From input information $R = \{r_1, r_2, \cdots, r_n\}$, we can gain the character primitive $Q = [q_1, q_2, \cdots, q_m]^{\mathrm{T}}$ through feature extraction. Following "and", "or" operation, the characteristic model $\Phi = \{\phi_1, \phi_2, \cdots, \phi_q\}$ can be obtained, i.e., we can divide the perception space into different parts with these character primitives.

$K \in \Sigma^{r\times m}$ is a relationship matrix, $\otimes$ is an operation signal. The relationship between characteristic mode $\Phi \in \Sigma^q$ and character primitive vector Q is

$$\Phi = K \otimes Q$$

that is

$$\phi_i = K_i \otimes Q = \bigcap_{j=1}^{m} k_{ij} q_j \in \Sigma^q \tag{3}$$

where k_{ij} is the coefficient which can be taken as -1, 0, or 1 to denote negative, zero, and positive, respectively.

For a soccer robot, the ball is one of the most important targets and the final role is to dribble it into the opponent goal. Because the ball is usually under a moving state, the following four input parameters are especially crucial

$$R = \left\{\ e_d \quad \dot{e}_d \quad e_\theta \quad \dot{e}_\theta\ \right\}$$

where e_d is the distance between the robot and the target $\dot{e}_d$ describes whether the target is close to the robot or not, e_θ is the angle between the robot positive direction, and the target direction $\dot{e}_\theta$ indicates whether the robot heads toward the target or not.

2) Action module

As to robot motion control, there are many basic control methods such as proportion control P, integral control I, differential control D, PD, PID, and so on. Each method has its own feature, so we must apply them according to the actual need and then the desired control effect can be gained. Therefore, the action module can be denoted as follows:

$$M = \{\ R \quad P \quad L \quad \Psi \quad U\ \}$$

where $P = [p_1, p_2, \cdots, p_n]^{\mathrm{T}} \in \Sigma^n$ is the control mode primitive vector, in which each element is the function of input information $p_i = f(R)$. Through the different combination of control mode primitives, we can get the control mode $\Psi = \{\psi_1, \psi_2, \cdots, \psi_n\} \in \Xi^p$, which represents the qualitative or quantitative relationship between control output U and input R, as described by

$$\Psi : U = L\cdot P, \psi_i = L\cdot P = \left\{ u_i = \sum_{j=1}^{n} l_{ij} p_j \right\} \in \Sigma^p. \tag{4}$$

Similarly, $L \in \Sigma^{p\times n}$ is the relationship matrix and l_{ij} is the coefficient which can be -1, 0, or 1.

3) Planning module

The main role of the planning module is to determine what kind of control mode to be selected under some perception state. The planning module is actually the heuristic and instinctive deduction rule sets Ω for harmonizing the perception module and the action module. It can be described by the production rule as

$$\begin{aligned} &\Omega : \Phi \xrightarrow{H} \Psi, \Omega = \{\omega_1, \omega_2, \cdots, \omega_r\} \\ &\omega_i = \left\{ \text{if} \bigcup_{j}^{h_i(x)} \phi_{ij} \text{ then } \psi_i \right\} \in \Sigma^r, \\ &x \in q, \quad h_i(x) = C_q^y, \quad 1 \leqslant y \leqslant q \end{aligned} \tag{5}$$

where x expresses the label of characteristic mode Φ, $h_i(x)$ $(i = 1, 2, \cdots, q)$ denotes a subset in characteristic mode label set $h_i(q)$ of Φ, i.e., a list of preconditions that are related with sensory information, while ψ_i is one of the possible responses. In addition, the mapping relationship between Φ and Ψ is one to one or multiple to one.

In fact, the planning module is a set of IF-THEN rules. In order to display the procedure of the HSIC algorithm, we take two bottom actions (obstacle avoidance and target tracking) as examples, and introduce them respectively in the following section.

3.2 Perception information acquisition

The Frontier-I robot has two cameras for vision perception. One camera is placed in front of the robot and the other is placed above, as shown in Fig. 1. Owing to the difference of installation location and the camera features, they play different roles in actual competition[22]. The nonlinearity of the foreground camera is smaller than that of the panoramic camera. Therefore, the foreground camera is mainly used to accurately locate the vertical and horizontal distances, while the panoramic camera is used to calculate the moving angle for the robot. The object in the match field can be identified by the camera according to its color and size.

However, relevant information extraction from the vision system is a complex process. Inaccurate information may result in a wrong action. Therefore, the design of the vision system must satisfy those specific requirements and constraints of the robot and its mission. Through the cameras, we can obtain the perception input information and some characters such as $e > \alpha_1$, $\dot{e} < \beta_1$. Based on these characters, the information space is divided into several parts which express the system states, as shown in Fig. 3.

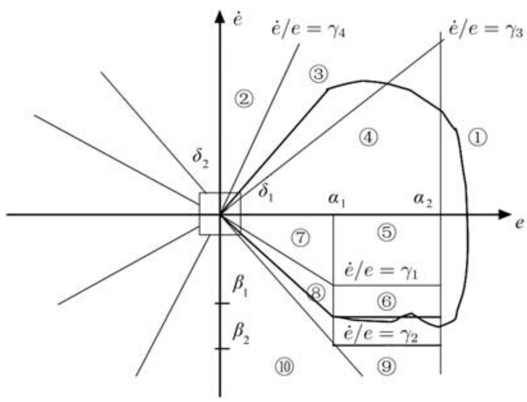


Fig. 3 Error phase orbit and character mode partition

3.3 Obstacle avoidance

In a soccer robot match, there are two kinds of obstacles, i.e., static obstacles (sidewall around the court) and dynamic obstacles (the opponent robot, robot of our team, etc). Not all of the objects can influence the robot; only those near it and affecting its motion path can be viewed as obstacles, for instance, the objects between the robot and the target. Thus, there must be a rule to describe the obstacles. As the collision between robot and obstacles may bring a serious disaster, a new method must be designed for collision avoidance.

Definition 1. (Obstacle avoidance rule) An obstacle is defined as the one that is the nearest to the robot at one moment. In other words, only the nearest one is taken as the obstacle if there are many objects around the robot.

Definition 2. (Collision avoidance method) A virtual point is set to avoid collision. If the distance between the robot and the obstacle is smaller than a preset value, the robot will turn back and move away. As shown in Fig. 4, the black rectangle, the black circle and the white hole circle denote the robot, the obstacle, and the virtual object, respectively.

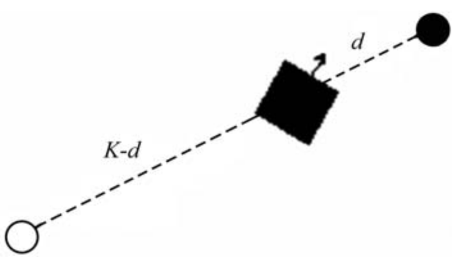


Fig. 4 Virtual point design

In terms of obstacle avoidance, the robot must detect and identify the obstacle moving around it. In addition, it also needs target localization so as to reach the destination (the opponent goal). Therefore, the input information can be defined as follows

$$R = \{e_{do}\ \dot{e}_{do}\ e_{dv}\ \dot{e}_{dv}\ e_{\theta v}\ \dot{e}_{\theta v}\ e_{dt}\ \dot{e}_{dt}\ e_{\theta t}\ \dot{e}_{\theta t}\}$$

where e_{do} is the distance between the robot and the obstacle, $\dot{e}_{do}$ denotes whether the robot is close to the obstacle or not. e_{dv} is the distance between the robot and the virtual point, $\dot{e}_{dv}$ describes whether the robot is near the virtual point or not. $e_{\theta v}$ is the angle between the robot positive direction and the virtual point direction, $\dot{e}_{\theta v}$ indicates the angle change between the robot and the virtual point. Similarly, e_{dt}, $\dot{e}_{dt}$, $e_{\theta t}$, and $\dot{e}_{\theta t}$ express the relative situations between the robot and the target.

As the change of relative positions between the obstacle and the robot can indicate whether the robot is far away from the obstacle or not, $e_{d0} > d_0$ and $e_{do} \leqslant d_0$ can be used to describe this situation. $|e_{\theta t}| > \theta_{c2}$ and $|e_{\theta t}| \leqslant \theta_{c2}$ are defined to describe whether the robot is towards the target or not. Therefore, we can get the following characteristic primitives:

$$Q_1\left\{\begin{array}{l} q_1|e_{do} \geqslant d_0 \\ q_2|e_{do} < d_0 \\ q_3|\dot{e}_{do} > 0 \\ q_4|\dot{e}_{do} \leqslant 0 \\ q_5|\dot{e}_{dv} > 0 \\ q_6|\dot{e}_{dv} \leqslant 0 \\ q_7|\,|e_{\theta v}| > \theta_{c1} \\ q_8|\,|e_{\theta v}| \leqslant \theta_{c1} \\ q_9|\dot{e}_{\theta v} > 0 \\ q_{10}|\dot{e}_{\theta v} \leqslant 0 \end{array}\right\} Q_2\left\{\begin{array}{l} q_1|\overline{e_{do} < d_0 \cap \dot{e}_{do} \leqslant 0} \\ q_1|e_{dt} > d_1 \\ q_2|e_{dt} \leqslant d_1 \\ q_3|\dot{e}_{dt} > 0 \\ q_4|\dot{e}_{dt} \leqslant 0 \\ q_5|\,|e_{\theta t}| > \theta_{c2} \\ q_6|\,|e_{\theta t}| \leqslant \theta_{c2} \\ q_7|\dot{e}_{\theta t} > 0 \\ q_8|\dot{e}_{\theta t} \leqslant 0 \end{array}\right\}.$$

The characteristic mode Φ, which describes the seven situations relative to the robot, the obstacle and the target, is finally obtained by combination of characteristic primitives, followed by

$$\Phi_1 = \left\{\begin{array}{l} \phi_1 : e_{do} < d_0 \cap \dot{e}_{do} \leqslant 0 \cap \dot{e}_{dv} > 0 \\ \phi_2 : e_{do} < d_0 \cap \dot{e}_{do} \leqslant 0 \cap \dot{e}_{dv} \leqslant 0 \cap e_{\theta v} > \theta_{c1} \\ \phi_3 : e_{do} < d_0 \cap \dot{e}_{do} \leqslant 0 \cap \dot{e}_{dv} \leqslant 0 \cap e_{\theta v} \leqslant \theta_{c1} \end{array}\right\}$$

$$\Phi_2 = \left\{\begin{array}{l} \phi_4 : \overline{e_{do} < d_0 \cap \dot{e}_{do} \leqslant 0} \cap \dot{e}_{dt} > 0 \\ \phi_5 : \overline{e_{do} < d_0 \cap \dot{e}_{do} \leqslant 0} \cap \dot{e}_{dt} \leqslant 0 \cap |e_{\theta t}| > \theta_{c2} \\ \phi_6 : \overline{e_{do} < d_0 \cap \dot{e}_{do} \leqslant 0} \cap e_{dt} > d_2 \cap \dot{e}_{dt} \leqslant 0 \cap \\ \qquad\qquad |e_{\theta t}| \leqslant \theta_{c2} \\ \phi_7 : \overline{e_{do} < d_0 \cap \dot{e}_{do} \leqslant 0} \cap e_{dt} \leqslant d_2 \cap \dot{e}_{dt} \leqslant 0 \cap \\ \qquad\qquad |e_{\theta t}| \leqslant \theta_{c2} \end{array}\right\}.$$

Based on actual control need and two control units of the robot, i.e., the linear velocity u_v and the angular velocity u_w, the control mode is finally deduced as follows:

$$\Psi_1 = \left\{\begin{array}{l} \psi_1 : u_v = K_1, u_w = k_1\,\mathrm{sign}(e_{\theta v}) \cdot pi \\ \psi_2 : u_v = K_2, u_w = k_2\,\mathrm{sign}(e_{\theta v}) \cdot pi \\ \psi_3 : u_v = V_{\max}, u_w = k_3 \cdot e_{\theta v} + k_4 \cdot \dot{e}_{\theta v} \end{array}\right\}$$

$$\Psi_2 = \left\{\begin{array}{l} \psi_4 : u_v = K_3, u_w = k_5\,\mathrm{sign}(e_{\theta t}) \cdot pi \\ \psi_5 : u_v = K_4, u_w = k_6\,\mathrm{sign}(e_{\theta t}) \cdot pi \\ \psi_6 : u_v = V_{\max}, u_w = k_7 e_{\theta t} + k_8 \dot{e}_{\theta t} \\ \psi_7 : u_v = k_9 e_{dt}, u_w = k_{10} e_{\theta t} + k_{11} \dot{e}_{\theta t} \end{array}\right\}.$$

Finally, the planning process of obstacle avoidance can be obtained with respect to the one to one mapping relationship between Φ and Ψ. Obstacle avoidance includes two behaviors with seven modes. One behavior is to avoid the obstacles, while the other one is to move toward the target. The process of planning is described as follows.

1) If ϕ_1 then ψ_1: the obstacle is gradually close to the robot. In order to avoid collision, it is necessary for the robot to turn back and move toward the virtual point. In addition, the robot will go along the arch owing to the two-wheel driving mechanism. Therefore, a smaller linear velocity and higher angular velocity enable the robot to turn back quickly.

2) If ϕ_2 then ψ_2: the robot just turns back with a higher angular error to the virtual point. Meanwhile, the obstacle is closer to it. Thus, the linear velocity should be increased to avoid collision between the robot and the obstacle.

3) If ϕ_3 then ψ_3: the robot has already aimed at the virtual point and is opposite the obstacle, thereby it should go ahead with full speed to keep away from the obstacle.

4) If ϕ_4 then ψ_4: the robot has broken away from the obstacle and is against the target, so it needs a lower linear velocity and higher angular velocity than before to turn back.

5) If ϕ_5 then ψ_5: the robot has gotten rid of the obstacle and moves toward the target, but the angular error is so big that it needs to increase its linear velocity.

6) If ϕ_6 then ψ_6: the robot has already aimed at the target but is far away, thereby it needs a higher linear velocity than before to move with full speed and control simultaneously the angular error within a certain range.

7) If ϕ_7 then ψ_7: when being close to the target, the robot needs to control the deviation of linear velocity and angular velocity to realize smooth action.

As discussed above, because the robot changes its position and speed all the time depending on the actual state, it will react quickly and move smoothly.

3.4 Target tracking

Similar to obstacle avoidance, the target tracking process is decomposed into 6 modes.

1) If $e_d > d_1 \cap |e_\theta| > \theta_{c1}$ then $u_v = k_1$, $u_w = k_2\ \mathrm{sign}(e_\theta)\cdot pi$

A low linear velocity and high angular velocity are applied to make the robot turn back, when the robot is opposite to the target and far away from it.

2) If $e_d \leqslant d_1 \cap |e_\theta| > \theta_{c1}$ then $u_v = k_3$, $u_w = k_4 e_\theta$

When the robot is opposite to the target and closer to it, if the velocity of the robot is increased, owing to the movement mechanism and wheel speed following feature, then the robot will be far away from the target. Therefore, a lower linear velocity and a higher angular velocity should be applied to the robot.

3) If $\theta_{c2} < |e_\theta| \leqslant \theta_{c1}$ then $u_v = k_5, u_w = k_6 e_\theta + k_7 \dot{e}_\theta$

Because the robot moves toward the target with significant angle deviation, its linear velocity can be increased at this time.

4) If $\dot{e}_d > -c \cap |e_\theta| \leqslant \theta_{c2}$ then $u_v = k_8, u_w = k_9 e_\theta + k_{10} \dot{e}_\theta$

At this moment, the robot has aimed at the target while its velocity is decreased, i.e., the robot follows the target, so the robot should be accelerated immediately.

5) If $e_d \geqslant d_2 \cap \dot{e}_d \leqslant -c \cap |e_\theta| \leqslant \theta_{c2}$ then $u_v = V_{\max}$, $u_w = k_{11} e_\theta$

The robot is far away from the target and aims at it. Therefore, the robot should go ahead with its maximal velocity.

6) If $e_d < d_2 \cap \dot{e}_d \leqslant -c \cap |e_\theta| \leqslant \theta_{c2}$ then $u_v = k_{12} e_d, u_w = k_{13} e_\theta + k_{14} \dot{e}_\theta$

At this situation, the robot is close to target with high speed, it needs to reduce the control parameter in order to smoothen the action.

3.5 Parameter confirming

There are many parameters in target tracking and obstacle avoidance. For example, the obstacle avoidance has 4 parameters (d_0, d_1, θ_{c1}, θ_{c2}) in perception module and 16 parameters ($K_1, \cdots, K_4$, $V_{\max}$, k_1, $k_2, \cdots, k_{11}$) in action module, while the target tracking has 5 parameters ($d_1, d_2, c, \theta_{c1}, \theta_{c2}$) in perception module and 15 parameters (k_1, $k_2, \cdots, k_{14}$, $V_{\max}$) in action module. How to determine their values is crucial and difficult. The trial and error method is usually used to determine these parameter values, and successfully applied in simple problems. However, its effects totally depend on expert knowledge and whether the determined values are the optimal ones or not.

In recent years, the genetic algorithms (GAs) has been applied in many fields, which is the adaptive and heuristic search algorithms based on the evolutionary ideas of natural selection and genetics[23,24]. In this paper, we adopt the GA to obtain the parameter values.

Traditional GA can be conducted as follows: first, an initial population is randomly generated from the encoded parameters in a chromosome; then, a new offspring is produced through selection, crossover, and mutation operators; finally, a satisfying solution is found. In order to acquire a higher quality of initialized solution and searching efficiency with sufficient diversity of population and no premature convergence, the traditional GA is improved by the following six steps.

Step 1. Generation of the initial population through mixed encoding of integral and real method. A chromosome can be generated by $x_i = l_i + \beta(u_i - l_i)$, where l_i and u_i are the upper and lower limits of gene i in the solution space. $\beta \in \{0, 0.1, 0.2, \cdots, 1\}$ is a random number. With this method, a higher quality of initial population can be gained.

Step 2. Selection with roulette wheel and definition of the Hamming distance between two chromosomes to avoid close relatives. The Hamming distance of two chromosomes is defined as

$$D_{\mathrm{Hamming}}(x, y) = \sum_{i=1}^{M} d_i, \quad d_i = \begin{cases} 1, & \text{if } x_i \neq y_i \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where x and y are two chromosomes, and d_i describes the different genes between x and y.

If $D_{\mathrm{Hamming}}(x, y) < 1$, these two chromosomes selected by roulette wheel will not generate a new chromosome through crossover operators. Two strategies are usually selected to avoid this case. When $D_{\mathrm{Hamming}}(x, y) = 0$, the first strategy is adopted to generate a new chromosome through mixed encoding of these two chromosomes to replace one of them. The second one is to select another gene j as a new cut point when $D_{\mathrm{Hamming}}(x, y) = 1$.

Step 3. Crossover with alterable precision for enhancing the variety of population. If current generation is singular, the 1-point crossover will be selected. Otherwise, the uniform crossover will continue.

Step 4. Generating an optimal chromosome through the orthogonal experiment and crossover between two chromosomes. Because of the introduction of the Hamming distance, the orthogonal experiments only act on the different parts between two chromosomes.

Step 5. Mutation by dynamic encoding to avoid premature convergence.

Let $x_{\mathrm{optimal}} = \{x_1, x_2, \cdots, x_i, \cdots, x_M\}$ be the best chromosome in the current population. The dynamic encoding will generate feedback mutation according to the following steps:

Step 5.1. Produce a random number ζ ($\zeta \in [0, 1]$).

Step 5.2. If $\zeta < \tau$ then using $y_i = \mu \cdot x_i + \beta \cdot \nu \cdot x_i$ to generate a new chromosome $y = \{y_1, y_2, \cdots, y_i, \cdots, y_M\}$ and replace the unsuitable one. Both μ and ν are constant and smaller than 1. If $\zeta \geqslant \tau$ then jump to Step 5.3.

Step 5.3. Generate a new chromosome through mixed encoding between two chromosomes, and substitute it for the one whose fitness is worse.

Step 5.4. Judge the suitability of the new produced chromosome. If it does not meet the needed number, jump to Step 5.1. Otherwise, jump to Step 5.5.

Step 5.5. End.

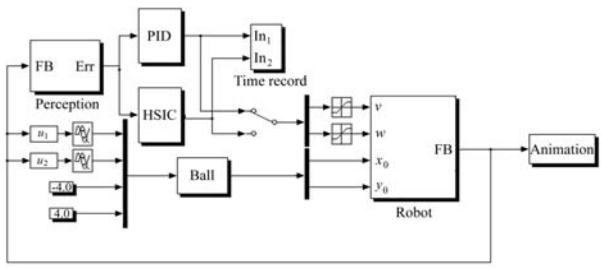
Step 6. Sort the parents and children according to fitness to create a new generation population.

4 Experiment analysis

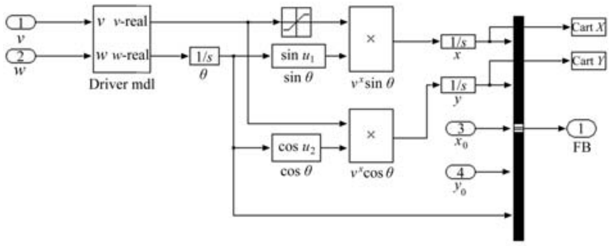
4.1 Simulation condition

In order to verify the algorithm presented above, some experiments were designed in a simulated environment and actual match. In simulation experiment, we used Matlab 6.5 software to realize the robot modeling and the HSIC controller. We constructed a match court of $8\,\mathrm{m} \times 6\,\mathrm{m}$ as the reduced actual court, and took its centre as the origin of the two-dimensional rectangular coordinate system whose axes were along its right and top directions. The unit was set as follows: the distance unit is a pixel, the angle unit uses degrees, the linear velocity unit is millimeter per second, and the angular velocity unit is degree per second. For simplicity, we used solid and hollow circles to denote the obstacle (or ball) and the robot, respectively.

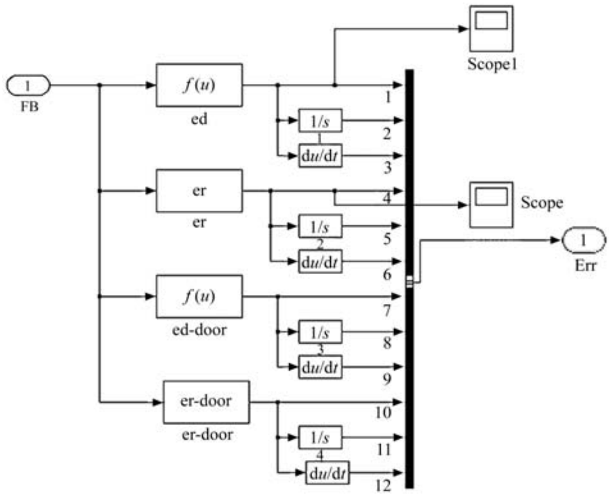
We designed the system structure in simulation environment, as shown in Fig. 5. In Fig. 5, "FB" denotes feedback block for information, "Err" denotes error, "mdl" denotes model defining link (actually it is defining for continuous system), "ed" denotes error in distance, and "er" denotes error revising in angle. This system includes four modules, i.e., perception module, robot module, animation module, and object control module. The animation module mainly displays the moving process of all objects, e.g., the robot, the ball, and the obstacles. The robot module is used to realize the kinematic model introduced in Section 2, as shown in Fig. 5 (b). The perception module deals with the input information and calculates the error, as shown in Fig. 5 (c). The object control module may control the object movement through different algorithms such as the PID algorithm. We will analyze the obstacle avoidance and target tracking in detail.



(a) Simulation of the total system



(b) Simulation of robot



(c) Simulation of perception

Fig. 5 System design in simulation environment

4.2 Obstacle avoiding

In the simulation environment, the robot started from the court centre and aimed at the goal. We designed two dynamic obstacles with a changeable speed. When the obstacle is far away from the robot, it would move quickly. To test the obstacle avoidance ability of the robot, we placed the obstacles at different positions and recorded its responses. Fig. 6 shows the recorded results when the obstacles were placed at different positions.

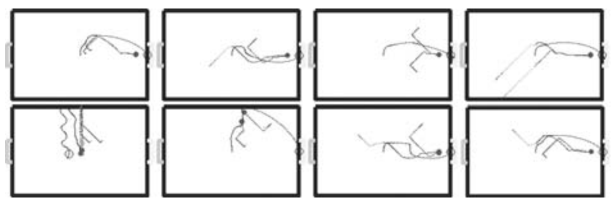


Fig. 6 Obstacle avoidance in simulation environment

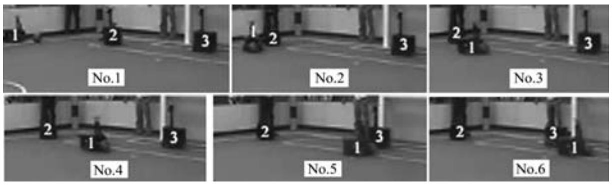
As shown in Fig. 6, the obstacles would pursue the robot all the time, thereby the robot had to change its control strategy to avoid the obstacles at any moment and finally reach the goal.

After testing the validity of the simulation experiments, the HSIC method was applied to a real robot soccer match. Through the GA, we could obtain the perception parameters as follows: $d_0 = 80, d_1 = 50$, $\theta_{c1} = 25$, and $\theta_{c2} = 30$. The action parameters are also listed as follows: $K_1 = 100$, $K_2 = 1000$, $K_3 = 500$, $K_4 = 1000$, $V_{\max} = 1800$, $k_1 = 0.7$,

$k_2 = 0.8$, $k_3 = 0.9$, $k_4 = 0.02$, $k_5 = 0.7$, $k_6 = 0.8$, $k_7 = 0.9$, $k_8 = 0.02$, $k_9 = 12$, $k_{10} = 0.9$, and $k_{11} = 0.6$.

Some representative snapshots, which came from the actual match video, are presented in Fig. 7. The twelve images were obtained from an actual game video.

(a) Multi-obstacles avoidance



(b) Two obstacles avoidance

Fig. 7 Examples of obstacles avoidance in an actual match

As shown in Fig. 7, mark 1 denotes our robot controlled by HSIC method. The other marked robots can be regarded as the obstacles. Fig. 7 (a) shows that our robot (mark 1) crosses three obstacles and passes around a static obstacle (mark 5). When our robot is close to the opposite goal, it holds the ball tightly and moves toward the goal in order to avoid the challenges from the opponent robot. From Fig. 7 (b) it can be seen that our robot first passes around the opponent robot (mark 2), then goes around the goalkeeper (mark 3). Finally, it moves toward the goal.

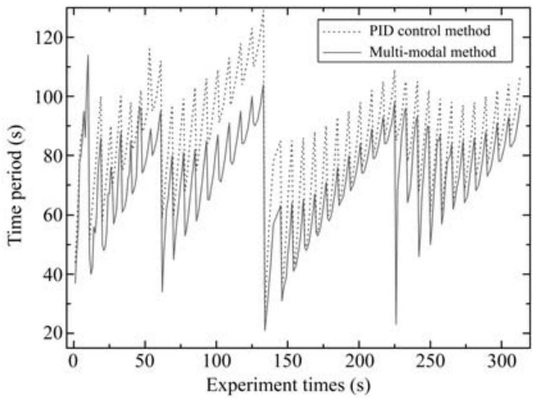
4.3 Target tracking

In the simulation experiment, the ball was set as a target, and the PID controller and the HSIC controller were designed for target tracking action control, respectively. Parameters of the PID controller were optimized. During the experiment, we put the ball at different positions then release it with a fixed speed along the positive direction. At the same time, the robot started from the coordinate origin. $(+, +)$ denotes that the robot tags after the target, $(+, -)$ means that the robot tracks head-on towards the target in the positive direction, while $(-, +)$ and $(-, -)$ indicate that the robot pursues the target face to face along the negative direction.

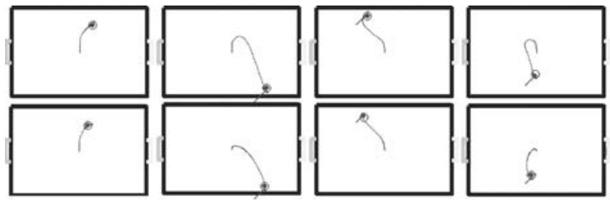
We performed the experiments 313 times by setting the ball at different positions. Table 1 illustrates nine representative experiments. Fig. 8 shows the simulation results.

Fig. 8 (a) is the contrast of times in 313 experiments by using both PID controller and HSIC controller. According to the different ball′s positions, the tracks of PID and HSIC are shown in the top and bottom insets of Fig. 8 (b), respectively. Fig. 8 (c) illustrates the process in which the robot pursues the ball and takes it into the goal.

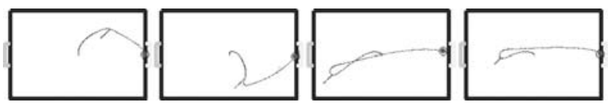
From Table 1 and Fig. 8 (a), it can be found that the HSIC algorithm spent less time than the PID algorithm in simulation experiment, independently of the direction and position of the target. Moreover, though the absolute distances were the same, the times spent on tracking the target were different. For example, when the positions of the target were $(1, 1)$, $(1, -1)$, $(-1, 1)$, and $(-1, -1)$, the times were 58, 65, 38, and 80 for the PID method and 50, 51, 35, and 66 for the HSIC method, respectively. Fig. 8 (b) also shows that the track of HSIC method was smoother than that of PID method. Fig. 8 (c) shows that the robot first captured the ball then went toward the goal along the optimal path by the HSIC method.



(a) Contrast of times for PID and HSIC



(b) Moving tracks of different states



(c) Moving tracks of robot based on HSIC

Fig. 8 The results of simulation

Table 1 Contrast of time periods

	No. 1	No. 2	No. 3	No. 4	No. 5	No. 6	No. 7	No. 8	No. 9
Coordinate (m)	0.5,1.0	0.5, −1.0	−0.5, 1.0	−0.5, −1.0	1.0, 0.5	0.5,4.0	−1.0, −2.5	−2.0,1.5	3.5,−1.5
New method (ms)	47	46	26	68	48	74	83	49	78
PID (ms)	50	64	38	87	54	88	90	52	102

Fig. 9 illustrates an example of the target tracking process in the actual match. In Fig. 9, the six images are obtained from an actual game video. Mark 1 is our robot, mark 2 is a robot that can be seen as a static obstacle, mark 3 is the opponent robot with the ball, and marked 4 is our goalkeeper. When our robot became aware of the threat of our goal from the opponent robot, it moved toward our goal and held up the opponent robot.

From Fig. 9, we can see the total process: first, our robot perceived that the opponent robot would attack our goal, as shown in inset No. 1; then it quickly moved towards our goal (see inset No. 2); because there was a static obstacle (mark 2) at that time, it decided to go around this obstacle, as shown in inset No. 3; when our robot was gradually close to our goal and perceived some threats from the opponent robot at the left, it turned back (see inset No. 4) and ran quickly towards the opponent robot to hold up this robot (see insets No. 5 and No. 6).

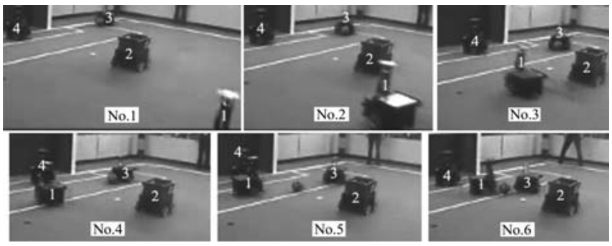


Fig. 9 The target tracking example in actual match

From the comparative results of the simulation and actual experiments, it can be concluded that the HSIC method is more suitable for a rapid reaction and a smoother track.

5 Conclusions

The bottom action control is very important to soccer robots. Especially, when the bottom hardware cannot be changed, the control algorithm is crucial for rapid reaction and optimal track. In this paper, we mainly discuss the architecture design of bottom action control, which includes the design of obstacle avoidance and target tracking. The HSIC method is inspired by human intelligence, which is performed in simulation and actual match separately. The results show that the HSIC method is effective in decreasing the execution time and optimizing the track of bottom action. At the same time, the comparative results also validate the superiority and advantages of the proposed method in this paper. However, the values of parameters are set offline. An adaptive parameter control mechanism is expected to do better, which will be investigated in the future.

Our team attended the 2006 RoboCup Chinese Public Competition in Changzhou, and gained the second prize in 4 versus 4 match and the third prize in 2 versus 2 match.

Acknowledgment

The authors thank Prof. Zu-Shu Li for his guidance in this paper. The authors also thank Fang Yu for her previous work on soccer robots.

References

[1] K. H. Low. Locomotion and depth control of robotic fish with modular undulating fins. *International Journal of Automation and Computing*, vol. 3, no. 4, pp. 348–357, 2006.

[2] A. De Cabrol, T. Garcia, P. Bonnin, M. Chetto. A concept of dynamically reconfigurable real-time vision system for autonomous mobile robotics. *International Journal of Automation and Computing*, vol. 5, no. 2, pp. 174–184, 2008.

[3] S. Ma, N. Tadokoro, B. Li, K. Inoue. Analysis of creeping locomotion of a snake robot on a slope. In *Proceedings of IEEE International Conference on Robotics and Automation*, IEEE, Taiwan, pp. 2073–2078, 2003.

[4] M. Friedmann, S. Petters, M. Risler, H. Sakamoto, D. Thomas, O. von Stryk. New autonomous, four-legged and humanoid robots for research and education. In *Proceedings of Simulation, Modeling and Programming for Autonomous Robots*, Springer, Venice, Italy, vol. 2, pp. 570–579, 2008.

[5] H. Fujii, K. Yoshida. Action control method for mobile robot considering uncertainty of information. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems*, IEEE, Alberta, Canada, pp. 3915–3920, 2005.

[6] L. Wong. Fault tolerant behavior-based robots. *Journal of Undergraduate Research*, vol. 4, no. 1, pp. 13–19, 2005.

[7] G. A. Kaminka, I. Frenkel. Flexible teamwork in behavior-based robots. In *Proceedings of the 20th National Conference on Artificial Intelligence*, American Association for Artificial Intelligence, USA, AAAI-05, pp. 108–113, 2005.

[8] W. Chatlatanagulchai, C. Srinangyam, W. Siwakosit. Trajectory control of a two-link robot manipulator carrying uncertain payload using quantitative feedback theory. *Journal of Research in Engineering and Technology*, vol. 5, no. 1, pp. 45–71, 2008.

[9] H. K. Gazit. Transforming High Level Tasks to Low Level Controllers, Ph. D. dissertation, University of Pennsylvania, USA, pp. 16–28, 2008.

[10] N. Ohnishi, A. Imiya. Corridor navigation and obstacle avoidance using visual potential for mobile robot. In *Proceedings of the 4th Canadian Conference on Computer and Robot Vision*, IEEE, pp. 131–138, 2007.

[11] S. Mastellone, D. M. Stipanović, C. R. Graunke, K. A. Intlekofer, M. W. Spong. Formation control and collision avoidance for multi-agent non-holonomic systems: Theory and experiments. *International Journal of Robotics Research*, vol. 27, no. 1, pp. 107–126, 2008.

[12] R. Nakanishi, J. Bruce, K. Murakami, T. Naruse, M. Veloso. Cooperative 3-robot passing and shooting in the RoboCup small size league. In *Proceedings of the RoboCup Symposium 2006, Lecture Notes in Computer Science*, Springer, Berlin, Germany, vol. 4434, pp. 418–425, 2009.

[13] G. Shao. Study on the Behavior, Behavior Selection and Evolution of Artificial Life Based on Sensor-motor Intelligent Schema, Ph. D. dissertation, Chongqing University, PRC, 2007. (in Chinese)

[14] M. Sato, A. Kanda, K. Ishii. A switching controller system for a wheeled mobile robot. *Journal of Bionic Engineering*, vol. 4, no. 4, pp. 281–289, 2007.

[15] M. W. Han, P. Kopacek. Neural networks for the control of soccer robots. *Computational Intelligence, Theory and Applications*, Berlin, Germany: Springer, vol. 33, pp. 621–628, 2005.

[16] J. Jang, C. K. Ahn, S. Han, W. H. Kwon. Rapid control prototyping for robot soccer system using SIM tool. In *Proceedings of International Joint Conference on SICE-ICASE*, IEEE, Korea, pp. 3035–3039, 2006.

[17] N. Ouadah, L. Ourak, F. Boudjema. Car-like mobile robot oriented positioning by fuzzy controllers. *International Journal of Advanced Robotic Systems*, vol. 5, no. 3, pp. 249–256, 2008.

[18] R. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, vol. 2, no. 1, pp. 14–23, 1986.

[19] Product Information Guide for Frontier-I Autonomous Mobile Robot. *Shanghai Jiao Tong University*, pp. 18–30, 2004.

[20] A. Coninx, A. Guillot, B. Girard. Adaptive motivation in a biomimetic action selection mechanism. In *Proceedings of Conference on Neurosciences Computation*, France, pp. 158–162, 2008.

[21] P. Guerrero, J. Ruiz-del-Solar, G. Díaz. Probabilistic decision making in robot soccer. *Lecture Notes in Computer Science, RoboCup 2007: Robot Soccer World Cup XI*, Springer, Berlin, Germany, vol. 5001, pp. 29–40, 2008.

[22] G. Shao, Y. Wen, F. Yu, Z. Li. Target localization for autonomous soccer robot based on vision perception. In *Proceedings of the 3rd International Symposium on Intelligence Computation and Applications, Lecture Notes in Computer Science*, Springer, Berlin, Germany, vol. 5370, pp. 831–840, 2008.

[23] R. M. M. Vallim, T. S. P. C. Duque, D. E. Goldberg, A. C. P. L. F. Carvalho. The multi-label OCS with a genetic algorithm for rule discovery: Implementation and first results. In *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, ACM, Montreal, Québec, Canada, pp. 1323–1330, 2009.

[24] C. D. Wu, Y. Zhang, M. X. Li, Y. Yue. A rough set GA-based hybrid method for robot path planning. *International Journal of Automation and Computing*, vol. 3, no. 1, pp. 29–34, 2006.

Tie-Jun Li received the B. Sc. and M. Sc. degrees in automation from the Chongqing University, PRC in 2000 and 2005, respectively. In 2005, he was a faculty member at Chongqing Jinmei Communication Co., Ltd., PRC. He is currently an engineer at Jimei University, PRC.

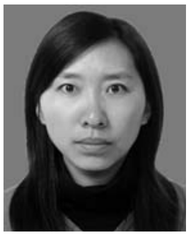
His research interests include robotics, communication systems, and electromagnetic compatibility.

E-mail: ltj_cqu@163.com

Gui-Qiang Chen received the B. Eng. degree in wireless communication from Chongqing Communication College, PRC in 2000. He received the M. Sc. degree in intelligent control and pattern recognition from Chongqing University, PRC. Now he is working with Chongqing Communication College.

His research interests include intelligent control, evolutional computation and pattern recognition.

E-mail: guiqiang_chen@163.com



Gui-Fang Shao received the B. Sc., M. Sc., and Ph. D. degrees from Chongqing University, PRC in 2000, 2003, and 2007, respectively, all in control theory and control engineering. She is currently an associate professor with the Institute of Pattern Recognition and Intelligence System, Xiamen University, PRC.

Her research interests include the pattern recognition, image processing, and robot control.

E-mail: gfshao@xmu.edu.cn (Corresponding author)