Decision Maker for Robot Soccer[1]

H. K. Lam
Centre for Multimedia Signal Processing, Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
hkl@eie.polyu.edu.hk

T. H. Lee
Centre for Multimedia Signal Processing, Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
thlee@eie.polyu.edu.hk

F. H. F. Leung
Centre for Multimedia Signal Processing, Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
enfrank@polyu.edu.hk

P. K. S. Tam
Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
enptam@hkpucc.polyu.edu.hk

Abstract **This paper presents a decision maker, which will select the appropriate tactics and actions for soccer robots according to the condition of a tournament. The selected tactic will be employed to assign each home robot an action to play the game. The decision maker consists of a tactic database, an action database, a tactic selection algorithm, a tactic selector and an action selector. Simulation results will be given to show the merits of the proposed algorithm.**

I. Introduction

Robot soccer tournament is a multi-agent system that involves the cooperation among agents, the role assignment to each agent and the action taken by each agent. For a multi-agent system, making decision is important to the efficient completion of a task. The difficulties on making decision are: 1) There is not a systematic and simple way to represent human way of thinking and knowledge by a model or a programming language, and training algorithms to evolve the decision making algorithm are hard to be derived; 2) The number of varying environmental and physical factors is large. It is difficult to choose suitable variables to serve as inputs of the decision maker. A large number of variables will make the decision-making algorithm complex and the computational demand will be high.

The micro robot soccer tournament (MiroSot) is used as a testing paradigm in this paper. A decision maker will be proposed for the MiroSot. MiroSot is a robot soccer game [1] that involves six robots, three in each team. Fig. 1 shows the situation of the MiroSot. Images of the game playing in the playground are captured by the visual system and sent to the host computer. The information of the game, e.g., the positions, angles and velocity of the robots, will be extracted. This information will be used by the game strategy implemented as a software program in the host computer. The outputs of the game strategy are commands transmitted to the three robots of the corresponding team that tell them how to play the game. The game strategy involves making a decision [2-3, 8-9] on what action to be taken, planning a path for the robot [6-7, 10], generating commands to control the robot [5-6]. Each part of the game strategy is important for wining the game. In particular, the decision-making is the brain of the robots and making a right decision is the basis of the team intelligence.

The contributions of this paper are threefold. First, a simple structure of the decision maker is proposed. It consists of four units, namely tactic database, tactic selection algorithm, tactic selector and action selector. Second, the decisions made by the proposed decision maker is similar to those made by human expert as the tactic and action databases are rules based on human expert knowledge. Third, a tactic selection algorithm is proposed to select the most appropriate tactic from the database. Different situations of the competition can be coped with by using different tactics.

II. Decision Maker

As shown in Fig. 2., the decision maker comprises a tactic database, a tactic selection algorithm, a tactic selector and an action selector. It first takes its inputs such as the positions of the home and opponent robots. These inputs are used by the tactic selector to select the best tactic based on the tactic database and the tactic selection algorithm. The selected best tactic will be fed into the action selector. The action selector will search the action database and provide the actions output to each home robot (except the goalie). The details about each unit are as follows.

A. Tactic Database

The tactic database stores the attack and defense tactics used by the robot soccer. More than one set of attack and defense tactics can be used in the tactic database to cope with different situations of the competition. The tactic database is derived from expert knowledge. As a result, the decision made will be similar to the decision made by human. In this paper, the tactics are applicable to the robots except the goalie.

A.1. Attack Tactics

The attack tactics are some sets of expert rules to govern the attacking actions of the home robots. These attack tactics will be used during the competition. To develop the attack tactics, the playground is first divided into a number of regions as shown in Fig. 3. The region size for each region can be different. In Fig. 3, the playground is divided into *num_regions* regions, which is a constant integer determined by the designer. The region numbering is from 1 (assigned to the region which is nearest to the opponent goal), to *num_regions* (assigned to the region which is nearest to the home goal). The format of the *i*-th attack tactic, $\mathbf{A}_i$, of the tactic database is as follows,

[1] The work described in this paper was substantially supported by a Research Grant of The Hong Kong Polytechnic University (project number B-Q394).

0-7803-7108-9/01/$10.00 (C)2001 IEEE

$$\mathbf{A}_i = \begin{bmatrix} A_1^i \\ A_2^i \\ \vdots \\ A_{num\_regions\times 4}^i \end{bmatrix}, i = 1, 2, \ldots, num\_attack\_tactics \quad (1)$$

$$A_j^i = [reg\_robot1 \ \ see\_goal \ \ see\_robot2 \ \ action1 \ \ reg\_robot2 \ \ action2] \quad (2)$$

i = 1, 2, ..., *num_attack_tactics*; j = 1, 2, ..., *num_regions*×4.

where,

i) A_j^i denotes the j-th rule of the i-th attack tactic.

ii) *num_attack_tactics*, which is a constant positive integer, denotes the number of attack tactics used in the attack tactic database. The value of *num_attack_tactics* is determined by the designer.

iii) $reg\_robot1 \in \{1 \ \ 2 \ \ \cdots \ \ num\_regions\}$ denotes the region of the home robot having the ball.

iv) *see_goal* ∈ {no yes} denotes whether the home robot having the ball can see the opponent goal (*see_goal* = yes) or not (*see_goal* = no).

v) *see_robot2* ∈ {no yes} denotes whether the home robot having the ball can see its member robot (not the goalie) or not.

vi) *action1* ∈ {blocking catching dribbling guarding passing shooting waiting} denotes the action that will be taken by the home robot having the ball. These actions are defined as follows,
Blocking - The commanded robot is to block the ball being shot to the home goal.
Catching - The commanded robot will catch the ball.
Dribbling - The commanded robot will dribble the ball.
Guarding - The commanded robot will guard the opponent robot without the ball in order to prevent the ball from reaching it.
Passing - The commanded robot will pass the ball to other member robots.
Shooting - The commanded robot will shoot the ball towards the opponent goal.
Waiting - The commanded robot will wait for the chance to take or shoot the ball at the assigned position.

vii) $reg\_robot2 \in \{1 \ \ 2 \ \ \cdots \ \ num\_regions\}$ denotes the region that the member robot will go.

viii) *action2* ∈ {blocking catching dribbling guarding passing shooting waiting} denotes the action that will be taken by the member robot.

It can be seen that the first three parameters in (2) are inputs of the attack tactic A_j^i while the last three parameters are output commands to the home robot. As the first three parameters in (2) are the inputs of the attack rule A_j^i, we have *num_regions*×2×2 rules for each attacking tactic.

A.2. Defense Tactics

The defense tactics are some sets of expert rules to govern the defense actions of the home robots. The defense tactics will be used when an opponent robot has the ball. The format of the i-th defense tactic, $\mathbf{D}_i$, of the tactic database is as follows,

$$\mathbf{D}_i = \begin{bmatrix} D_1^i \\ D_2^i \\ \vdots \\ D_{num\_regions^2}^i \end{bmatrix}, i = 1, 2, \ldots, num\_defense\_tactics \quad (3)$$

$$D_j^i = [reg\_opp1 \ \ reg\_opp2 \ \ reg\_robot1 \ \ action1 \ \ reg\_robot2 \ \ action2] \quad (4)$$

i = 1, 2, ..., *num_defense_tactics*; j = 1, 2, ..., *num_regions*^{2}.

where,

i) D_j^i denotes the j-th rule of the i-th defense tactic.

ii) *num_defense_tactics*, which is a constant positive integer, denotes the number of defense tactics used in the defense tactic database. The value of *num_defense_tactics* is determined by the designer.

iii) $reg\_opp1 \in \{1 \ \ 2 \ \ \cdots \ \ num\_regions\}$ denotes the region of the playground that the opponent robot is in.

iv) $reg\_opp2 \in \{1 \ \ 2 \ \ \cdots \ \ num\_regions\}$ denotes the region of the playground that the other opponent robot is in.

v) $reg\_robot1 \in \{1 \ \ 2 \ \ \cdots \ \ num\_regions\}$ denotes the region of the playground that the home robot will go.

vi) *action1* ∈ {blocking catching dribbling guarding passing shooting waiting} denotes the action that the home robot will take.

vii) $reg\_robot2 \in \{1 \ \ 2 \ \ \cdots \ \ num\_regions\}$ denotes the region that the other home robot will go.

viii) *action2* ∈ {blocking catching dribbling guarding passing shooting waiting} denotes the action that will be taken by the member robot.

It can be seen that the first two parameters of (4) are inputs while the last four parameters are output commands to the home robot. As the first two parameters of (4) are inputs of the defense rule D_j^i, we have *num_regions*× *num_regions* rules for each defense tactic.

B. Tactic Selector

The tactic selector is to select the most appropriate attack or defense tactic from the tactic database according to the tactic selection algorithm. When the ball is held by the home team, the tactic selector will select the most appropriate attacking tactic, $\mathbf{A}_i$, for attacking. When the ball is held by the opponent team, the tactic selector will select the most appropriate defense tactic, $\mathbf{D}_i$, for defending.

C. Tactic Selection Algorithm

A tactic selection algorithm is proposed to help the tactic selector to select an appropriate attack or defense tactic from the tactic database.

C.1. Tactic Selection Algorithm under Attack Mode

In this sub-section, an algorithm will be presented to select an appropriate attack tactic $\mathbf{A}_i$, $i \in \{1, 2, \ldots, num\_attack\_tactics\}$. Let H_k and O_k, $k = 1, 2, \ldots, num\_regions$, be the accumulated numbers of times the home and the opponent robots holding the ball for a time unit (defined by the designer) in region i of the playground respectively. For a sampling period of 1 second for instance, if in the first second, the home robot with ball is in region 4, then $H_4 = 1$; if in the next second, the home robot with ball is in region 3, then $H_3 = 1$; if in the third second, the home robot with ball is in region 4 again, then $H_4 = 2$. The content of O_k for the opponent team is obtained by the same process. We assume that the degree of aggression of $\mathbf{A}_i$ increases in ascending order of i, i.e. $\mathbf{A}_1$ is the least aggressive attacking tactic and $\mathbf{A}_{num\_attacking\_tactics \times 4}$ is the most aggressive attacking tactic. Let $\mathbf{A}_E$ be the used attacking tactic at the present moment. The initial attacking tactic of $\mathbf{A}_E$ can be any one of the $\mathbf{A}_i$. To update $\mathbf{A}_E$, a random positive floating-point number, r_A, is generated for every time unit. If the current $\mathbf{A}_E = \mathbf{A}_i$, $\mathbf{A}_E$ will be updated according to the following conditions,

$$\begin{cases} \mathbf{A}_E = \mathbf{A}_{i+1} \text{ if } r_A < p_A^{\min} \\ \mathbf{A}_E = \mathbf{A}_i \text{ if } p_A^{\min} \le r_A \le p_A^{\max} \\ \mathbf{A}_E = \mathbf{A}_{i-1} \text{ if } r_A > p_A^{\max} \end{cases} \quad (5)$$

$$p_A^{\max} \in [0 \quad 1] = \max\left[\frac{H_{current\_region}}{\sum_{k=1}^{num\_regions} H_k} \quad 1-\frac{O_{current\_region}}{\sum_{k=1}^{num\_regions} O_k}\right] \quad (6)$$

$$p_A^{\min} \in [0 \quad 1] = \min\left[\frac{H_{current\_region}}{\sum_{k=1}^{num\_regions} H_k} \quad 1-\frac{O_{current\_region}}{\sum_{k=1}^{num\_regions} O_k}\right] \quad (7)$$

where *current_region* denotes the region that the home robot with ball is in at the current moment. The value of $\frac{H_{current\_region}}{\sum_{k=1}^{num\_regions} H_k}$ denotes the probability of attack of the home team at the region *current_region*. The value of $1-\frac{O_{current\_region}}{\sum_{k=1}^{num\_regions} O_k}$ denotes the probability of attack of the opponent team at the region *current_region*. $p_A^{\max}$ and $p_A^{\min}$ are the maximum and minimum probabilities of attack respectively. It should be noted that when $i+1 \ge num\_attacking\_tactics$ in (5), $\mathbf{A}_{i+1} = \mathbf{A}_{num\_attack\_tactics}$, when $i-1 \le 0$ in (5), $\mathbf{A}_{i-1} = \mathbf{A}_1$.

C.2. Tactic Selection Algorithm under Defense Mode

We assume that the degree of the defense of $\mathbf{D}_i$ increases in ascending order of i, i.e. $\mathbf{D}_1$ is the least defensive tactic and $\mathbf{D}_{num\_defense\_tactics}$ is the most defensive tactic. Let $\mathbf{D}_E$ be the defensive tactic at the current moment. The initial defense tactic of $\mathbf{D}_E$ can be any one of the $\mathbf{D}_i$. To update $\mathbf{D}_E$, a random positive floating-point number, r_D, is generated for every time unit. If the current $\mathbf{D}_E = \mathbf{D}_i$, $\mathbf{D}_E$ will be updated according to the following conditions,

$$\begin{cases} \mathbf{D}_E = \mathbf{D}_{i+1} \text{ if } r_D < p_D^{\min} \\ \mathbf{D}_E = \mathbf{D}_i \text{ if } p_D^{\min} \le r_D \le p_D^{\max} \\ \mathbf{D}_E = \mathbf{D}_{i-1} \text{ if } r_D > p_D^{\max} \end{cases} \quad (8)$$

$$p_D^{\max} \in [0 \quad 1] = \max\left[1-\frac{H_{current\_region}}{\sum_{k=1}^{num\_regions} H_k} \quad \frac{O_{current\_region}}{\sum_{k=1}^{num\_regions} O_k}\right] \quad (9)$$

$$p_D^{\min} \in [0 \quad 1] = \min\left[1-\frac{H_{current\_region}}{\sum_{k=1}^{num\_regions} H_k} \quad \frac{O_{current\_region}}{\sum_{k=1}^{num\_regions} O_k}\right] \quad (10)$$

The value of $1-\frac{H_{current\_region}}{\sum_{k=1}^{num\_regions} H_k}$ denotes the probability of defense of the home team at the region *current_region*. The value of $\frac{O_{current\_region}}{\sum_{k=1}^{num\_regions} O_k}$ denotes the probability of defense of the opponent team at the region *current_region*. $p_D^{\max}$ and $p_D^{\min}$ are the maximum and minimum probabilities of attacking. It should be noted that when $i+1 \ge num\_defense\_tactics$ in (8), $\mathbf{D}_{i+1} = \mathbf{D}_{num\_defense\_tactics}$, when $i-1 \le 0$ in (8), $\mathbf{D}_{i-1} = \mathbf{D}_1$.

D. Action Selector

The action selector is to determine the actions for the home robots according to the selected output attack or defense tactic and some information of the robots. The outputs of the action selector are the actions of the home robots.

D.1. Action Selector under Attack Mode

The outputs of the action selector under attack mode are the attack actions of the home robots. The attack mode will only be triggered when the home team has the ball. The action selector will first select the appropriate rule from the attack tactic $\mathbf{A}_i$ (which is the output of the tactic selector as well as a part of the inputs of the action selector). To single out the rule from $\mathbf{A}_i$, some information of the robots (some inputs of the action selector), namely *reg_robot1*, *see_goal*, *see_robot2*, are needed. The rule A_j^i will be singled out if the input values match the values of *reg_robot1*, *see_goal* and *see_robot2* of A_j^i. For instance, we consider the following attack tactic with two rules :

$$\mathbf{A}_1 = \begin{bmatrix} A_1^1 \\ A_2^1 \end{bmatrix} = \begin{bmatrix} 1 & \text{no} & \text{yes} & \text{passing} & 2 & \text{catching} \\ 2 & \text{yes} & \text{no} & \text{shooting} & 1 & \text{waiting} \end{bmatrix}.$$

If the home robot having the ball is in the region 1 cannot see the opponent goal but can see its member robot, i.e. *reg_robot*1 = 1, *see_goal* = no and *see_robot2* = yes, the first rule A_1^1 is selected. Then, the action taken by the home robot having the ball is to pass the ball to the member robot and the action of the member robot is going to catch the ball at region 2.

D.2. Action Selector under Defense Mode

The outputs of the action selector under defense mode are the defense actions of the home robots. The defense mode of the action selector will only be triggered when the home team does not have the ball. To determine the actions of the home robots under the defense mode, the action selector will first select the appropriate rule from the defense tactic $\mathbf{D}_i$ (which is the output of the tactic selector as well as a part of the inputs of the action selector). To single out the rule from $\mathbf{D}_i$, some information of the robots (some inputs of the action selector), namely *reg_opp*1 and *reg_opp*2, are needed. The rule D_j^i will be singled out if the input values match the values of *reg_opp*1 and *reg_opp*2 of D_j^i. For instance, we consider the following defense tactic:

$$\mathbf{D}_1 = \begin{bmatrix} D_1^1 \\ D_2^1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & \text{catching} & 2 & \text{waiting} \\ 2 & 4 & 2 & \text{cartching} & 4 & \text{guarding} \end{bmatrix}.$$

If the opponent robot having the ball and the other opponent robot (not the opponent goalie) are both in region 1, i.e. *reg_opp*1 = 1 and *reg_opp*2 = 1, D_1^1 will be selected. Then, the action taken by the home robot is to catch the ball in region 1 and that of its member robot (not the home goalie) is to wait at region 2.

III. Simulation Results

Simulations will be carried out to test the merits of the proposed decision maker. In the simulations, the home team will employ the proposed decision maker while the opponent team does not. The numbers of attack and defenses tactics are both 2 ($\mathbf{A}_1$ and $\mathbf{A}_2$, $\mathbf{D}_1$ and $\mathbf{D}_2$). The playground is divided into *num_regions* = 5 even regions. The attack and defense tactics are listed in Table I to Table IV. Table V shows the simulation results when the home team uses and does not use the proposed decision maker. The initial attack and defense tactics are $\mathbf{A}_1$ and $\mathbf{D}_1$. It can be seen that the total scores of the home team with the proposed decision maker is higher than that of the opponent team. The scores of both teams are similar when both teams use the same sets of attack and defense tactics ($\mathbf{A}_1$ and $\mathbf{D}_1$) without using the proposed decision maker.

4. Conclusion

A decision maker for the robot soccer has been proposed. The proposed decision maker consists of four units, namely tactic database, tactic selection algorithm, tactic selector and action selector. Each unit has been detailed in this paper. As the rules of the tactic databases are based on expert knowledge, the decision made by this decision maker is similar to that made by an expert. A simple algorithm has been proposed to select an appropriate attack or defense tactics to cope with different tactics of the opponent team. Simulation results have been presented which show that the team using the proposed decision maker can get a higher score than the team without using the proposed decision maker.

References:

[1] Hiroaki Kitano (ed.), *Robot Soccer World Cup I / RobotCup-97.*, New York: Spring-Verlag, Berlin, 1998.

[2] H. Soo Kim, H. Sik Kim, M.J. Jung, J. H. Kim, "Action selection mechanism for soccer robot," *IEEE International Symposium on Computational Intelligence in Robotics and Automatiom*, 1997, pp. 390-395

[3] H. Soo Kim, H. Sik Kim, M. J. Jung, J.H. Kim, "Action selection and strategy in robot soccer systems," *Proc. of the 40 Midwest Symposium on circuits and systems*, Vol: 1.1, 1998, pp.518-521.

[4] Ronald C. Arkin., *Behavior-Based Robotics*, London, England: The MIT Press, 1998.

[5] J. Yen and R. Langari, *Fuzzy Logic: Intelligence, Control, and Information.* NJ: Prentice-Hall, 1998.

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

[7] J. W. Kim, K. C. Kim, D. H. Kim, Y. J. Kim and P., Vadakkepat, "Path planning and role selection machanism for soccer robots," *Proceedings of Robotics and Automation*, vol. 4, pp. 3216-3221, 1998.

[8] J. H. Kim, H. S. Shim, H. S. Kim, M. J. Jung and P. Vadakkepat, "Action selection and strategies in robot soccer systems," *Proceedings of the 40th Midwest Symposium on Circuits and Systems 1997*, vol. 1, pp. 518-521, 1997.

[9] H. S. Kim, H. S. Shim, M. J. Jung and J. H. Kim, "Action selection mechanism for soccer robot," *Proceedings of 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation, 1997, CIRA'97*, pp. 390-395, 1997.

[10] T. H. Lee, H. K. Lam, F. H. F. Leung and P. K. S. Tam, "A Fast Path Planning-and-Tracking Control for Wheeled Mobile Robots," *Proceedings of 2001 IEEE International Conference on Robotics and Automation (ICRA'2001)*, Seoul, Korea, 21-26 May, 2001, pp. 1736-1741.

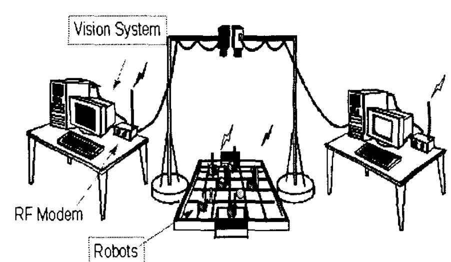

Fig. 1. Setup of the MiroSot.

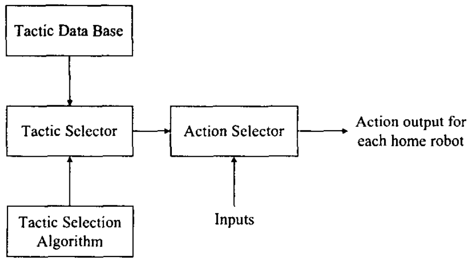

Fig. 2. Block diagram of the decision maker.

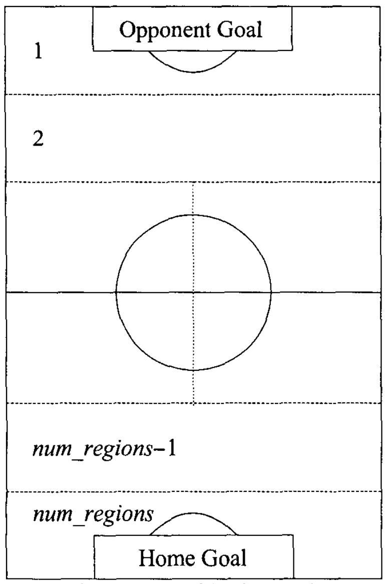

Fig. 3. Regions of the playground.

A_j^i	reg_ robot1	see_ goal	see_ robot2	action1	reg_ robot2	action2
A_1^1	1	No	No	Shooting	2	Waiting
A_2^1	1	No	Yes	Passing	2	Catching
A_3^1	1	Yes	No	Shooting	1	Waiting
A_4^1	1	Yes	Yes	Shooting	1	Waiting
A_5^1	2	No	No	Shooting	3	Waiting
A_6^1	2	No	Yes	Passing	3	Catching
A_7^1	2	Yes	No	Shooting	1	Waiting
A_8^1	2	Yes	Yes	Shooting	1	Waiting
A_9^1	3	No	No	Dribbling	4	Waiting
A_{10}^1	3	No	Yes	Passing	4	Catching
A_{11}^1	3	Yes	No	Shooting	2	Waiting
A_{12}^1	3	Yes	Yes	Shooting	2	Waiting
A_{13}^1	4	No	No	Dribbling	5	Waiting
A_{14}^1	4	No	Yes	Passing	5	Catching
A_{15}^1	4	Yes	No	Shooting	1	Waiting
A_{16}^1	4	Yes	Yes	Shooting	1	Waiting
A_{17}^1	5	No	No	Dribbling	4	Waiting
A_{18}^1	5	No	Yes	Passing	4	Catching
A_{19}^1	5	Yes	No	Shooting	1	Waiting
A_{20}^1	5	Yes	Yes	Shooting	1	Waiting

Table. I. Attack tactic $\mathbf{A}_1$.

A_j^i	reg_ robot1	see_ goal	see_ robot2	action1	reg_ robot2	action2
A_1^2	1	No	No	Shooting	2	Waiting
A_2^2	1	No	Yes	Shooting	2	Waiting
A_3^2	1	Yes	No	Shooting	1	Waiting
A_4^2	1	Yes	Yes	Shooting	1	Waiting
A_5^2	2	No	No	Shooting	3	Waiting
A_6^2	2	No	Yes	Shooting	3	Waiting
A_7^2	2	Yes	No	Shooting	1	Waiting
A_8^2	2	Yes	Yes	Shooting	1	Waiting
A_9^2	3	No	No	Dribbling	2	Waiting
A_{10}^2	3	No	Yes	Passing	2	Catching
A_{11}^2	3	Yes	No	Shooting	1	Waiting
A_{12}^2	3	Yes	Yes	Shooting	1	Waiting
A_{13}^2	4	No	No	Passing	2	Waiting
A_{14}^2	4	No	Yes	Passing	2	Catching
A_{15}^2	4	Yes	No	Shooting	1	Waiting
A_{16}^2	4	Yes	Yes	Shooting	1	Waiting
A_{17}^2	5	No	No	Passing	2	Waiting
A_{18}^2	5	No	Yes	Passing	2	Catching
A_{19}^2	5	Yes	No	Shooting	1	Waiting
A_{20}^2	5	Yes	Yes	Shooting	1	Waiting

Table. II. Attack tactic $\mathbf{A}_2$.

D_j^i	reg_ opp1	reg_ opp2	see_ robot1	action1	reg robot2	action2
D_1^1	1	1	1	Catching	1	Waiting
D_2^1	1	2	1	Catching	2	Waiting
D_3^1	1	3	1	Catching	2	Waiting
D_4^1	1	4	1	Catching	3	Guarding
D_5^1	1	5	1	Catching	4	Guarding
D_6^1	2	1	2	Catching	1	Waiting
D_7^1	2	2	2	Catching	2	Waiting
D_8^1	2	3	2	Catching	2	Guarding
D_9^1	2	4	2	Catching	3	Guarding
D_{10}^1	2	5	2	Catching	4	Guarding
D_{11}^1	3	1	3	Catching	2	Waiting
D_{12}^1	3	2	3	Catching	3	Waiting
D_{13}^1	3	3	3	Catching	3	Blocking