ELSEVIER

Robotics and Autonomous Systems 21 (1997) 149–165

Robotics and Autonomous Systems

Designing distributed control architecture for cooperative multi-agent system and its real-time application to soccer robot

Hyun-Sik Shim [a,*], Heung-Soo Kim [a], Myung-Jin Jung [a], In-Hwan Choi [a], Jong-Hwan Kim [a], Jin-Oh Kim [b,1]

[a] *Department of Electrical Engineering KAIST, Taejon-shi, 305–701, South Korea*
[b] *Samsung Electronics Co. Ltd., Suwon-shi, 442–742, South Korea*

Abstract

The soccer robot system consists of multi-agents, with highly coordinated operations and movements so as to fulfill specific objectives, even under adverse situations. The coordination of the multi-agent is associated with a lot of supplementary work in advance. The associated issues are the position correction, prevention of communication congestion, local information sensing in addition to the need for imitating the human-like decision making. A control structure for soccer robot is designed and several behaviors and actions for a soccer robot are proposed. The action selection mechanism (ASM) is proposed as a high-level controller and is applied to a one-on-one soccer game. The intervention module of ASM uses multilayer perceptrons and its effectiveness is shown by experimental results. Modified zone defense as a basic strategy and several special strategies for fouls as applied to SOTY and MIRO teams for MIROSOT are discussed. The authors have been placed in the second and fourth positions in MIROSOT'96 (Micro-Robot World Cup Soccer Tournament) held at KAIST in November 1996, with the proposed scheme and micro-robots.

Keywords: Soccer robot; Multi-agent system; MIROSOT; Micro robot

1. Introduction

From the standpoint of multi-agent systems, a soccer game is a good example of the problems in real world, which can be moderately abstracted. We have chosen soccer as one of the standard problems for the study on multi-agent systems, which can be used as a benchmark. Multi-agent systems deal with research subjects such as cooperation protocol by distributed control, effective communication and fault tolerance, while exhibiting efficiency of cooperation, adaptation, robustness and being in real-time.

Multi-agent systems (the control architecture, communication scheme, vision tracking algorithm, sensing and fault tolerance) have been studied by several research groups in recent years. An architecture called ALLIANCE is presented for fault tolerance in [12]. Cooperation of heterogeneous mobile robots and a collection of particle-like-small-cell-concept micro-robots, called cellular robots, was developed by Kawauchi et al. [3]. Mitsmoto et al. [10] developed micro-robots and proposed a self-organizing algorithm using immune network. Despite the superb idea of introducing a new concept in robotics called

* Corresponding author. E-mail: hsshim@vivaldi.kaist.ac.kr.
[1] E-mail: kj@rnd.sec.samsung.co.kr.

PII S0921-8890(97)00023-7

multi-agent systems, their experiments and applications [1,9,13] can have limited number of agents and a narrow field of application as well. Computational intelligences such as fuzzy system, neural networks, evolutionary computation and some machine learning algorithms like Q-learning profit sharing plan could be used for robot intelligence. Application of multi-agent system to a robot-soccer is a challenging problem since it needs a lot of supplementary work in advance – position correction, preventing communication congestion, and sensing local information – and it should imitate human-like decision making.

In this paper, three control schemes for soccer robot control are dealt with. The first scheme is a remote-brainless soccer robot system in which a host computer controls the robots by commanding robot's velocities like a radio-controlled car. The second scheme is a vision-based one which controls or manipulates the robots by processing the information from the vision with a supervisor processor and sends commands to the robots directly. So it is rather like a traditional supervisory system. The third scheme is a robot-based one, wherein each autonomous robot can make a decision based on information it collects with its sensors and if needed, and can communicate with other robots. Robots which can be operated autonomously are designed for the robot-based one. Among the above three schemes, only the vision-based soccer robot system has been dealt with, in this paper as it is easy to develop, though the robots are designed as autonomous ones for the robot-based system. A control structure, several behaviors and actions are proposed for a soccer robot. The action selection mechanism (ASM) is proposed as a high-level controller and applied to a one-on-one soccer game. The intervention module of ASM uses multilayer perceptrons and its effectiveness is shown by experimental results. Modified zone defense as a basic strategy and several special strategies for fouls as applied to SOTY and MIRO teams for MIROSOT are also discussed. A soccer game of the MIROSOT (Micro-Robot World Cup Soccer Tournament) [5] played by three robots from each team is used as a testbed of the multi-agent systems. The authors have been placed in second and fourth positions in MIROSOT'96 held at KAIST in November 1996, with the proposed scheme and micro-robots.

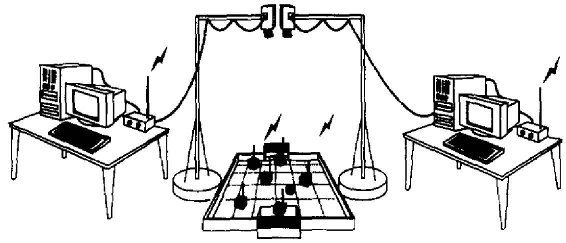


Fig. 1. Overview of the soccer robot system with a vision system, a host computer and an RF communication system.

2. Soccer robot system

There are various methods to make a robot soccer system for MIROSOT. In addition to the varieties in hardware like CPU, actuators, sensors and so forth, they also differ in software for the control algorithms, the strategies and the total integration system. Therefore, the selection of what kind of hardware and software among various methods is a challenging problem to the robot designers. This paper first deals with the method to develop a soccer robot system and then describes its hardware and software.

Basically, robots, a vision system, a host computer and a communication system are needed for a robot soccer game as in Fig. 1. Three operating methods are considered for a soccer game using this system: remote-brainless soccer robot system, vision-based soccer robot system and robot-based soccer robot system.

2.1. Remote-brainless soccer robot system

In this system, each of the robots has its own driving mechanism, communication part and CPU board. The computational part controls robot's velocity according to command data received from a host computer. All calculations of vision data processing, strategies, position control of robots and so on are done in the host computer which controls robots like a radio controlled car. The robot can be implemented easily as its structure is simpler than that of the other systems. Since it has no sensors like encoders, the calculating power of the host computer has to be higher compared with other systems for controlling positions of robots accurately. Generally, the position control of mobile robots is done by an embedded processor using its encoders or sensors, but it has to be carried out by the host computer in this case. To control the positions of robots accurately, the sampling time of the host computer must be very small. Vision processing is a key technology, especially, in this system. Because all algorithms of the remote-brainless soccer robot system are centralized in the host computer, the communication protocol for multi-agent cooperative system is very simple. But the problem is that the burden of the host computer will get multiplied with the increase in the number of agent.

2.2. *Vision-based soccer robot system*

The vision-based system can be considered as a system at an intermediate level between the remote-brainless and the robot-based system. In the vision-based system, the robots should have functions such as velocity control, position control, obstacle avoidance and so on. The host computer processes vision data and calculates next behaviors of robots according to strategies and sends commands to the robots using a RF modem. The robots make their moves according to these commands keeping away from obstacles autonomously. The robots have to have sensors for position control (encoders) and obstacle avoidance (IR sensors). The calculation load of the host computer is much less compared with the former system.

2.3. *Robot-based soccer robot system*

In the robot-based soccer robot system, each robot has many functions for autonomous behaviors. All calculations are done locally in each of the robots. The host computer processes vision data on the position of the ball and robots and forwards the same to the robots. Each of the robots decides its own behavior autonomously using the received vision data, its own sensor data and strategies. This can be considered as a distributed control system, where each robot has its own intelligence. It may be very hard to implement the robot with that established for velocity control, position control, obstacle avoidance, communication, decision making, etc. The host computer processes only vision data and can be considered as a kind of sensor.

2.4. *Designer's choice*

A designer may choose one of these methods according to the research interests. Researchers interested in vision processing, artificial life application, etc. may prefer the first method and those interested in mobile robot control, intelligent system application and multi-agent cooperative system may go for the second or third methods. Among the three methods, the first method is easy to implement as a soccer robot system, but, in the long run, the third method should be researched in depth. The robots of MIRO and SOTY teams are designed for the vision-based and the robot-based systems for the research purpose. However, in MIROSOT'96, we employed the vision-based system to compete with other teams because its development is easier than the robot-based one.

3. Soccer robot

The design philosophy is to realize autonomous micro-robots with various functions so that they can be used not only in soccer game but in a lot of other applications as well. The designed soccer robot has no body frame, but has six PCBs with surface mounting devices, as its sides (Fig. 2). CMOS-type devices are used considering power consumption. The complete robot's size is within $(7.5\,\text{cm})^3$. The CPU, eight side

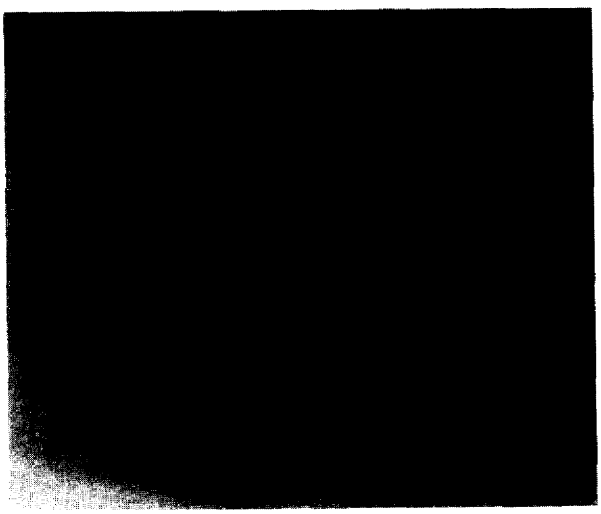


Fig. 2. The soccer robot.

IR sensors, eight bottom IR sensors and the caterpillar moving mechanism make it possible to control the robot's motion with obstacle avoidance and other intelligent behaviors.

It is designed in such a way that it will be useful in other applications as well.

3.1. CPU board

To realize high-intelligence, a high-performance CPU is needed. But, considering the limitation on size and power consumption, a one-chip controller with versatile functions found to be suitable. The more the computation of capability and the number of sensors, the more will be goals it achieved autonomously. It is tempting to use more CPUs, but the number is restricted by the size limitation of the robot.

The Intel 16 bit processor 80C196KC (20 MHz) is thus judiciously selected, which has useful functions such as PWM generators and communication modes for multi-CPUs expansion.

3.2. Sensors

The following are the minimum information each robot should gather by itself: (1) the location of the robot in the field; (2) the location of the ball; (3) whether the robot has the ball or not; (4) there is a blocking robot when kicking the ball.

There are several methods for each robot to reckon its position. The first one is using odometers, such as encoders and gyroscopes. The grids on the field and the wall help the robots to correct the accumulated errors. Another method, somewhat indirect, but a useful way is done by a vision camera. Despite its usefulness of getting precise location and orientation of the objects on the field, its relatively long processing time inhibits the user from using the vision system alone. A piezoelectric gyroscope and a digital compass are useful for correcting errors.

Each robot has two on-board encoders which make it possible to calculate its current position. The resolution is 0.4689 mm suitable enough for a small robot's motion control. The errors from these dead-reckoning sensors are corrected by detecting the grid lines on the field. Eight IR sensors specially attached at the bottom board are used for this purpose.

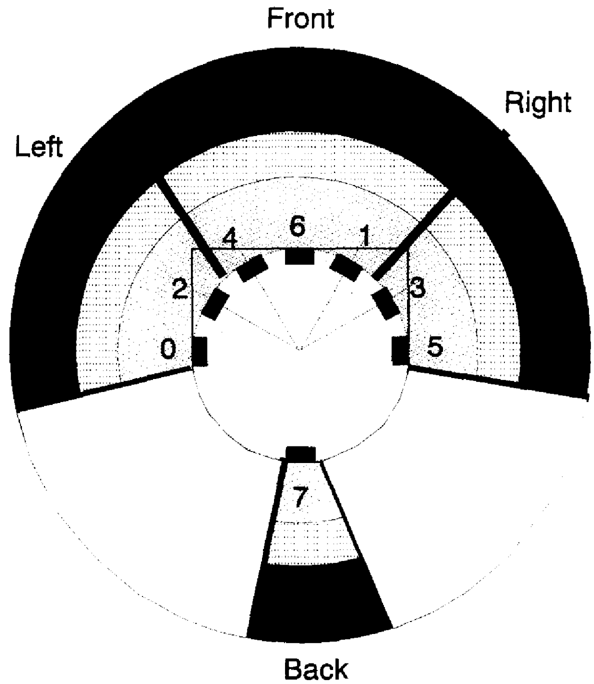


Fig. 3. Sensors for obstacle detection and their ranges.

Other eight IR sensors on the sides are for obstacle detection (Fig. 3). Unlike the conventional method that measures the amount of light [11], the modulated carrier signal is hired to be insensitive to environment light conditions. The obstacles in such discrete distances as 20 (very far), 10 (far), 5 cm (close) can be detected by just one pair of sensors. Linguistic sensing makes it easy to realize artificial intelligence approach.

3.3. Actuators and power unit

To make small robots, it is necessary to get the one chip motor controller along with the motors. Torque, speed and power consumption characteristics are important in choosing the motors.

Among the various methods of locating motors on board, it is popular to fix the two motors with their axes aligned, with two casters in the front and rear sides. If this is difficult due to lack of space, two motors with right-angled wheels [11] or with parallel wheels but not aligned [7] might be used. For smooth dribbling, aligning the wheels or adding the third motor for steering is recommended. Considering calculation loads on CPU, this paper only deals with kinematics [14] because the robot dynamics [15] is very complicated for the CPU to process.

The robot must carry the power source and rechargeable batteries. The power, the length of the competing time, the size and weight are the key factors in choosing the batteries. 9 and 4.8 V Ni–Cd re-chargeable batteries are used for the motors and the logic power, respectively. The average power consumptions are about 2 and 2.7 W, respectively.

3.4. Communication

According to MIROSOT rules, robots must not be wired, so IR-remote control or RF-digital communication system is necessary. In IR-remote control method, the circuitry is relatively simple when the number of channels is small and considerably complex. The robot uses the full duplex RF-digital communication. As the RS-232C data transmission uses a baseband without modulation, it uses an FM modulation. The communication between robots facilitates performing the team's objectives. Communication deadlock may happen when the amount of information increases, so communication strategy and protocol should be carefully designed.

Fig. 4 shows the communication protocol used to command robots from the host computer through RF modem for the vision-based soccer robot system. Three bytes are assigned for each robot of which two bytes are used for movements in x and y directions. The third byte is used as a flag representing 'robot's velocity', 'direction', 'straight or turn', 'enable obstacle avoidance' and 'exit from current action'.

3.5. External perception system

Because of the size limitation, the robots may not show fully autonomous behaviors with the current technologies. Considering this point, MIROSOT also allows a human operator to give the robots additional information such as locations of the ball and of other robots, through the host computer. However, the human operation will be slower than with the electric vision system. The vision system [13] will be helpful to improve the performance and the intelligence of the total system.

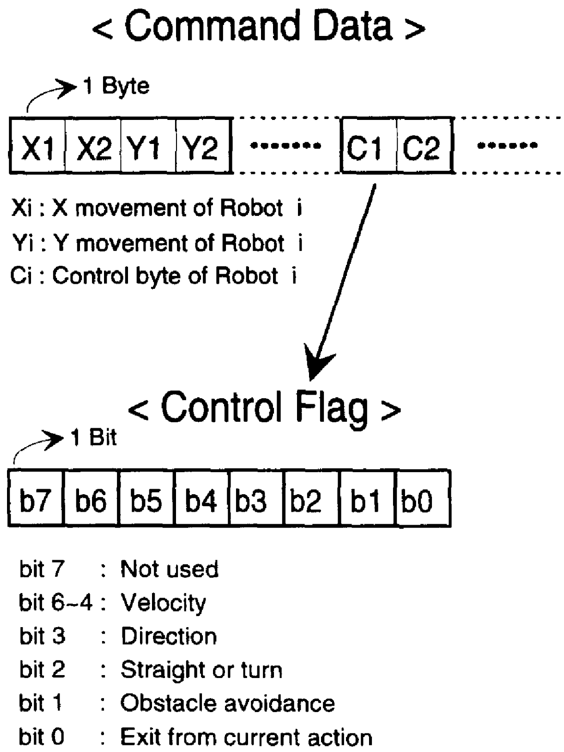


Fig. 4. Communication protocol for vision-based soccer robot system.

In the vision-based soccer robot system, it should have the capabilities to do some processing on strategies as well as for calculating locations of the ball and robots.

3.5.1. Position calculation of the ball and robots using a color vision system

For object detection using a vision system, an edge detection or a gray level detection method can be used. But, if there are objects with similar shapes or brightness of the object being detected, it becomes difficult. In that case, the color information of the object can make the detection easier. Specifically, if the target object has a certain uniform color, its information on position and direction under the given environment can be calculated by the color information through the vision system, for example, using the RGB values (see Fig. 5). Based on this method, the directions and positions of the orange ball and the robots with the unique color pattern can be calculated. The detection

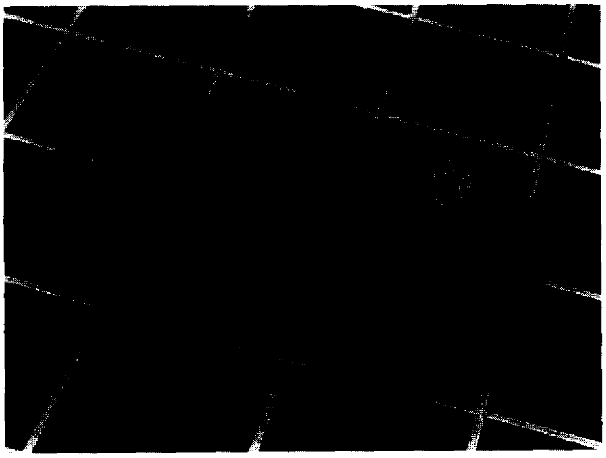


Fig. 5. Uniform colors on soccer robots.

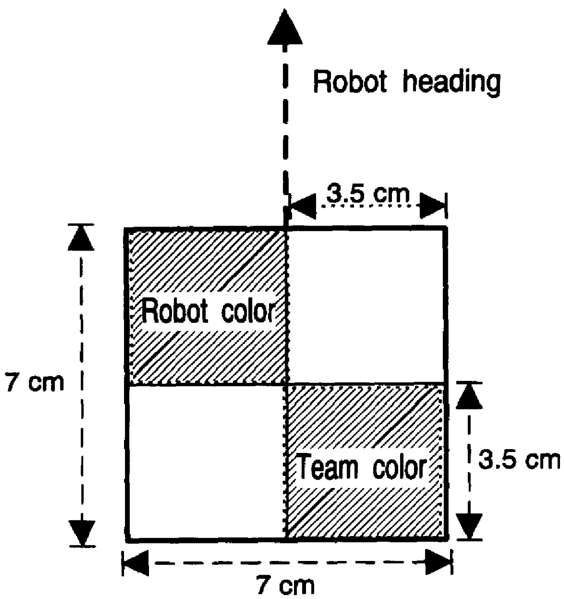


Fig. 6. Robot uniform according to the MIROSOT rule.

algorithm used here is a pixel search method, reducing the processing time and increasing the sampling rate under the given color vision system.

3.5.2. The algorithm for object detection

A robot uniform is shown in Fig. 6. Its pattern conforms to the rule on the robot uniform in MIROSOT. The right bottom square is a team color which must be blue or yellow. The left top one is the robot color which is freely assigned not to be confused with the team color. We need to classify six colored objects consisting of two team colors, three robot colors and one ball color (orange). The detection method is very simple. First, RGB values for each pixel are read to host memory from the frame buffer of the vision board. Then, the pixel is normalized by the sum of the three components and examined for satisfying the boundary conditions of RGB for the six colors. If there is any color satisfying the pixel, the X and Y coordinate values of the pixel are stored for that color and its pixel count is increased by 1. In this way, the whole pixels on the soccer field are searched and classified to one among the six colors whenever the condition is satisfied. Finally, the averages of X and Y summations for each color are calculated. To get the robustness from the pixel noise, the appropriate grouping method is required. For example, if one pixel is far away from other pixels satisfying the same conditions, it is regarded as the pixel of another object. At present, the processing time is about 150 ms because of supporting software libraries. If the memory is accessible directly, it can be reduced. Just using normalized RGB values is very simple, but, the simplicity makes our detection algorithm sensitive to the environment conditions such as illumination. To solve this problem, the detection method must be studied with more complex schemes such as HSI color space, pattern recognition and fuzzy classification.

4. Control structure

Fig. 7 shows a basic control structure of the soccer robots. Robot hardware architecture is described in the dotted line box, and the robot software is represented in a solid line box. Each robot has the same control structure except the specialized intelligence part. The specialized intelligence part consists of each robot's own special behavior or strategy. The basic behaviors are "move", "obstacle avoidance" and the basic actions are "shoot", "position_to_shoot", "intercept_ball", "sweep_ball" and "block". These are important behaviors and actions in a robot soccer game. Though the strategy is good, if moving ability is poor, the system is useless. A central controller [2,4] carries out practical strategies including the above basic behaviors and actions, by selecting an action mode based

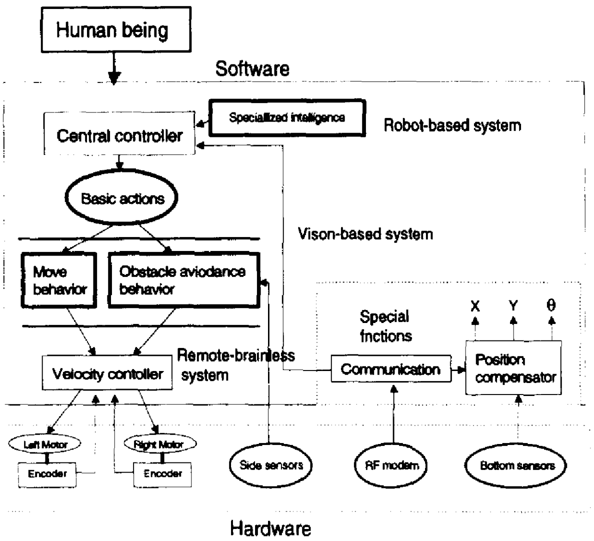


Fig. 7. Control structure of the soccer robot.

on present states, locations of the ball and robots. It also determines proper tactics according to specialized intelligence. The lowest-level controller in the control structure is a velocity controller with the shortest sampling time. As it goes to a higher level, the sampling time is longer. Especially, the sampling time of the central controller is determined by the data update time from the vision system. Other special functions of communication with the host computer and correcting the positioning errors are also incorporated.

4.1. Basic behaviors

4.1.1. Move behavior

Move behavior consists of two operations performed sequentially, rotating and running (Fig. 8). When the operator or the host computer informs the robot of the coordinates of its destination, the robot rotates and runs toward that point. Since a small orientation error leads to a larger location error, a 16-bit float type variable is declared for the orientation. A method to align the destination point with the robot orientation is to calculate the inner product between the robot orientation vector and the destination vector (from the robot position to the destination) and to determine whether this value is within the desired boundary. Since we have made a table of one degree

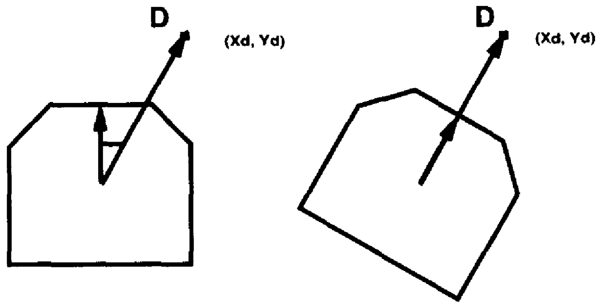


Fig. 8. Move behavior.

step, the maximum error can be one degree. When error bound has been considered, as, if the condition is too strict, the robot may oscillate around the destination. In experiments, the range of aligning error is set at one degree and its real error was found to be smaller than one degree. When the destination is in the right back direction, the robot is let to have a left-turn optionally. After completing the rotation mode, the robot has to run toward the destination. No error correction code was inserted during the run with both wheels at the same speed. In this behavior, the initial misalignment may lead to severe deviation at the final point. But from experimental results the final deviation was found to be less than 10 cm in the field of MIROSOT, 130 cm × 90 cm. Keeping both wheels at the same speed is preferable, as different speeds may cause an unsmooth motion in the caterpillar mechanism. As the robot is close to the destination, it can stop, if either of the following conditions are met: (i) the robot passes over the destination or (ii) the difference between the robot position and the destination is smaller than a given threshold.

4.1.2. Obstacle avoidance behavior

The robot has IR sensors to detect obstacles as shown in Fig. 3. The eight sensors detect obstacles in the front, back, right-hand or left-hand sides.

When the robot finds an obstacle in front of itself, to avoid the collision it turns to the left first and moves 10 cm forward, then turns to the right and moves 10 cm forward and then turns and moves towards the original destination D if the obstacle is not detected any more as shown in Fig. 9(a). However, if the robot detects the obstacles in the front and left-hand side as in Fig. 9(b), it turns to the right first and moves towards the original destination in a similar way as in Fig. 9(a). When the

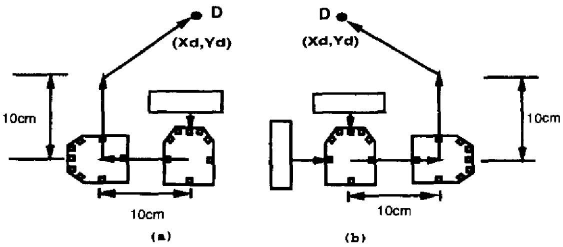


Fig. 9. Obstacle avoidance behavior.

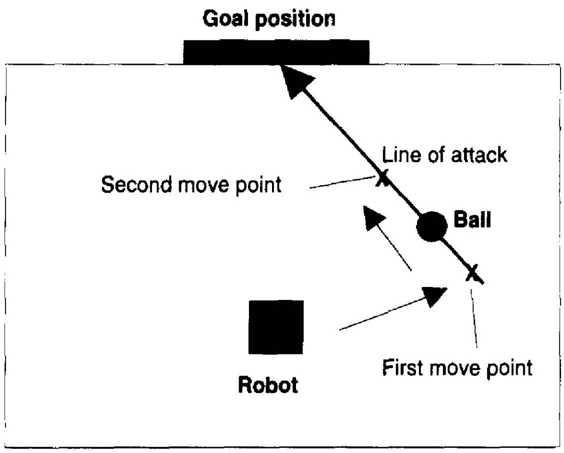


Fig. 10. Shoot and position_to_shoot actions.

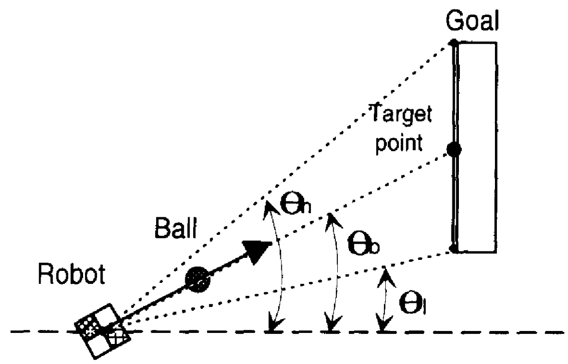


Fig. 11. Shoot action.

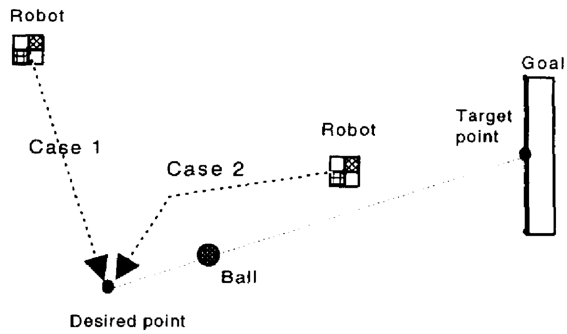


Fig. 12. Position_to_shoot action.

obstacle is in the dead angle of the robot, it turns to the destination once and then may find the obstacle again. In this case, it repeats the above procedure.

4.2. Basic actions

The basic actions are on the higher level than the basic behaviors. Only one action is selected by a high-level controller (central controller).

4.2.1. Shoot action and position_to_shoot action

These are used for shooting to the goal or for passing the ball to the other robot as in Fig. 10. All robots have these actions and mostly the attacking robots use them. Given the ball and goal positions, the two relative positions are calculated. The first is for *position_to_shoot* action as in Fig. 12 and the other is for shoot action as in Fig. 11. The shoot action is done if the following two conditions are met with: (i) the ball is located in between the robot and goal, and (ii) a straight line from the robot to the ball is covered by goal areas, that is, $\theta_l \le \theta_b \le \theta_h$.

Particularly, to prevent our robot from kicking the ball towards our side, it is designed with *position_to_shoot* action as in Case 2 in Fig. 12.

4.2.2. Intercept_ball action

After predicting a trajectory of the fast rolling ball, the robot moves to the point to intercept. The trajectory of the ball is calculated from the current and previous ball positions as its velocity is assumed to be constant within such a short span. Since the ratio of the distance moved by the ball to that of the robot is the same as the ratio of the predicted ball velocity and the maximum possible robot velocity, the intercept position is calculated from the following:

$$V_b \Delta t : V_r \Delta t = \sqrt{(X_i - X_b)^2 + (Y_i - Y_b)^2} : \sqrt{(X_i - X_r)^2 + (Y_i - Y_r)^2}. \quad (1)$$

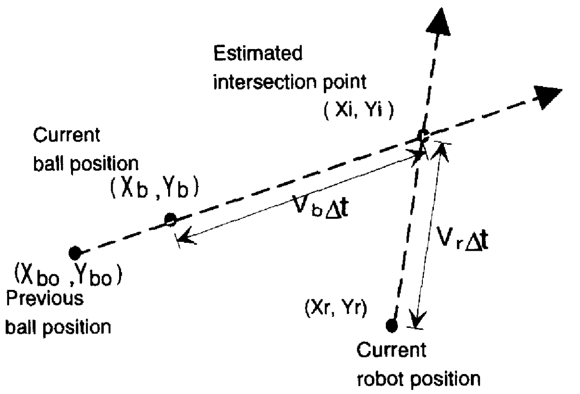


Fig. 13. Intercept_ball action.

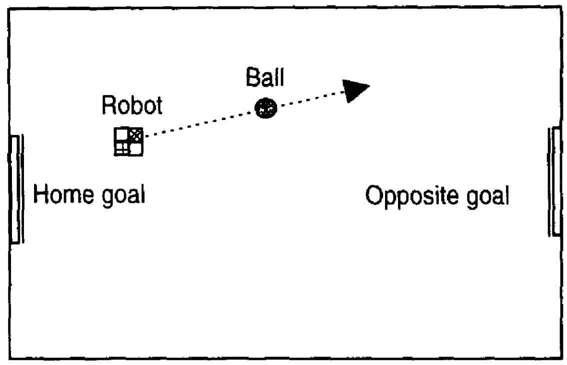


Fig. 14. Sweep_ball action.

Due to the time taken for the robot to turn, the vision processing time and the delay time caused by communication and so forth, intercept position errors might exist in practical systems. The position can be calculated easily from a second-order equation, bringing down the calculation time. Let the current and previous ball position be (X_b, Y_b) and (X_{bo}, Y_{bo}), respectively, as shown in Fig. 13 and (X_r, Y_r), (X_i, Y_i), V_b, V_r and Δt are the current robot position, the intercept point, the estimated ball velocity, the maximum robot velocity and intercept time, respectively. The trajectory of the ball can be fit by $y = ax + b$. From Eq. (1), the following second-order equation can be obtained:

$$
\begin{aligned}
&Ax^2 + 2Bx + C = 0,\\
&R = V_b^2/V_r^2,\\
&A = (R-1)(a^2+1),\\
&B = R(-X_r + a(b - Y_r)) + X_b - a(b - Y_b),\\
&C = R(X_r^2 + (b - Y_r)^2) - X_b{}^2 - (b - Y_b)^2.
\end{aligned}
$$

Among the two solutions solved from the above second-order equation, the solution pointing towards the moving direction of the ball will be selected.

4.2.3. *Sweep_ball action*

When the ball is located on our side, the robot kicks it towards the opponent's area as in Fig. 14. *Sweep_ball* action is same as *shoot* action in that the robot kicks the ball.

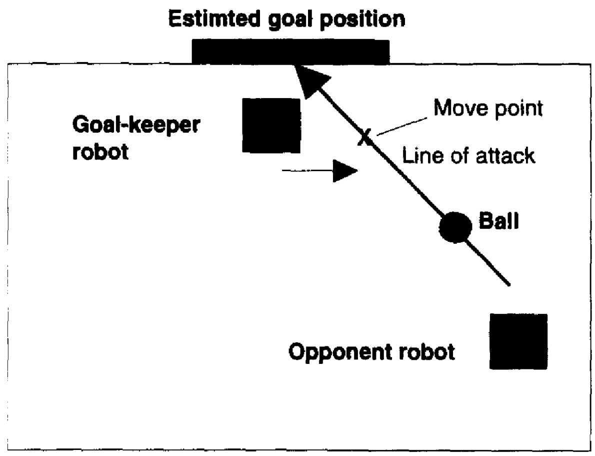


Fig. 15. Block action.

4.2.4. *Block action*

In contrast to the shoot action, the block action intercepts the ball or keeps off the opponent robots (Fig. 15). This action is mainly used by defense robots. The robot accomplishes this behavior by moving its position to the expected attack point, considering the ball and our goal positions. The attack route may depend on the point aimed at, by the opponent robot. So, the defense robots' positions are dependent on positions of the opponent robot and of the ball. In this action no estimation of the ball trajectory is used as in *intercept_ball* action.

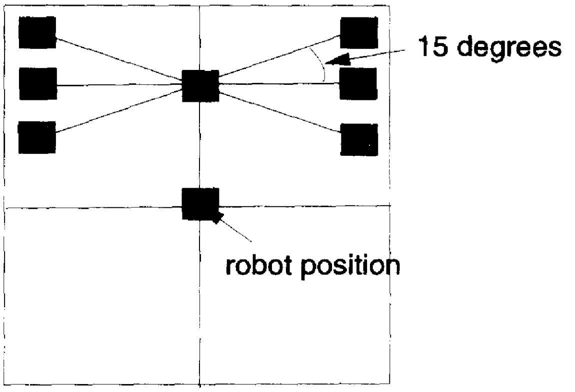


Fig. 16. Positions of bottom sensors.

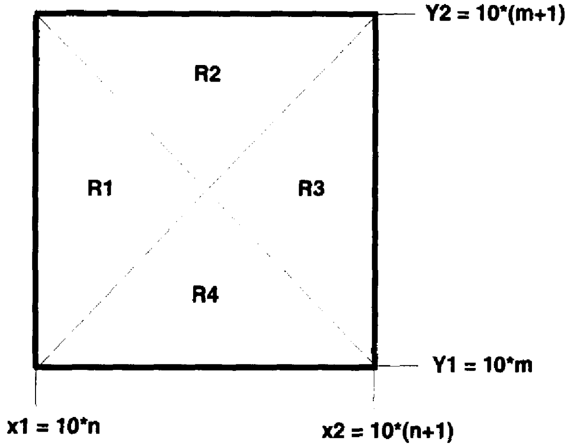


Fig. 17. Correction of position (X, Y) using bottom sensors.

4.3. Position correction by detecting the field lines

The robot position is defined as the intersection of the middle line between the front and back wheels and the vertical centerline of the robot. The position correction is done by detecting the grid lines on the field. If the sensor at the robot center detects a line, the X or Y coordinate of the robot will be $10n$ cm, where n is a positive integer. If the robot detects a line, its X or Y coordinates are changed as given by (Fig. 17)

$$X = \begin{cases} X1 & \text{if } (X, Y) \text{ in R1}, \\ X2 & \text{if } (X, Y) \text{ in R3}, \end{cases} \tag{2}$$

$$Y = \begin{cases} Y1 & \text{if } (X, Y) \text{ in R4}, \\ Y2 & \text{if } (X, Y) \text{ in R2}. \end{cases} \tag{3}$$

This method is relatively simple but, proved to be useful from the actual experiments, where the position error was found to be 2–3 cm. However, the rotation error θ was found to increase in cases of over 5–6 turns. As a result, the position error (X, Y) also increases and makes the compensation more difficult. If the exact θ is available, the position correction by the above algorithm guarantees only 2–3 cm of position errors. When the robot passes a line and the sensors are arranged in a certain direction to match the line, θ can be precisely corrected. In Fig. 16, sensors at the front part of the robot detect the line only when the robot moves the line with the slope of $0°$, $15°$, $-15°$. Otherwise, the robot cannot correct the θ error. Correction works well or not, depending on the robot trajectory. Error increases in the case of over 5–6 turns in 30–60 cm movement (relatively long) compensation is not possible.

5. One-on-one soccer game using ASM

In this section, an ASM [4] for soccer-playing robots is briefly explained and some experimental results of one-on-one robot soccer game are shown. Here, the ASM can be considered as a concrete implementation of the *central controller* part and *basic actions* part of the above-mentioned robot architecture.

5.1. ASM for robot soccer

It is a basic problem to determine what action the agent should take in a given situation. Particularly, the soccer-playing robot should take an appropriate action according to its role such as striker, sweeper or goalkeeper. Some computational mechanism for such action selection is essential in that sense. It is assumed that the role of each agent is fixed and the number of its available actions is finite. The approach to design the ASM for robot soccer is as follows. First, a relatively simple ASM is designed for the situation with no opponents. After that, the mechanism incorporates some additional action selection schemes with the opponents. The opponents are considered as a kind of disturbances for each agent. The ASM for each agent considers the opponents only when the disturbance level is over a prespecified threshold by the opponent's robot.

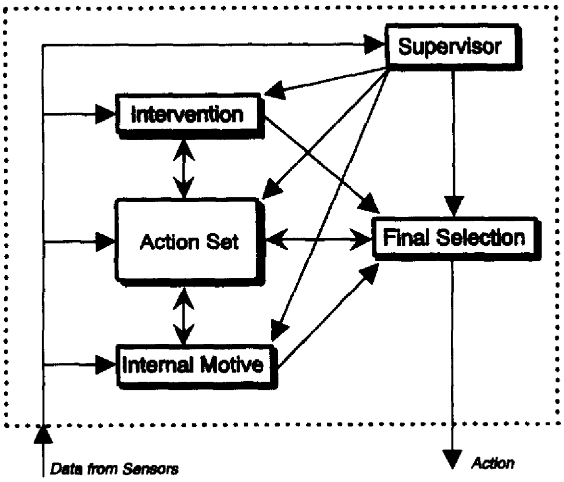


Fig. 18. The structure of action selection mechanism (ASM) as a high-level controller.

The structure of the ASM is shown in Fig. 18. It consists of *action set, supervisor, internal motive, intervention*, and *final selection* modules. *Action set* module consists of several actions for each agent to satisfy its given role and provides informations of actions which other modules inquire from the *action set*, such as run-time parameters and the feasibility of each of the actions. *Action set* is based on basic actions described in the previous section. *Supervisor* plays the part of an enforcement for the agent to do certain actions and a modification of the attributes of actions. *Internal motive* module is the action selection module for the situation without considering opponents. *Intervention* module calculates the level of disturbance of opponents. If the disturbance level due to opponent's agents is above some threshold, the *intervention* module suppresses *internal motive* module and selects a new action. No explicit model of the opponents is assumed here. Human judgment can be used to model opponents' behaviors and strategies, and a simple multilayer perceptron (MLP) is adopted as a tool to learn. Human beings select the situations in which the disturbance level due to opponent's agents is very high. The appropriate situation variables for the selected situations are calculated and used as the training data for MLP. *Final selection* module takes into account the outputs of other modules and select the final proper action for each agent to do at that situation.

5.2. MLP for intervention module

5.2.1. The structure of MLP

The MLP uses 10 inputs, two outputs and two hidden layers. The number of nodes in the first hidden layer is 12, that in the second is 6. The activation function is used as a sigmoid function $\varphi(v) = 1/(1 + e^{(-av)})$. While the number of input and output nodes are dependent on problem, 10 situation variables for input nodes and two actions variables for output nodes are considered here.

5.2.2. Output of MLP

Actions of intervention modules are given as *sweep_ball* and *block*. *Sweep_ball* represents an action of kicking away the ball when the disturbance level is over some threshold by opponent's agents. Also *Block* does when the risk level of missing the goal is high. The output value of MLP is between 0 and 1.

5.2.3. Input of MLP

The situation variables for MLP are as follows:

- Situation variables representing possessing of the ball (Fig. 19):
  - θ_{BR}: angle between moving direction and direction toward the ball of home robot;
  - θ_{BO}: angle between moving direction and direction toward the ball of opponent's robot;
  - D_{BR}: distance between the ball and home robot;
  - D_{BO}: distance between the ball and opponent's robot.
- Situation variables representing a risk level of missing the goal (Fig. 20):
  - D_{BRG}: distance between the ball and home goal;
  - D_{IRG}: distance between the center of goal and the intersection of home goal line and a line between opponent's robot and the ball.
- Situation variables representing winning score measure against opponent's goal (Fig. 20):
  - D_{BOG}: distance between the ball and opponent's goal;
  - D_{IOG}: distance between the center of goal and the intersection of opponent's goal line and a line between home robot and the ball.
- Velocities of the ball and opponent's robot.

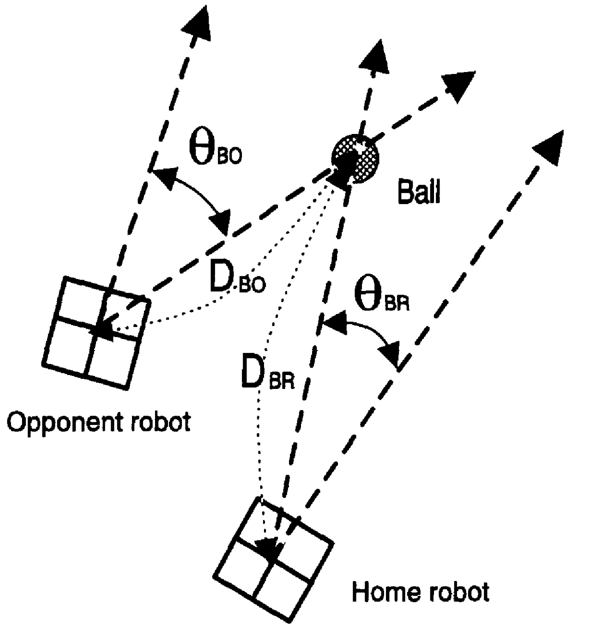


Fig. 19. Situation variables representing possessing of the ball.

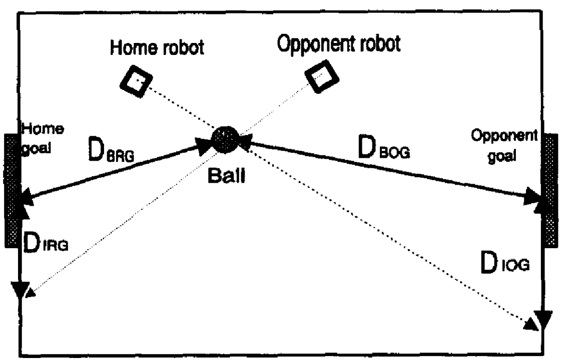


Fig. 20. Situation variable representing winning score measure against opponent's goal and a risk level of missing the goal.

5.3. Learning of MLP

The training data for MLP are collected through a real robot soccer game. The game is between an agent with the proposed ASM excluding *intervention* module and an opponent agent which may have some kind of ASM or other control algorithms. The game data such as the X, Y positions and heading of each robot and the X, Y positions of ball are stored. Then, a human being observes the replayed game on a computer 2D graphic display, and judges the situation where the disturbance level due to opponent's agents is very high. The best among the many actions given to the *intervention* module will be identified and the corresponding data will be stored. The graphic display used here is a simple two-dimensional animation. If the game is displayed on a three-dimensional delicate graphics, the better its realization the better will be the judgments of the human being. Situation variables, the inputs of MLP, are thus calculated from the selected game. The desired output of the selected action is set to 1 and the outputs of other actions are set to 0. The error back-propagation algorithm, one of the supervised learning methods, is used to learn the MLP with the above collected training data. The learned MLP is applied to the stored game to check whether it works well in the game. Once its performance is within the desired levels, the *intervention* module implemented by the MLP can be applied to a one-on-one soccer game (Fig. 21).

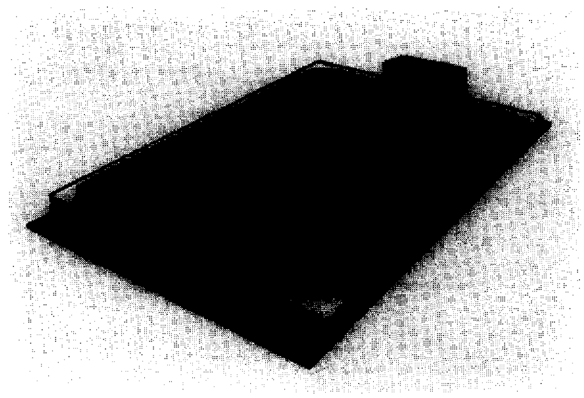


Fig. 21. One-on-one soccer game between two robots.

5.4. Action selection results of the proposed ASM

Fig. 22 shows the trajectories of robots and the ball at some time interval in the whole game. The right side is for robot R which has the *intervention* module and the left side is for robot O which does not have it. In this figure, characters B, R and O represent initial positions for ball, robot R and robot O at the beginning of the time interval, respectively. This time interval is denoted as *interval A*. A solid line representing the trajectory of ball, a bold dotted line that of robot R and

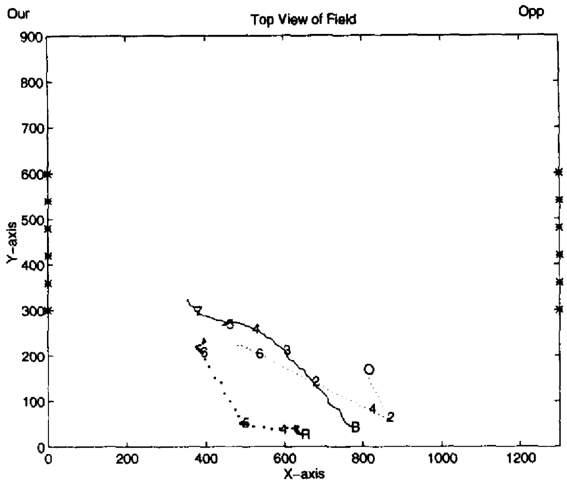


Fig. 22. Robots and the ball's trajectories in interval A.

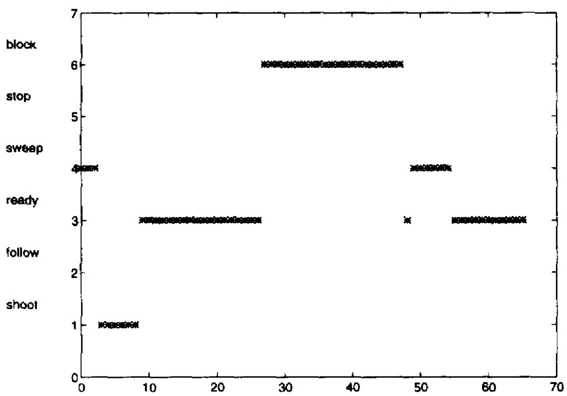


Fig. 23. Selected actions in interval A.

a fine dotted line that of robot O are shown in Fig. 22. The numbers on each trajectory are the elapsed time in seconds. Fig. 23 shows a sequence of actions selected by robot R, which has the proposed ASM within the *interval A*. Y-axis values represent selected actions, 1 for *shoot*, 2 for *intercept_ball*, 3 for *position_to_shoot*, 4 for *sweep_ball*, 5 for *stop* and 6 for *block*. The time unit of X-axis is 100 milli-second. For example, $y = 6$ at $x = 40$ represents the robot R selected *block* action at $t = 4$ of *interval A*. At $t = 4$, the robot O has the good opportunity to shoot towards the robot R's goal. So, the robot R selected *block* action, a defense action to block ball and keep its goal.

The changes of the action selection of robot R occurred six times for 7 s, a relatively short duration, which means the situations in *interval A* were very dynamic.

5.5. One-on-one soccer game

The one-on-one robot-soccer game was done between the two agents with the proposed ASM. Table 1 shows the scores for five games.

If the disturbance level is over some threshold by the robot O, the robot R selects a new action by *intervention* module to reduce it. *Intervention* module using MLP made an appropriate action about almost all situations needing interventions. But, it often happened that *intervention* module selected other actions as well with the help of a human operator. It may have resulted due to the learning with inappropriate game data or insufficient situation variables. Human operators can select two conflict actions such as *sweep_ball* and *block* actions under the same situation in turn, so the made learning data has the conflict rules. Or replayed game in 2D graphic environment may not be reflected in real game situation. But the inappropriate outputs of *intervention* module which occurred occasionally did not make a bad result in the total processing of the games. It is noted that if the training set taken from games done by agents with various action patterns are used to learn the MLP, the applicability of the MLP can be improved. As it can be seen, the robot R having an *intervention* module is a little better than the robot O and the score difference is not much. It is unreasonable to assert that the robot R is superior to the robot O only seeing the scores. But, the predominance of the robot R could be observed through the above games, although the score table does not reveal sets.

When only the robot R entered the game without an opponent's agent, the number of goals were found to

Table 1
Scores for five games

Robot	Game				
	1	2	3	4	5
R	2	1	2	1	3
O	1	1	2	0	1

be 3–5 in 5 min on an average. Comparing the scores in Table 1, it can be observed that the disturbance level is highly dependent on the opponent's agent in competing the soccer game.

6. Experimental results in MIROSOT and discussion

6.1. Experimental methods

MIRO and SOTY robot teams were developed for the experimental purpose and the exercise games were conducted between the two teams to develop and confirm the strategies. As the development of robot-based soccer robot system took a lot of time, only vision-based one was implemented in both teams. It was not possible to predict game results since robots did not move purposely due to incompleteness in algorithm and uncertainties at the beginning of its development. As the algorithm got updated, the performances, also improved. If the two teams have the same abilities, algorithms and strategies, similar situations occur during every game and only the observed problems could be solved. Since various aspects of the games were needed to show whether the algorithms of the soccer robot system operate well, we made the two teams' abilities different. They have the same hardware, but their speeds were made different with changes in gear ratios and hence one team used fast robots and the other, slow ones.

6.2. Strategies

The basic strategy of SOTY and MIRO were the zone defense, but the applied strategies were very complicated in order to meet the several situations in the game. Each robot has its own role such as a striker, a defender and a goal-keeper. It has also its own area according to its role as in Fig. 24. At the beginning of development, the concept of zone defense was that the robot selected an appropriate action among basic actions if only the ball is located in its own area and the other robots did not select any actions because the ball was out of their areas. But, this concept had two problems: (i) If a robot gets blocked by the opponents' robot, our robots should stop and the game comes to a standstill. While the ball is located in the opponent's area and the striker robot is blocked, the opponent's team has an advantage, as the goal-keeper and defender robots get stopped. (ii) If the ball is located in one of the boundary areas, two robots can move towards it and may collide with each other.

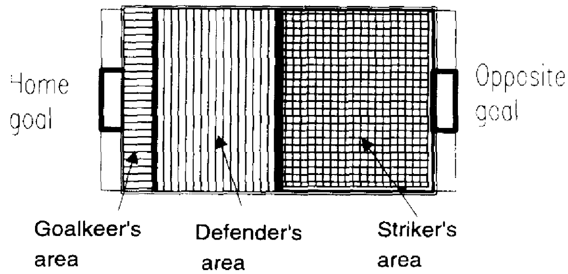


Fig. 24. The areas assigned to robots. These can change according to strategies.

So the concept of zone defense has to be modified to give a priority of action selection to the robot in its own area, while the other robots can move to any place of the playground. A robot in its own area according to its role has given a higher priority of action selection. For example, the goal-keeper robot has a higher priority in goal-keeper area and the other robots should select other actions which do not conflict with the goal-keeper action. When the ball is located in the goal-keeper area, the ball must be kicked out towards opponent's area to reduce the risk of missing goal. In this case, the goal-keeper selects *sweep_ball* action according to its priority, and the other robots select *block* action or move toward any place to avoid the conflicts with the goal-keeper's action.

Besides the modified zone defense strategy, several strategies were used in real soccer games. If the opponent's team performance was not so good, normally '3–0–0' strategy (three strikers) that all robots' roles were strikers was used. Otherwise, '1–1–1' strategy (a striker, a defender and a goal-keeper) or '0–2–1' strategy (two defenders and a goal-keeper) were used. According to the opponent's team performance, game importance (final or preliminary game) and the game score, different strategies were employed. We implemented 'strategy data-base' to select a strategy properly according to the game situation. This was done by the human operator as it is difficult to make the robot decide by itself.

6.3. *Game result when a freekick happened*

Fouls may get called in robot soccer game as in a real soccer. When penalty-kick, freekick, freeball or goalkick are called according to rules, the game is replayed after placing robots on predefined positions according to rules [5]. In cases of fouls, special strategies were used, such as 'set-position play' as in a real soccer game. Using a modified zone defense and several strategies including the set-position play resulted in successful defense through 20 official games in MIROSOT'96 whenever our teams were faced with freekick or penalty-kick defense (refer MIROSOT'96 game video tape) [8,6]. These were implemented using 'if–then' rule bases.

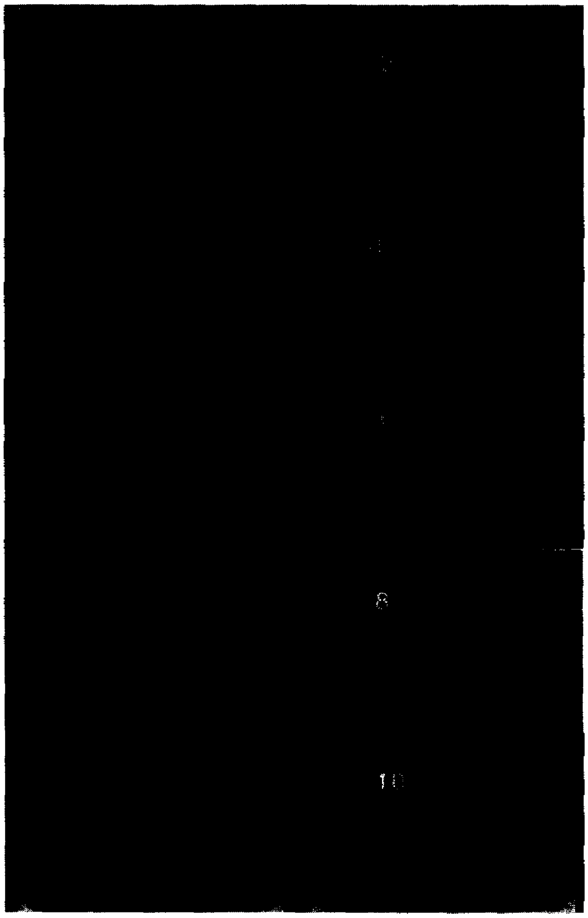


Fig. 25. The game between MIRO and SOTY (when freekick foul is called): The ball is drawn by the square and MIRO consists of robots drawn by circles.

Fig. 25 shows pictures of a game between MIRO and SOTY when a freekick foul is called. SOTY is a defense team and MIRO is an opponent's team. It is captured using the vision system, then post-processed to identify the ball and six robots. Both teams were vision-based soccer robot systems and used the same strategies. Different frequencies (418 and 433 MHz) were used by the teams to avoid interference in communication. The ball and robots are placed on the playground according to rules, as in region 1 of Fig. 25. Opponent team's (MIRO) robot kicks the ball as it is located at an available shooting area in region 2. Defense team's (SOTY) robots block the ball after getting aligned towards the goal center in accordance with the special strategy in regions 2 and 3. After that situation, SOTY employs a general strategy (modified zone defense) and robots behave according to roles in regions 4–10. MIRO also uses a general strategy in this situation in regions 2–10.

7. Conclusion

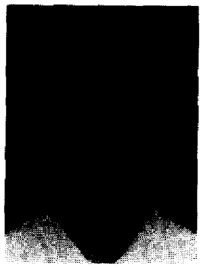
Three control schemes such as a remote-brainless soccer robot system, a vision-based one and a robot-based one have been defined for robot soccer. The developed soccer robots can be used with the robot-based system as well as the vision-based system, as they were designed as autonomous ones. A control structure, behaviors and actions have been proposed for soccer robots. The action selection mechanism (ASM) has been proposed as a high-level controller and applied to a one-on-one soccer game. The intervention module of ASM has used multilayer perceptrons and showed its effectiveness through experimental results. Also modified zone defense as a basic strategy and several strategies for corresponding situations have been developed for robot soccer. The SOTY and MIRO have been placed in second and fourth positions in MIROSOT'96 (Micro-Robot World Cup Soccer Tournament) held at KAIST in November 1996, with the proposed scheme and micro-robots.

References

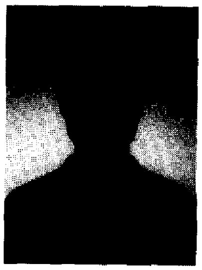
[1] B.L. Brumitt and A. Stentz, Dynamic mission of planning for multiple mobile robots, *IEEE. Proc. Conf. on Robotics and Automation* (1996) 2396–2401.

[2] M.-J. Jung, An artificial immune network-based decision making scheme for multi-agent systems, Master Thesis, KAIST (1997).
[3] Y. Kawauchi, M. Inaba and T. Fukuda, A principle of distributed decision making of cellular robotic system (CEBOT), *IEEE Proc. Int. Conf. on Robotics and Automation*, USA (1994) 833–838.
[4] H.-S. Kim, Action selection mechanism of multi-agent systems for robot soccer, Master Thesis, KAIST (1997).
[5] J.-H. Kim, A booklet on MIROSOT'96 (Micro-Robot World Cup Soccer Tournament), MIROSOT Organizing Committee, April 1996; accessible from http://www.mirosot.org/.
[6] J.-H. Kim, Call for participation: 1997 Micro-robot World Cup Soccer Tournament (MIROSOT'97), 1997; accessible from http://www.mirosot.org/.
[7] J.-H. Kim, M.-J. Jung, H.-S. Shim and S.-W. Lee, Autonomous micro-robot 'Kity' for maze contest, *Proc. Int. Symp. on Artificial Life and Robotics (AROB)*, Oita, Japan (1996).
[8] J.-H. Kim, H.-S. Shim, M.-J. Jung, H.-S. Kim, I.-H. Choi and W.-S. Han, Building a soccer robot system for MIROSOT'96, *Video Proc. IEEE Int. Conf. on Robotics and Automation*, Albuquerque, New Mexico (1997).
[9] C.R. Kuba and H. Zhang, The use of perceptual cues in multi-robot box-pushing, *IEEE Proc. Int. Conf. on Robotics and Automation* (1996) 2085–2090.
[10] N. Mitsmoto, T. Hattori, T. Idogaki, T. Fukuda and F. Arai, Self-organizing micro robotic system (biologically inspired immune network architecture and micro autonomous robotic system, *IEEE Proc. Int. Sym. on Micro Machine and Human Science-Toward Micro-Mechatronics*, Japan (1995) 261–270.
[11] F. Mondada, E. Franzi and P. Ienne, Mobile robot miniaturization: A tool for investigation in control algorithms, *Proc. 3rd Int. Symp. on Experimental Robotics*, Japan (1993).
[12] L.E. Parker, Heterogeneous multi-robot cooperation, Ph.D. Thesis, MIT, AI Lab., Cambridge, MA (1994).
[13] M.K Sahota, A.K. Mackworth, S.J. Kingdon and R.A. Barman, Real-time control of soccer playing robots using off-board vision: The dynamic test bed, *IEEE Proc. Int. Conf. on Robots and Automation*, Japan (1995) 3690–3693.
[14] H.-S. Shim and J.-H. Kim, A stable locomotion control of autonomous mobile robot, *Proc. 1st Asian Control Conf.*, Tokyo, Japan (1994).
[15] H.-S. Shim, J.-H. Kim and K. Koh Variable structure control of nonholonomic wheeled mobile robot, *Proc. IEEE Int. Conf. on Robotics and Automation*, Japan (1995).

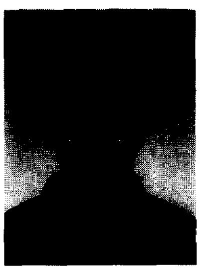
Hyun-Sik Shim received the B.S. and M.S. degrees in Electrical Engineering from Korea Advanced Institute of Science and Technology (KAIST), Taejon, South Korea, in 1991 and 1993, respectively. He is currently a candidate for the Ph.D. degree in the Department of Electrical Engineering at KAIST. His research interests are in the areas of mobile robots, intelligent machines, and multiple robots cooperation. He is a student member of IEEE, KIEE and KFIS and a secretariat of MIROSOT'97.



Heung-Soo Kim received the B.S. and M.S. degrees in Electrical Engineering from Korea Advanced Institute of Science and Technology (KAIST), Taejon, South Korea in 1995 and 1997, respectively. He is currently a Ph.D. candidate in the Department of Electrical Engineering at KAIST. His research interests are in the areas of neural networks, mobile robots and artificial intelligence.



Myung-Jin Jung received the B.S. and M.S. degrees in Electrical engineering from Korea Advanced Institute of Science and Technology (KAIST) in 1995 and 1997, respectively. He is currently a Ph.D. candidate at Department of Electrical engineering, KAIST. His research interests cover micro-robotics, multi-robot systems and artificial immune networks. He has been working for MIROSOT (Micro Robot Worldcup Soccer Tournament) since 1995.



In-Hwan Choi received the B.S. degrees in Electrical Engineering from Korea Advanced Institute of Science and Technology (KAIST), Taejon, South Korea, in 1996. He is currently an M.S. student in the Department of Electrical Engineering at KAIST. His research interests are in the areas of mobile robots and Evolutionary Computation.

Jong-Hwan Kim received the B.S., M.S., and Ph.D. degrees in Electronics Engineering from Seoul National University, Seoul, South Korea, in 1981, 1983, and 1987, respectively. From 1984 to 1988, he was a Research Assistant in the Department of Electronics Engineering at Seoul National University. Since 1988, he has been with the Department of Electrical Engineering at the Korea Advanced Institute of Science and Technology (KAIST), where he is currently an Associate Professor. He was a Visiting Scholar at Purdue University from September 1992 to August 1993. His research interests are in the areas of intelligent control and systems. Dr. Kim was the recipient of the 1988 Choon–Gang Young Investigator Award from Choon–Gang Memorial Foundation. He is an Associate Editor of the *International Journal of Intelligent and Fuzzy Systems* and *IEEE Transaction on Evolutionary Computation*. He is the founder and Organizing Chair of the MIROSOT (Micro-Robot World Cup Soccer Tournament).

Jin-Oh Kim received B.S. (1983) and M.S. (1985) degrees in Mechanical Engineering from Seoul National University, South Korea. He joined CAD/CAM Lab. of Korean Institute of Science and Technology as a commissioned researcher in 1985. He received the Ph.D. degree in Robotics from the Robotics Ph.D. Program of Carnegie-Mellon University in USA in 1992. He joined SECOM Intelligent Systems Lab. in Tokyo, Japan. in 1992. and joined SAMSUNG Electronics Co., Ltd. as a principal engineer in 1994. Currently, he is now in charge of Robot Business Group as a Senior Manager in Suwon, South Korea. His research interests include optimal design of robots and automation systems, intelligent systems and optimization algorithms.