ELSEVIER

Fuzzy Sets and Systems 127 (2002) 49–64

FUZZY sets and systems

www.elsevier.com/locate/fss

Strategy-based decision making of a soccer robot system using a real-time self-organizing fuzzy decision tree

Han-Pang Huang *, Chao-Chiun Liang

Department of Mechanical Engineering, Robotics Laboratory, National Taiwan University, Taipei, Taiwan 10674, ROC

Abstract

In a soccer robot game, the environment is highly competitive and dynamic. In order to work in the dynamically changing environment, the decision-making system of a soccer robot system should have the features of flexibility and on-line adaptation. This paper proposed a strategy-based decision-making system for a soccer robot system. The decision system is based on the proposed self-organizing fuzzy decision tree (SOFDT). Due to event-driven strategies, the proposed decision-making system has the advantage of flexibility. The proposed SOFDT possesses simple, apparent, and fast generation/reasoning processes. The resultant system is a self-organizing strategy-based decision-making system. According to decision results, SOFDT can on-line modify its parameters and structure for achieving better adaptability and improvement. The performance of the proposed systems was verified by applying them to the simulation and experiment of 3-to-3 robot soccer games. The results are satisfactory.

Keywords: Fuzzy learning decision tree; Self-organizing; Strategy-based decision making; Soccer robots

1. Introduction

In a soccer robot game, the environment is highly competitive and dynamic. In order to work in the dynamically changing environment, the decision-making system of a soccer robot system should have the features of flexibility and on-line adaptation. A typical multi-agent soccer robot system, such as NTU Formosa built by our laboratory, is shown in Fig. 1 [1,5–9,11,14,15]. This study used NTU-Formosa as the platform to develop a self-organizing strategy-based decision-making system. NTU-Formosa consists of multiple mobile robots, a vision system, a wireless communication system, and a host computer.

In Fig. 1, the image information of the entire soccer field is captured by a CCD camera and sent to the corresponding host computer. The host computer then analyzes the image information to determine the situation of the soccer field. According to the determined situation, the host computer decides a strategy and plans the motion modes and the corresponding velocity commands for every soccer robot of the same team. Each soccer robot of NTU-Formosa then receives velocity command from the host computer and regulates the rotational velocities of its right and left wheels.

A decision tree has several nodes arranged in a hierarchical structure. It implements decisions in a simple,

* Corresponding author. Tel./fax: +886-2-2363-3875.
E-mail address: hanpang@ccms.ntu.edu.tw (H.-P. Huang).

0165-0114/02/$ - see front matter
PII: S0165-0114(01)00152-X

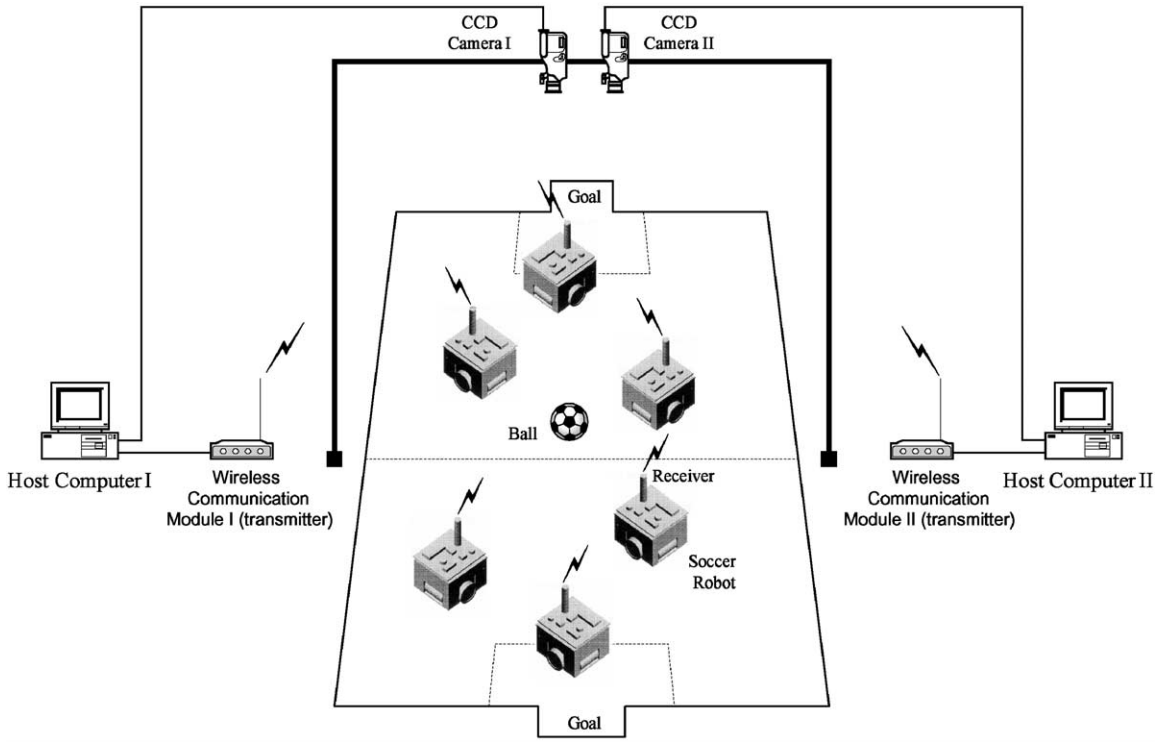

Fig. 1. Graphical diagram of two competing teams of NTU-Formosa.

apparent, multistage manner. Furthermore, since each node of a decision tree uses only a simple splitting rule and a small subset of all features, the entire decision process is fast and efficient [2,3,10,13]. This study proposed a self-organizing strategy-based decision-making system for NTU-Formosa to modify strategies and decide which strategy should be applied to the current situation. The decision system is based on the proposed self-organizing fuzzy decision tree (SOFDT). Due to event-driven strategies, the proposed decision-making system has the advantage of flexibility. It can be shown that the proposed SOFDT possesses simple, apparent, and fast generation/reasoning processes.

To generate a practical decision tree, numerous training data are necessary. The fuzzy learning decision tree adopted in this study uses fuzzy statistics [12,16] to generate several fuzzy sets from the training data. This method effectively compresses the training data and meaningfully represents its statistical distribution. This has the advantages of increasing computation speed, saving storage memory, and maintaining satisfactory performance. Given a set of training data, an optimal decision tree is inherently difficult to obtain [10,13]. To cope with this difficulty, the adopted fuzzy learning decision tree uses a sub-optimal splitting rule at each node to generate a decision tree. This is a cost-effective method that quickly obtains a decision tree with satisfactory performance.

In order to adapt to the changing environment, the proposed SOFDT on-line tunes its parameters and adapts its structure to improve its performance in terms of decision results and the input data. To increase learning speed, the proposed SOFDT modifies only a small subset of the decision tree by a local learning mechanism in each learning cycle. If the decision result is correct, SOFDT tunes its parameters to reflect the new statistical distribution. If the decision result is incorrect, SOFDT tunes its parameters for single-modal case and adapts its structure for multi-modal case. If the decision result is unknown, SOFDT adapts

its structure to cope with the multi-modal and new model cases. Finally, if the input data is unreasonable, SOFDT has a set of rules to define and identify them and then ignore them in learning process. SOFDT can perform on-line recognition and learning in real-time applications. To examine the performance of the proposed systems, the simulation and experiments of 3-to-3 robot soccer games will be conducted.

The organization of this paper is as follows. The strategy-based decision-making system of NTU-Formosa is developed in Section 2. In Section 3, a self-organizing fuzzy decision tree is proposed and its performance is analyzed. The proposed SOFDT is applied to the strategy-based decision-making system in Section 4. In Section 5, simulation and experiment of 3-to-3 soccer robot games are given and discussed. Finally, several concluding remarks are drawn.

2. A strategy-based decision-making system of NTU-Formosa

The strategy-based decision-making system for selecting strategies is implemented by decision trees. The decision-making system first recognizes the current situation of the soccer field, including the ball lying in which zone and under which team's control. A strategy is then selected, according to the current situation, to coordinate the multiple robots of NTU-Formosa to play the soccer game. This section first constructs the architecture of the decision-making system. A set of practical strategies is then developed to cope with the situations considered in the constructed decision-making system.

2.1. Architecture

In a soccer field, the current situation depends on many factors. For two n-robot teams, the number of factors affecting the current situation can be estimated by the following equation:

$$N_f = 4(2n + 1), \tag{1}$$

where N_f is the number of affecting factors; the item '4' represents two-dimensional position and velocity vectors; the item '1' represents the ball. Note that Eq. (1) does not consider accelerations of the ball and the robots. It is only an estimation of the complexity considering all kinds of situations. If all the factors are considered, the efficiency of the decision-making system will be poor. Decision trees have the advantage of considering only relevant factors to make a decision. To further improve the efficiency of the adopted decision tree, this study first considers the most important factor and places it in the root node of the decision tree. The second most important factor is then considered and placed in the level next to the root node. This tree-generation process continues until all the available and feasible factors are considered and placed in the proper nodes of the decision tree.

In a soccer game, the two most important issues are to bring the ball into the opponent's goal and to prevent the ball from going into teammate's goal. Hence, much attention should be paid to the ball. This study regards the position of the ball as the most important factor and places it into the root node of the decision tree. A soccer field is divided into four zones in this study and hence the ball should lie in one of them. The four zones are offense zone, formation zone, defense zone, and near-wall zone. The offense zone is near the opponent's goal and shooting is the major goal. The formation zone is in the middle field and making up the formation is the main goal. The defense zone is near one team's own goal and preventing shooting is the first duty. The near-wall zone is a region in which the ball is difficult to shoot or pass directly. Since the root node decides in which zone the ball lies, there are four sub-nodes for the root node. Next, this study considers the controllability of the ball as the second most important factor and places it into each of the sub-nodes of the root node. This is because the team controlling the ball has the initiative to attack or to defend. These sub-nodes are non-terminal nodes deciding which team controls the ball. There are three cases about the controllability of the ball: controlled by teammate, controlled by opponent, and uncontrolled. Thus, there are three sub-nodes for each of these non-terminal nodes. The decision tree for the strategy-based decision-making system is shown in Fig. 2. In Fig. 2, there are 12 terminal nodes, and each represents a strategy coping with the corresponding situation. By experience and heuristic rules, the decision tree is optimized in its tree-generation process. This optimization indirectly optimizes the recall process of the decision tree. When the number of robots

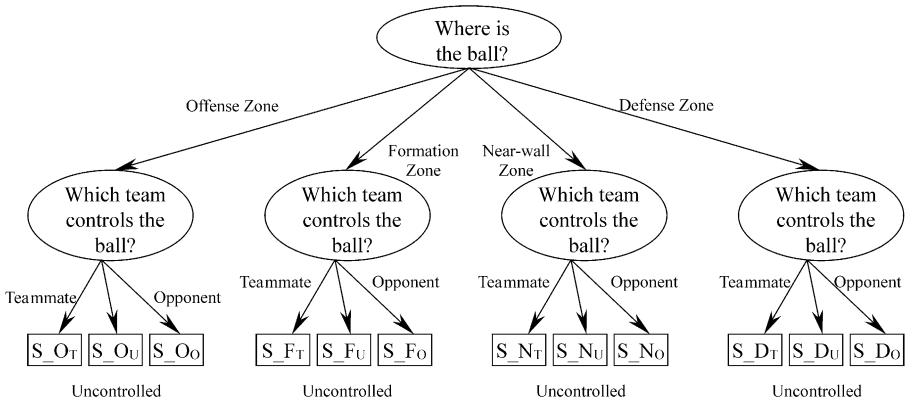

Fig. 2. The decision tree which implements the strategy-based decision-making system.

increases, however, the decision tree will become more and more sophisticated in order to consider most of the important situations. According to Eq. (1), it is very difficult or even impossible to manually list all kinds of situations. Therefore, a self-organizing decision tree is necessary and will be developed in this study.

2.2. Practical strategies in the decision-making system

In a 3-to-3 soccer game, each team has three robots with different duties. The robot in charge of offense zone is called offense-robot, the robot in charge of formation zone is called formation-robot, and the robot in charge of defense zone is called defense- robot. The strategies for the decision tree in Fig. 2 are developed based on heuristic rules and game dynamics described by the concepts of attack angle and pass angle [5,4,6,9]. The proposed attack angle is approximately proportional to the probability of successful shooting. Similarly, the proposed pass angle is derived from attack angle and is approximately proportional to the probability of successful passing. The developed strategies for 3-to-3 soccer games are described as follows:

1. S_O_T: The *offense-robot* tries to find a suitable attack angle to shoot the ball. The *formation-robot* estimates the velocity of the ball and prepares to hold the rebounded ball. The *defense-robot* defends the goal.
2. S_O_U: The *offense-robot* tries to find a suitable attack angle to shoot the ball. The *formation-robot* blocks the pass angle of the ball. The *defense-robot* defends the goal.
3. S_O_O: The *offense-robot* tries to intercept the ball. The *formation-robot* blocks the pass angle of the ball. The *defense-robot* defends the goal.
4. S_F_T: The *offense-robot* prepares to receive the ball. The *formation-robot* tries to pass the ball to the defense-robot. The *defense-robot* defends the goal.
5. S_F_U: The *offense-robot* prepares to receive the ball. The *formation-robot* tries to intercept the ball. The *defense-robot* defends the goal.
6. S_F_O: The *offense-robot* blocks the pass angle of the ball. The *formation-robot* tries to intercept the ball. The *defense-robot* defends the goal.
7. S_N_T: The *offense-robot* prepares to receive the ball. The *formation-robot* clears the ball to offense zone. The *defense-robot* defends the goal.
8. S_N_U: The *offense-robot* prepares to intercept the ball. The *formation-robot* tries to clear the ball. The *defense-robot* defends the goal.
9. S_N_O: The *offense-robot* tries to intercept the ball. The *formation-robot* blocks the pass angle. The *defense-robot* defends the goal.
10. S_D_T: The *offense-robot* and *formation-robot* prepare to receive the ball. The *defense-*

robot tries to pass the ball to the offense-robot or formation-robot.

11. S_D_U: The *offense-robot* prepares to receive the ball. The *formation-robot* tries to intercept the ball. The *defense-robot* defends the goal.
12. S_D_O: The *offense-robot* tries to intercept the ball. The *formation-robot* blocks the attack angle of the ball. The *defense-robot* defends the goal.

In the 12 strategies developed above, the terms 'shoot', 'block', 'pass', 'defend', 'clear', 'intercept', and 'receive' are all soccer skills. The reader can refer to [9] for the details.

3. A self-organizing fuzzy decision tree

3.1. Architecture of the self-organizing fuzzy decision tree

The concept and architecture of the proposed SOFDT (self-organizing fuzzy decision tree) can be outlined in a flowchart, as shown in Fig. 3. Clearly, the SOFDT architecture is the kind of divide-and-conquer. The components of SOFDT, including tree-generation algorithm, on-line learning mechanism, error-rejection rules, and performance analysis, are explained below.

3.2. Automatic generation of a fuzzy decision tree

The tree-generation algorithm consists of three steps: (1) select a set of features that can well describe the objects and then massively measure every feature of each object to acquire sufficient training data, (2) apply fuzzy statistics to find fuzzy sets for compressing and representing the training data, (3) determine a decision tree from the fuzzy sets. The structure of a decision tree denotes the topology of the tree. The parameters of a decision tree include the objects and the threshold in each non-terminal node. The tree-generation algorithm can be detailed in a flowchart, as shown in Fig. 4. The above-mentioned three steps and components of the flowchart are described in the subsequent context.

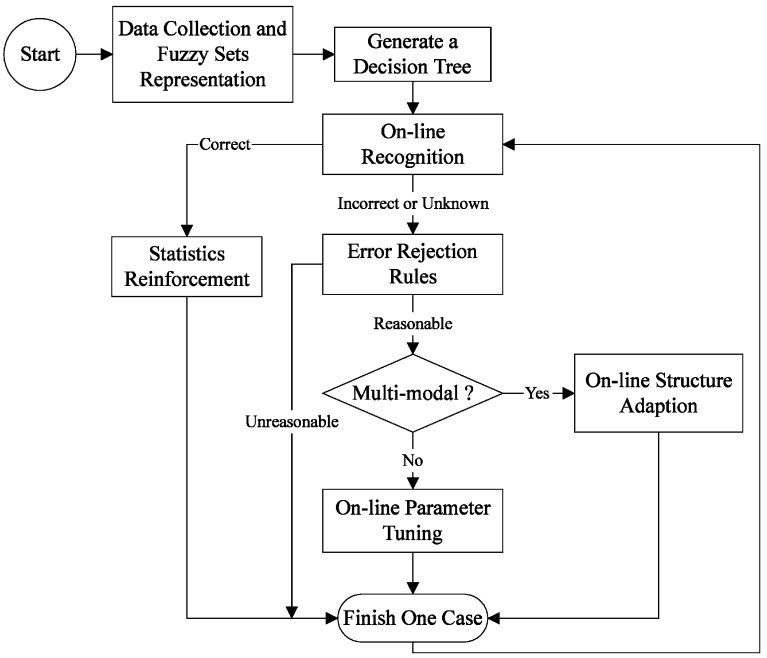

Fig. 3. The flowchart describing the concept and architecture of SOFDT.

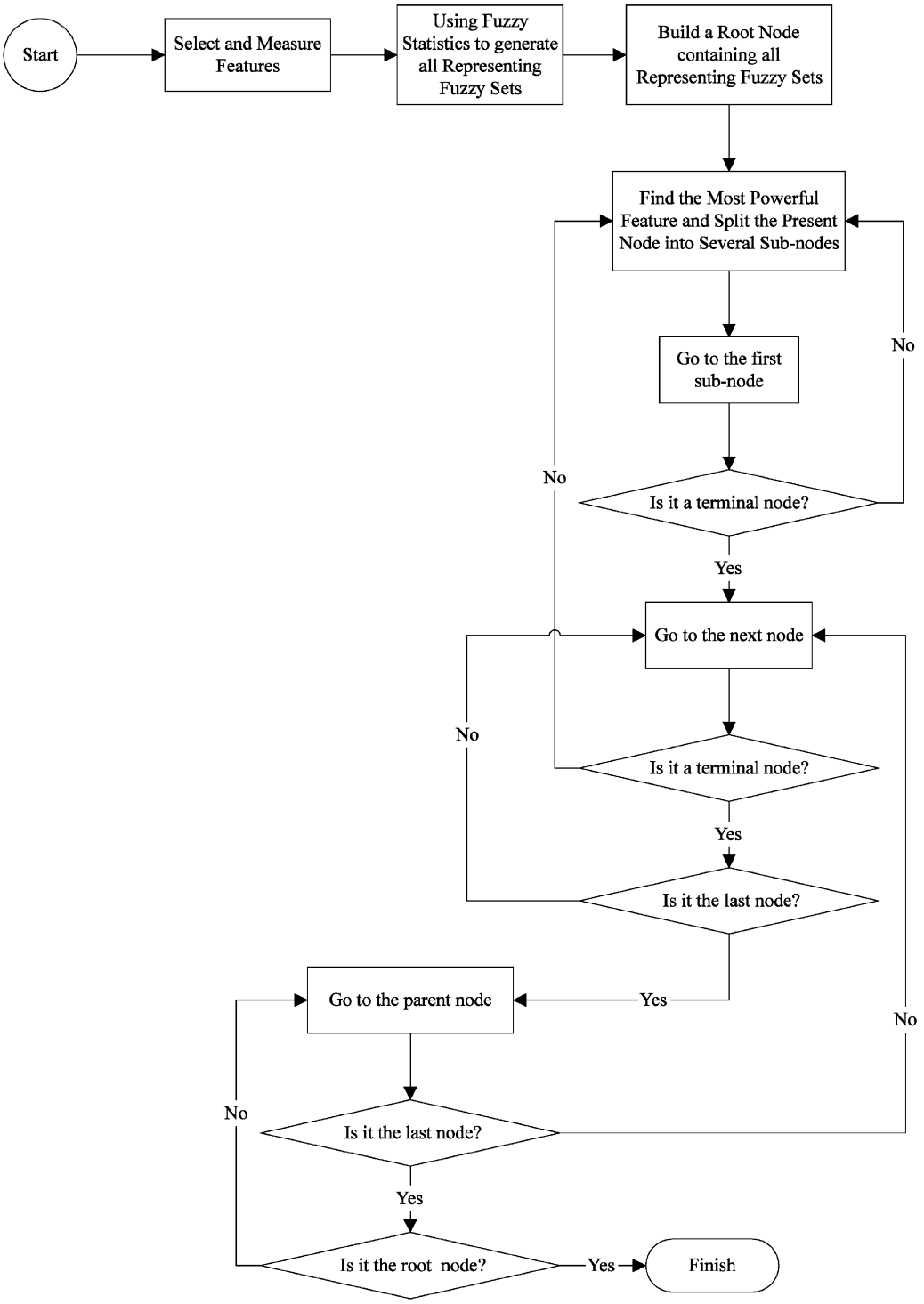

Fig. 4. The flowchart of the algorithm for determining a fuzzy decision tree.

3.2.1. Features selection

This study selected complex moment invariants as the describing features. The definition of a continuous complex moment is

$$C_{pq} = \int\int (x+iy)^p (x-iy)^q g(x,y)\,\mathrm{d}x\,\mathrm{d}y, \qquad (2)$$

where p and q are non-negative integers; $g(x, y)$ is the gray level function of the image. The corresponding discrete complex moments are:

$$C^d_{pq} = \sum_{j=1}^{n}\sum_{k=1}^{m} (j\Delta x + ik\Delta y)^p \times (j\Delta x - ik\Delta y)^q g(j\Delta x, k\Delta y)\Delta x\Delta y. \qquad (3)$$

The equation can be simplified by setting $\Delta x = \Delta y = 1$ as

$$C^{ds}_{pq} = \sum_{j=1}^{n}\sum_{k=1}^{m} (j+ik)^p (j-ik)^q g(j,k), \qquad (4)$$

where (j, k) represents the coordinates, relative to the corresponding center of pixels of the image. A set of complex moment invariants then can be obtained as

$$\begin{aligned}
S_1 &= C^{dsc}_{11}, & S_7 &= |C^{dsc}_{40}|^2,\\
S_2 &= |C^{dsc}_{20}|^2, & S_8 &= |C^{dsc}_{31}|^2,\\
S_3 &= |C^{dsc}_{30}|^2, & S_9 &= C^{dsc}_{22},\\
S_4 &= |C^{dsc}_{21}|^2, & S_{10} &= (C^{dsc}_{40})^*(C^{dsc}_{31})^2 + C.C.,\\
S_5 &= (C^{dsc}_{30})^2(C^{dsc}_{21})^* + C.C., & S_{11} &= C^{dsc}_{31}(C^{dsc}_{20})^* + C.C.\\
S_6 &= (C^{dsc}_{21})^2(C^{dsc}_{20})^* + C.C.,
\end{aligned} \qquad (5)$$

where the $*$ represents a complex conjugate, and $C.C.$ denotes the complex conjugate of its previous term.

3.2.2. Fuzzy sets for compressing training data

This study adopted *LR* type fuzzy sets for compressing and representing training data. Fuzzy statistics [12,16] were then used to determine several points of the membership functions of the fuzzy sets. Finally, the curve fitting method and the pre-determined points were used to identify the *LR* type fuzzy sets. The procedure for identifying the representative *LR* type fuzzy sets is described below.

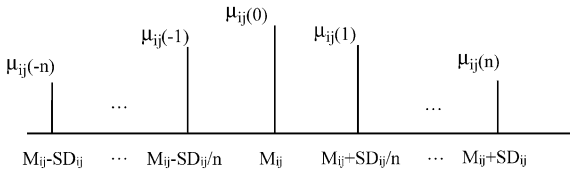

Fig. 5. $2n + 1$ points on a fuzzy set F_{ij}.

Determining several points on a fuzzy set. The training data are denoted as $T = \{T_1, T_2, \ldots, T_n\}$, where $T_i = \{T_{i1}, T_{i2}, \ldots, T_{im}\}$ is the training data for the object i, and T_{ij} is the training data for the feature j of the object i. The goal is to find fuzzy sets F_{ij} to represent T_{ij}, $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, m$. The algorithm for estimating several points on F_{ij} for identification is outlined as follows:

(1) The standard deviation and the mean of T_{ij} are calculated as SD_{ij} and M_{ij}, respectively.
(2) The number of points, $(2n + 1)$, to be identified is determined.
(3) For the $2n + 1$ points at M_{ij}, $M_{ij} \pm (1/n)SD_{ij}$, $M_{ij} \pm (2/n)SD_{ij}, \ldots, M_{ij} \pm SD_{ij}$, determine their membership grades $\mu_{ij}(k)$ with respect to F_{ij}, as shown in Fig. 5, using $\mu_{ij}(k) = S_{ij}(k)/S_{ij}(0)$.

$S_{ij}(k)$ is the number of data located in the interval $[M_{ij} + (k-1)SD_{ij}/n,\ M_{ij} + (k+1)SD_{ij}/n]$, $k = 0, \pm1, \pm2, \ldots, \pm n$. From the above equation and Fig. 5, it is apparent that $\mu_{ij}(k)$ is proportional to $S_{ij}(k)$, the number of data located around the identified point. Hence, the above equation possesses the physical meaning of fuzzy sets and statistics and $\mu_{ij}(k)$ is the desired point.

This algorithm is valid under the assumption that the training data probability distribution has the property $S_{ij}(0) > S_{ij}(k), \pm1, \pm2, \ldots, \pm n$. Under this assumption, the identified fuzzy sets are normal fuzzy sets. If the symmetrical fuzzy set assumption is further assumed, only $n + 1$ points need to be estimated.

Identifying a LR type fuzzy set. A fuzzy set M is of the *LR*-type if there exists reference functions L (for

left) and R (for right), and scalars $a>0$, $b>0$ with

$$\mu_M = \begin{cases} L\left(\dfrac{m-x}{a}\right) & \text{for } x \leqslant m, \\ R\left(\dfrac{x-m}{b}\right) & \text{for } x \geqslant m, \end{cases} \tag{6}$$

where m is the mean value of M. If the reference functions are approximated by polynomials of order n, L and R can be formulated as

$$L = a_0 + a_1 x + a_2 x^2 + \Lambda + a_n x^n,$$
$$R = b_0 + b_1 x + b_2 x^2 + \Lambda + b_n x^n. \tag{7}$$

Since we have determined in the previous sub-section $n+1$ points for L and R, respectively, the coefficients $a_0, a_1, \ldots, a_n$ and $b_0, b_1, \ldots, b_n$ can be computed by curve fitting method.

3.2.3. Generating a decision tree

The generating procedure of the fuzzy decision tree is stated in the following:

Step 1: Select a set of affecting factors suitable to partition the situation space, and then massively measure each affecting factor to get sufficient training data. The situation space is built on all the affecting factors and contains all the situations in soccer games.

Step 2: Use the algorithm proposed in Section 3.2.2 to find the fuzzy sets F_{ij}, $i=1$–n, $j=1$–m, to represent the training data. Use the fuzzy sets to form fuzzy pattern vectors $F_{\mathrm{pv}i} = (F_{i1}, F_{i2}, \ldots, F_{im},)$ for representing situation i, $i=1$–n.

Step 3: Build the root node and start the tree-generation process. The root node contains all the situations represented by $F_{\mathrm{pv}i}$, $i=1$–n.

Step 4: Find the most powerful affecting factor for the present node. An affecting factor is the most powerful factor if it can split the node into the greatest number of sub-nodes. If more than one factor has the same power, the factor requiring the least computation time is chosen.

Step 5: Split the present node into several sub-nodes using the factor selected in step 4 and the splitting rule described below.

According to a heuristic threshold T and the distances between the situations, the splitting rule clusters the situations in the present node into several groups. These groups are the content of the desired sub-nodes. The heuristic threshold T is specific to the selected feature and the current tree level (root node is level 0). It is defined by the following equation:

$$T = M_{\mathrm{SD}} \times \frac{L_{\mathrm{MAX}} - L_{\mathrm{current}}}{2}, \tag{8}$$

where M_{SD} is the mean of the standard deviation of the selected factor; L_{MAX} and L_{current} are the maximum allowable level and current level of the fuzzy decision tree under design, respectively. The distance between situations i and j can be viewed from any factor and is defined by the following equation:

$$d_k(i,j) = \alpha|M_{ik} - M_{jk}| + \beta|SD_{ik} - SD_{jk}|, \quad \alpha+\beta=1, \quad \alpha \text{ and } \beta \text{ are real}, \tag{9}$$

where $d_k(i,j)$ is the distance between situations i and j, viewed from factor k. M_{ij} and SD_{ij} are the mean and standard deviation of the training set represented by F_{ij}, and α and β are heuristically selected weighting factors for indicating the relative importance between M_{ij} and SD_{ij}. The above definition has the advantage of considering both M_{ij} and SD_{ij}. Several values for α and β may be tried out to get a satisfactory result.

Step 6: For each non-terminal node, repeat steps 4 and 5 until there are no non-terminal nodes. A node is a terminal-node if it satisfies a predefined stopping criterion; otherwise it is a non-terminal node. A node satisfies the predefined stopping criterion if it contains only one fuzzy pattern vector or more than one fuzzy pattern vector but are all too similar to be split.

Step 7: For every terminal node containing more than one fuzzy pattern vector, the next most powerful factor is linearly combined to split it. This step is repeated until each termi-

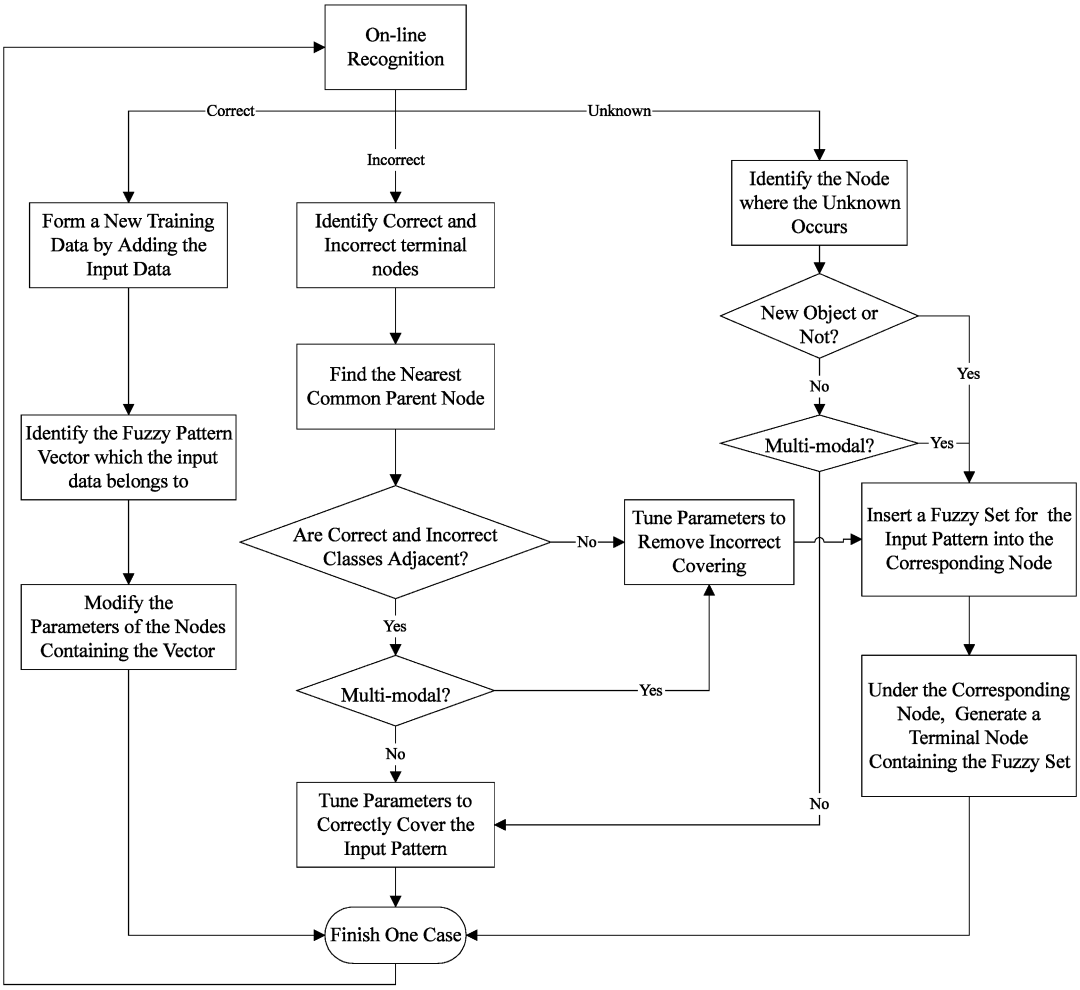

Fig. 6. The detailed flowchart of SOFDT without error-rejection.

nal node contains only one fuzzy pattern vector.

3.3. Learning law of the self-organizing fuzzy decision tree

The on-line real-time learning algorithm of SOFDT can be detailed in a flowchart, as shown in Fig. 6. The proposed on-line learning algorithm consists of three parts: correct, incorrect, and unknown recognition. The three parts are explained in the subsequent context. The term 'class' in Fig. 6 and subsequent context represents a situation in soccer games. For clarity of presentation, Fig. 6 does not show the error-rejection part of SOFDT and this part will be described in the next sub-section.

3.3.1. Part I: correct recognition

When the input data is correctly recognized, SOFDT adds it to the training data and uses the following steps to modify its parameters.

Step 1: Form a new training data by adding the input data.

Step 2: Identify the fuzzy pattern vector to which the input data belongs.

Step 3: Re-calculate the fuzzy pattern vector in step 2 with the new training data in step 1.
Step 4: Modify the nodes containing the new fuzzy pattern in step 3.
Step 5: Finish this case and restart to recognize another object.

The objective is to make the corresponding fuzzy sets more correctly represent the training data. Considering the size of the training data, it is easy to see that the modification scheme in this part only finely tunes the parameters.

3.3.2. Part II: incorrect recognition

When the input data is recognized as an incorrect class, SOFDT applies the following steps to tune its parameters or adapt its structure to accommodate itself to this kind of input data.

Step 1: Identify the terminal nodes of correct and incorrect classes of the input data.
Step 2: Find the nearest common 'parent node' of the nodes found in step 1. The definition of 'parent node' is: A node N_f is a 'parent node' of a node N_s if, and only if, N_s can be reached from the root node through N_f. The nearest common parent node is regarded as the current node.
Step 3: In the current node, check whether or not the correct and incorrect classes are adjacent. If the answer is yes, go to step 4; otherwise go to step 6.
Step 4: In the current node, check that the input pattern is single-modal or multi-modal. If the input data is single-modal, go to step 5. If the input data is multi-modal, go to step 6. The input data is single-modal if, and only if, the input data is in the 'modification zones' of both the correct and incorrect classes, as shown in Fig. 7. The input data is multi-modal if, and only if, the input data is in the 'modification zones' of the incorrect class but not correct class, as shown in Fig. 8. The 'modification zone' of a correct class is defined as a small region on the boundary but outside of the fuzzy set representing this class. The 'modification zone' of an incorrect class is defined as a small region on the boundary but inside the fuzzy set representing this class. The fuzzy set is considered to be modified if, and only if, the input data falls into the modification zone. The definition of 'modification zone' is illustrated in Figs. 7 and 8, where the correct and incorrect classes are represented by triangular fuzzy sets C_c and C_i, respectively.

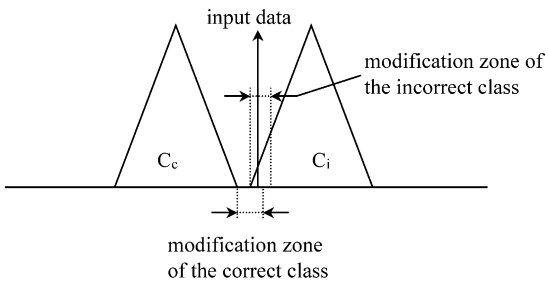

Fig. 7. The input data is single-modal.

Step 5: The key situation in the current node is shown in Fig. 7. The fuzzy set C_c and C_i are modified in a way of reducing error. After modification, go to step 9. The modification of the correct class is formulated by the following equation:

$$\text{center}(C_c') = \text{center}(C_c) + 0.5\eta_c\varepsilon_c, \quad (10)$$

$$\text{right}(C_c') = \text{right}(C_c) + \eta_c\varepsilon_c, \quad (11)$$

where C_c' is the fuzzy set representing the correct class after modification; $\text{center}(C_c)$ is the center point of C_c; $\text{right}(C_c)$ is the right boundary point of C_c; η_c is the learning rate of the correct class and in the range of $(0,1)$; ε_c is the distance from the input data to the right boundary of C_c. Similarly, the modification of the incorrect class is formulated by the following equation:

$$\text{center}(C_i') = \text{center}(C_i) + 0.5\eta_i\varepsilon_i, \quad (12)$$

$$\text{left}(C_i') = \text{left}(C_i) + \eta_i\varepsilon_i, \quad (13)$$

where C_i' is the fuzzy set representing the incorrect class after modification; $\text{left}(C_i)$ is the left boundary point of C_i; η_i is the learning rate of the incorrect class and in the range of $(0,1)$; ε_i is the distance from the

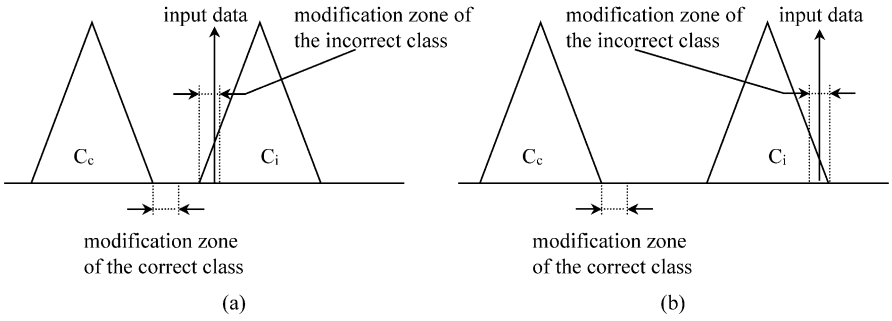

Fig. 8. The input data is multi-modal.

input data to the left boundary of C_i. If C_c is in the right side of C_i, the modification of the correct class is changed to the following equations:

$$\text{center}(C_c') = \text{center}(C_c) - 0.5\eta_c\varepsilon_c, \quad (14)$$

$$\text{left}(C_c') = \text{left}(C_c) - \eta_c\varepsilon_c. \quad (15)$$

Similarly, the modification of the incorrect class is changed to the following equations:

$$\text{center}(C_i') = \text{center}(C_i) - 0.5\eta_i\varepsilon_i, \quad (16)$$

$$\text{right}(C_i') = \text{right}(C_i) - \eta_i\varepsilon_i. \quad (17)$$

Step 6: The key situations in the current node are shown in Fig. 8, where the correct class needs not to be changed and the incorrect class is modified to reduce the error. After modification, go to step 7. The modification of the incorrect class follows Eqs. (12), (13), (16), and (17).

Step 7: Insert a fuzzy set C_{cn} into the current node, as shown in Fig. 9, and go to step 8. C_{cn} is a triangular fuzzy set and its shape is an isosceles. The center of C_{cn} is the input data, and the base of C_{cn} is double the modification zone of the correct class. Since SOFDT can recognize the similar data, inserting a fuzzy set is better than inserting a fuzzy singleton.

Step 8: Under the current node, generate a new terminal node to include the new fuzzy set in step 7 and go to step 9.

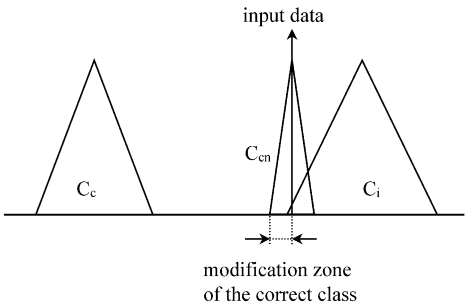

Fig. 9. The insertion of a fuzzy set for the input data in multi-modal case.

Step 9: Finish this case and restart to recognize another situation.

3.3.3. *Part III: unknown object*

SOFDT cannot recognize the input data when the representing fuzzy sets do not cover it. In this case, the input data can be single-modal, multi-modal, or represent a new situation. SOFDT applies the following steps to adapt itself to this kind of input data.

Step 1: Identify the node where the unknown case occurs and go to that node.

Step 2: Check whether or not the input data represents a new situation. If the answer is yes, go to step 5; otherwise go to step 3.

Step 3: Check whether or not the input data is multi-modal. If the answer is yes, go to step 5; otherwise go to step 4.

Step 4: Tune the parameters of the corresponding fuzzy sets of the input data in terms of Eqs. (10), (11), (14), and (15). After the parameter tuning, go to step 7.

Step 5: Insert a fuzzy set C_{cn} into the current node, as shown in Fig. 9, and go to step 6.

Step 6: Under the current node, generate a new terminal node to include the new fuzzy set in step 5 and go to step 7.

Step 7: Finish this case and restart to recognize another situation.

It can be seen from Eqs. (10)–(17) that the parameter tuning of SOFDT is a gradual learning process because the learning rate η is in the range of (0,1). It has the advantage of considering the general property of the training data. On the other hand, SOFDT needs to learn the training data several times to obtain a convergent learning result. To make SOFDT fast and accurately learn the training data, the learning rate η should be large at the beginning and gradually decrease as the training goes on.

3.4. Error rejection in learning process

If the on-line input data is erroneous or unreasonable, it will not only decrease the performance of the decision tree but also waste learning time. On the other hand, if there is a restriction on the tree size, the multi-modal input data may not always be acceptable. To overcome these problems, SOFDT uses the following error-rejection rules to filter out the unreasonable data.

Rule I: For incorrect recognition case, if the input data is not in the modification zone of the correct or incorrect class, ignore it in the corresponding modification.

Rule II: For unknown situation case, if the input data is single-modal and does not fall into the modification zone of the correct class, ignore it in the corresponding modification.

Rule III: For unknown situation case, if the input data is multi-modal or new model and adding a node will exceed the tree size restriction, ignore it in the corresponding modification.

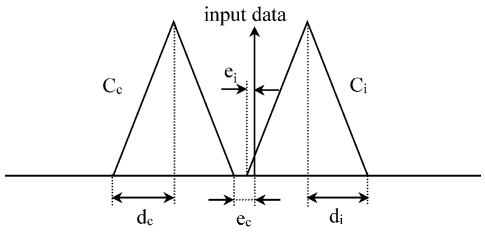

Fig. 10. The definition of recognition error.

3.5. Performance analysis

3.5.1. Convergence property of learning process

The convergence property of SOFDT can be understood by observing the recognition error of training data during the learning process. To well describe the convergence property, the recognition error E_r is defined by the following equation and shown in Fig. 10:

$$E_r = (e_c/d_c + e_i/d_i). \tag{18}$$

The change of recognition error can be obtained by the following equation:

$$\begin{aligned}\Delta E_r &= [(d_c\Delta e_c - e_c\Delta d_c)/d_c^2 + (d_i\Delta e_i - e_i\Delta d_i)/d_i^2] \\ &= [-\Delta d_c(d_c + e_c)/d_c^2 - \Delta d_i(d_i + e_i)/d_i^2] \\ &\quad \because \Delta d_c = -\Delta e_c \text{ and } \Delta d_i = -\Delta e_i. \end{aligned} \tag{19}$$

Therefore, $\Delta E_r < 0$ and the convergence of learning can be guaranteed.

3.5.2. Comparison with rule-based systems

In a decision tree, the splitting rules in a path must be simultaneously satisfied so that the terminal node of the path can be reached from the root node. Using 'and' and 'not' operators, the splitting rules of a path can be combined to form a new rule to represent the path. Repeating this procedure for every path in a decision tree, the decision tree can then be mapped into a rule-based system with each path represented by a rule. According to this mapping of structure and parameter, the learning mechanism of SOFDT can find its counterpart in its corresponding rule-based system. Adding a node in SOFDT will increase one decision path and hence one rule in its corresponding rule-based system. Tuning the parameters of SOFDT will modify the parameters of the fuzzy sets in "if" part

Table 1
Comparison between a decision tree and its corresponding rule-based system

	Decision trees	Rule-based systems
Speed	Fast	Moderate
Reliability	Moderate	Good
Storage space	Small	Large
Learning speed	Fast	Moderate
Learning easiness	Easy	Moderate

of its corresponding rule-based system. During learning process, SOFDT finds the nodes to be modified, whereas the corresponding rule-based system finds the rules to be tuned.

In every decision process of a decision tree, only the splitting rules in one path will be checked. On the other hand, in the corresponding rule-based system, every rule will be checked in every decision process, where each rule is composed of all of the splitting rules in the corresponding path. Therefore, the inference speed of a decision tree is approximately n-times faster than that of its corresponding rule-based system, where 'n' is the number of rules (or paths). SOFDT has an apparent hierarchical structure so that it can easily and fast find the nodes and its parameters to be modified. In the corresponding rule-based system, however, there are at least two rules containing any non-terminal node in SOFDT. Therefore, it is not easy for the corresponding rule-based system to find the rules and its parameters to be tuned. The comparison between a decision tree and its corresponding rule-based system is summarized in Table 1.

4. A self-organizing strategy-based decision-making system

The SOFDT presented in Section 3 is embedded in the decision-making system of a multi-agent soccer robot system. However, the following transformations must be performed.

- *Initial decision tree*: The decision tree constructed in Section 2 is used as the initial decision tree in SOFDT. The tree-generation algorithm presented in Section 3 can then be applied to generate the desired initial decision tree from the obtained training data.
- *Correct recognition*: The correct recognition in SOFDT corresponds to the situation in the strategy-based decision-making system where the strategy works.
- *Incorrect recognition*: The incorrect recognition in SOFDT corresponds to the situation in the strategy-based decision-making system where the strategy does not work and the right strategy is known.
- *Unknown object*: The unknown case in SOFDT corresponds to the situation in the strategy-based decision-making system where the strategy does not work but the right strategy is not known.
- *Multi-modal*: The multi-modal in SOFDT corresponds to the situation in the strategy-based decision-making system where some strategy works in separate regions of the situation space.
- *Single-modal*: The single-modal in SOFDT corresponds to the situation in the strategy-based decision-making system where some strategy works only in one region.
- *Parameter tuning*: The parameter tuning in SOFDT corresponds to the situation in the strategy-based decision-making system where a different strategy should be used in some terminal nodes. For example, "a robot's action changes from shooting at middle goal to shooting at left-most goal" requires a different strategy at some terminal nodes.
- *Structure adaptation*: The structure adaptation in SOFDT corresponds to the situation in the strategy-based decision-making system where a different partition of the situation space should be performed. For example, the positions of opponents in offense zone will partition the original offense zone into offense zone with no opponent, offense zone with one near-front opponent, offense zone with one far-front opponent, etc.
- *New object*: The new object in SOFDT corresponds to the situation in the strategy-based decision-making system where a new strategy is required in some terminal nodes. For example, "a robot's action changes from shooting to passing" requires a new strategy at some terminal nodes.

Using the basic decision tree in Fig. 2 as an example, the proposed self-organizing strategy-based decision-making system is illustrated as follows. In a soccer game, the basic decision tree encounters three problems. First, the offense-robot controls the ball in

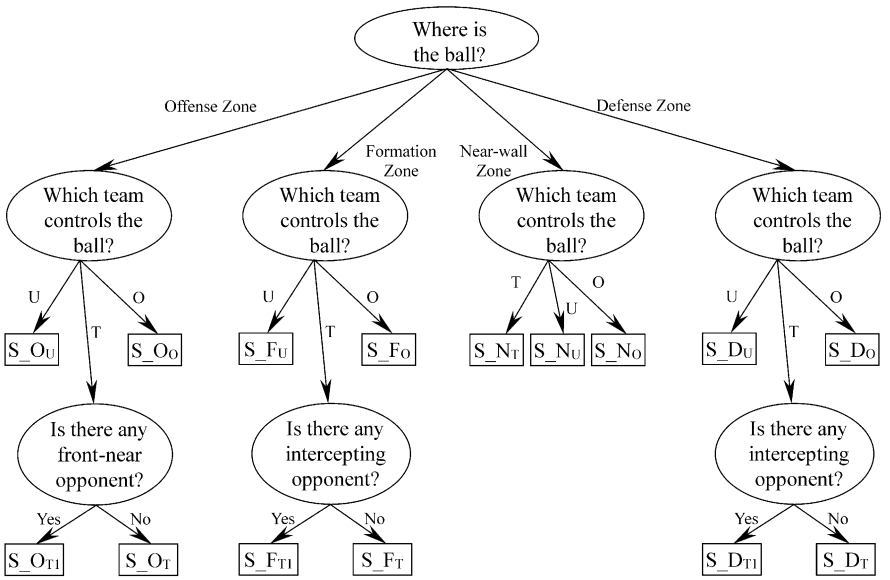

Fig. 11. The improved decision tree for the strategy-based decision-making system.

offense zone and then shoot the goal, but it fails. The reason is that one opponent in offense zone blocks the ball. Second, the formation-robot controls the ball in formation zone and then passes it to the defense-robot in offense zone, but it fails. The reason is that one opponent between the formation-robot and defense-robot intercepts the ball. Third, the defense-robot in defense zone controls the ball and passes it to the formation-robot in formation zone, but it fails. The reason is that one opponent between the defense-robot and formation-robot intercepts the ball. According to the results of these strategies and encountered problems, the proposed self-organizing strategy-based decision-making system adapts its structure from Figs. 2 to 11. In Fig. 11, U represents the ball is uncontrolled, T represents the ball is under teammate's control, and O represents the ball is under opponent's control. Three extra strategies are self-generated in Fig. 11 to cope with the above-mentioned problems. The strategies S_O_O, S_O_T, S_O_U, S_F_O, S_F_T, S_F_U, S_N_O, S_N_T, S_N_U, S_D_O, S_D_T, S_D_U in Fig. 11 are the same as those in Fig. 2. The three self-generated strategies in Fig. 11 are listed in the following:

1. S_O_{T1}: The *offense-robot* passes the ball in offense zone to the formation-robot. The *formation-robot* rushes into offense zone and tries to shoot the ball. The *defense-robot* defends the goal.
2. S_F_{T1}: The *offense-robot* prepares to receive the ball. The *formation-robot* dribbles the ball into offense zone. The *defense-robot* defends the goal.
3. S_D_{T1}: The *offense-robot* prepares to receive the ball. The *formation-robot* prepares to intercept the ball. The *defense-robot* goal-keeps the ball to a free space.

The soccer skills 'dribble' and 'goal-keep' can refer to [9].

5. Simulations and experiments

The proposed self-organizing strategy-based decision making system was applied to the 3-to-3 robot soccer games. Both simulations and experiments were performed to justify its performance. In simulation, 30 runs of 3-to-3 robot soccer games were conducted in a soccer robot simulator developed by the authors [5,9]. The duration of each run was 5 minutes. The simulator was developed using Visual C++ with OpenGL as

Fig. 12. Three-to-three soccer game where the 'logo' team is attacking.

Fig. 13. Experiment of 3-to-3 soccer robot game.

the API for graphics hardware. The simulations were conducted on Pentium III 450 with TNT2 Ultra graphics engine as the video accelerator.

The snapshot of the conducted 3-to-3 robot soccer games is shown in Fig. 12, where the 'logo' team is in charge of defending the left goal and attacking the right goal while the 'head' team has the reverse duties as 'logo' team. In the figure, the 'logo' team is attacking the right goal. In the conducted simulations, two teams were equipped with the same decision-making system. The average time to goal is 12.5 s. Effective offense and defense were observed in each team. Fig. 13 shows the robots on the field. The decision-making system off-line learned the data collected from the image system. Based on the 100 training data, the decision system generated 20 strategies. Then the decision system on-line performed decision-making. A decision-making, including generating a new strategy, is less than 7 ms. These results are quite satisfactory.

6. Conclusions

A self-organizing fuzzy decision tree (SOFDT) was developed in this study. Comparing to traditional decision trees, the proposed SOFDT has less memory space and faster generation speed and possesses the real-time on-line adapting capacity. The proposed SOFDT was integrated with a strategy-based decision-making system to select among the developed strategies according to the situation of the soccer field. The proposed decision-making system is more flexible and effective than other type of decision-making systems. NTU-Formosa performs centralized coordination control to coordinate the multiple mobile robots in accordance with the adopted strategies. From the simulation and experiment of 3-to-3 soccer games, the results are satisfactory.

References

[1] R.A. Barman, S.J. Kingdon, J.J. Little, A.K. Mackworth, D.K. Pai, M. Sahota, H. Wilkinson, Y. Zhang, DYNAMO: real-time experiments with multiple mobile robots, Proc. of Intelligent Vehicles Symp., Tokyo, 1993, pp. 261–266.

[2] H.P. Huang, C.C. Liang, A learning fuzzy decision tree and its application to tactile image, Int. Conf. on Intelligent Robots and Systems (IROS'98), 1998, pp. 1578–1583.

[3] H.P. Huang, C.C. Liang, Development of A fuzzy learning decision tree and application to tactile recognition, IEEE Trans. Fuzzy System, submitted for publication.

[4] H.P. Huang, C.C. Liang, Intelligent game strategies based on soccer dynamics for a multi-agent soccer robot system, J. Robot. Systems, submitted for publication.

[5] H.P. Huang, C.C. Liang, C.W. Lin, Building a multi-agent robot system for robot soccer game and developing game strategies based on soccer dynamics, Int. Conf. on Mechatronic Technology, 1998, pp. 303–308.

[6] H.P. Huang, C.C. Liang, C.W. Lin, Construction and soccer dynamics analysis for an intelligent multi-agent soccer robot system, Proc. of the National Science Council, R.O.C. Part A: Physical Science and Engineering, 25(2) (2001) 84–93.

[7] J.H. Kim, H.S. Shim, H.S. Kim, M.J. Jung, I.H. Choi, J.O. Kim, A cooperative multi-agent system and its real time application to robot soccer, IEEE Int. Conf. on Robotics and Automation, 1997, pp. 638–643.

[8] K.H. Kim, K.W. Ko, J.G. Kim, S.H. Lee, H.S. Cho, The development of a micro robot system for robot soccer game, IEEE Int. Conf. on Robotics and Automation, 1997, pp. 644–649.
[9] C.C. Liang, Development of a multiple-soccer-robot system, Ph.D. Dissertation, Department of Mechanical Engineering, National Taiwan University, 2000.
[10] Y. Park, J. Sklansky, Automated design of linear tree classifiers, Pattern Recognition 23 (12) (1990) 1393–1412.
[11] S.W. Park, J.H. Kim, E.H. Kim, J.H. Oh, Development of a multi-agent system for robot soccer game, IEEE Int. Conf. on Robotics and Automation, 1997, pp. 626–631.
[12] W. Pedrycz, Fuzzy sets in pattern recognition: methodology and methods, Pattern Recognition 23 (1–2) (1990) 121–146.
[13] I.K. Sethi, Entropy nets: from decision trees to neural networks, Proc. IEEE 78 (10) (1990) 1605–1613.
[14] W.M. Shen, J. Adibi, R. Adobbati, B. Cho, A. Erdem, H. Moradi, B. Salemi, S. Tejada, Building integrated mobile robots for soccer competition, IEEE Int. Conf. on Robotics and Automation, 1998, pp. 2613–2618.
[15] W.N. Shen, S.C. Lee, C.W. Lin, Y.B. Chen, Visual servoing control of multi-agent robot on playing robot soccer game, Bull. College Eng. N.T.U. (73) (1998) 135–144.
[16] P.Z. Wang, From the fuzzy statistics to the falling random subsets, in: P.P. Wang (Ed.), Advances on Fuzzy Sets Theory and Applications, Pergamon Press, New York, 1983, pp. 81–95.