MCT Susano Logics 2019 Team Description

Shohei Degawa, Yuki Kurosaki, Naoki Kanyama, Yoshitaka Sota,
Toshiyuki Beppu

National Institute of Technology, Matsue College
14-4, Nishi-Ikuma, Matsue, Shimane, 690-8518, Japan

beppu[at]matsue-ct.jp
http://www.matsue-ct.ac.jp/

Abstract. We developed AI software, which organize the defense and the offense during a game. The AI consists of three levels of strategy, tactics and play. Strategy level decides the combination of multiple robot actions. A tactic is a series of single robot actions. Play is detailed robot commands. A ball possession finder estimates the team in possession of the ball. AI chooses an offense or a defense strategy by the estimation of ball possession, the ball and robots' positions, and the command from the auto referee.

Keywords: RoboCup, SSL.

1 Introduction

MCT Susano Logics was founded in 2011, and the team was named after a hero of Japanese mythology, *Susano*. *Susano* was a brother of *Amaterasu*, the goddess of the Sun. He exterminated a huge dragon *Yamatano-Orochi* which had an eight-forked head and an eight-forked tail. Our team was named with the hope of defeating strong and intelligent dragons in SSL. Our robots have a distinctive transparent shell (Fig. 1).

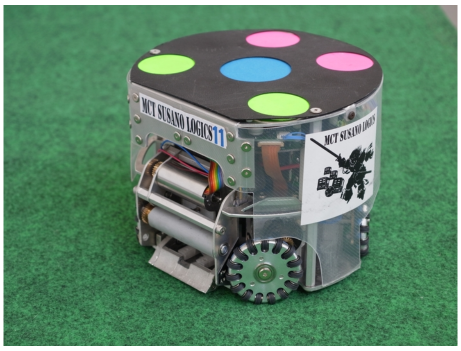


Fig. 1. MCT Susano Logics' robot

Until the year before last, we tried to adopt a rapidly exploring random tree (RRT) algorithm to determine the moving path of our robots. The RRT is a good method of finding a route to the desired location while avoiding opponent robots. We programmed our moving path finder to calculate a new travel path when an opponent robot was on the previous path. However, the positions of robots on the SSL field vary greatly every moment. Our RRT program indicated a different moving path for each calculation, thus our robot often could not arrive at the target location. Last year, we gave up embedding the RRT and implemented heuristic obstacle avoidance algorithms [1]. The avoidance program performed well, thus we can start developing new AI algorithms.

At RoboCup 2019 Sydney, MCT Susano Logics used the man-to-man system throughout a game. Team AI assigns the opponent's robot to be tracked by the team's robot at every re-start of the game. However, the calculation of the man-to-man position was fairly static, so we could not keep up with fast moving teams. In addition, with our man-to-man system, even if our team robot got the ball, the ball owning robot could not pass the ball to a friend robot since friends continued tracking the opponent robot. In Sydney, our robot sometimes owned the ball, however, the robot could shoot the ball, but could not continue organized attacks. Thus, MCT Susano Logics got no wins.

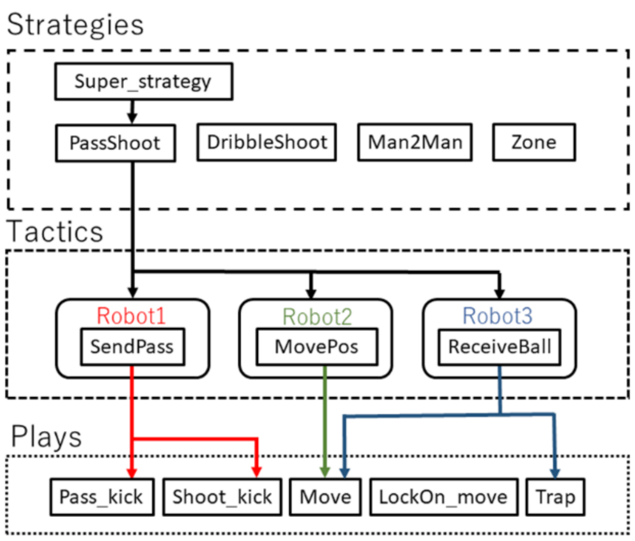
This year, we developed a new AI software, which organizes the defense and the offense during a game. The AI estimates which the team owns the ball and selects an offense or a defense strategy. This TDP describes our new software.

2 AI Software

2.1 AI Software Structure

Figure 2 shows the AI software structure of MCT Susano Logics. Our AI software consists of three levels of strategy, tactics, and play. The strategy level decides the combination of multiple robot actions. The top level strategy, *Super_strategy*, chooses a strategy by the estimation of the ball owning team, the ball and robots' positions, and the command from the auto referee.

Table 1 shows strategies in play. If a team robot owns the ball, *Super_strategy* chooses *PassShoot Strategy* or *DribbleShoot Strategy* for attacking. The table also shows the tactics assigned by the chosen strategy. A tactic is a series of single robot actions. For example, when *PassShoot Strategy* is chosen, *PassShoot Strategy* assigns *SendPass Tactic* to a robot of ball possession, *ReceiveBall Tactic* to robots in positions where the robot can shoot, and *MovePos Tactic* to other robots to get other shooting positions.



Ball Possession	Strategy	Actions in the Strategy	Assignable Tactics by the strategy
Friend's	PassShoot	Calculate positions for shooting, move robots to positions, pass the ball to a robot, and kick the ball toward goal.	MovePos
			ReceiveBall
			ReceiveBallShoot
			SendPass
	DribbleShoot	Calculate positions for shooting, move the robot of ball possession to the position, and kick the ball toward goal.	MovePos
			MovePosShoot
Opponent's or Ambiguous	Man2Man	Assign the opposing team's robot to be marked by the team's robot.	DrawBall
			LockOn
			MovePos
			PickBall
	Zone	Calculate robot positions from the ball and opponent robots' positions.	DrawBall
			MovePos
			PickBall

Table 2 shows the tactics in play. For example, *PickBall Tactic* is assigned to a robot closest to the free ball by *Man2Man Strategy* or *Zone Strategy*, the tactic calculates the coordinates of the location where the robot can encounter a rolling ball and chooses the plays to compose the robot action. For the prediction of the rolling ball speed, we used ER-Force's study as a reference [2].

Table. 2. List of tactics in play

Tactic	Actions in the Tactic	Plays assigned by the Tactic
DrawBall	Draw the opponent's ball	Move
LockOn	Track the designated opponent robot	LockOn_move
MovePos	Move to the specified position	Move
MovePosShoot	Dribble to specified position and shoot	Move
		Shoot_kick
PickBall	Catch a rolling ball	Move
		Trap
ReceiveBall	Move to the specified position and receive the ball passed	Move
		Trap
RecieveBallShoot	Move to the specified position and receive the passed ball and shoot	Move
		Trap
		Shoot_kick
SendPass	Pass (chip) the ball to another robot	Move
		Pass_kick

Table 3 shows the plays for a robot. A play consists of detailed robot commands. For example, *LockOn_move Play*, called by *LockOn Tactic* under *Man2Man Strategy* calculates the robot position from the coordinates of the designated opponent robot and moves the robot to the calculated position.

Another play, *Pass_kick Play* called by *SendPass Tactic* under *PassShoot Strategy*, calculates the speed of the kicked ball at the position of the receiving robot, rotates the kick robot in the direction of the specified receiving position, and sends a kick command to the kick robot. At the same moment, *PassShoot Strategy* assigns a robot *RecieveBall Tactic*, then *Move Play* called by the tactic moves the receiving robot to the specified position, and then *Trap Play* adjusts the robot's position to the locus of the kicked ball.

Table. 3. List of plays

Play	**Action of the Robot**
LockOn_move	Track the designated opponent robot
Move	Move to the specified position and rotate to the specified direction
Pass_kick	Kick the ball toward the specified position at the receivable ball speed
Shoot_kick	Kick the ball toward the goal at the limitted ball speed
Trap	Receive the ball passed

2.2 Estimation of the possession of the ball

Super_strategy needs to know the team that owns the ball in order to choose a strategy. We developed a ball possession finder to estimate the team in possession. The finder uses the polar coordinate system with its origin on the center of the team's goal.

The velocity of the ball at time *n* v_n is represented using the speed scalar v_{rn} and the polar angle of the moving direction $v_{\theta n}$,

$$v_n \equiv v_{rn} \angle v_{\theta n} \tag{1}$$

Because the fluctuation of speed value affects the estimation, the finder calculates the filtered ball speed at time *n* V_n with Eq. (2).

$$V_n = \alpha v_n + (1-\alpha) V_{n-1} \tag{2}$$

where α is the filtering coefficient. The value of α was experimentally set to 0.005.

If the average ball speed V_n is equal to or lower than the speed threshold V_{stop}, which was experimentally set to 0.3 meter per second, the finder calculates evaluation values of each robot's ball domination E_{robot} by the distance between the robot and the ball r_{ball} and the ball position angle θ_{ball} (Fig. 3).

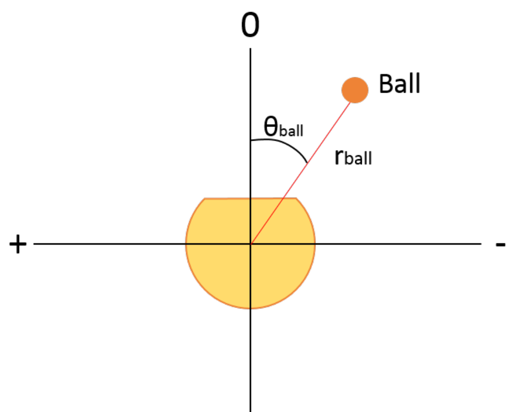


Fig. 3. Distance between the robot and the ball and the ball position angle

The evaluation value of the distance between the robot and the ball E_L is,

$$\begin{cases} E_L = 1 - \dfrac{r_{ball}}{L_{limit}} \left(r_{ball} < L_{limit} \right) \\ E_L = 0 \left(r_{ball} \geq L_{limit} \right) \end{cases} \tag{3}$$

where L_{limit} is the limit distance of the ball domination and was set to 500 millimeters. The evaluation value of the angle between the center line of the robot and the direction of the ball E_θ is,

$$\begin{cases} E_\theta = 1 - \dfrac{\left|\theta_{ball}\right|}{\theta_{robot}} \left(\left|\theta_{ball}\right| \leq \theta_{robot} \right) \\ E_\theta = 0 \left(\left|\theta_{ball}\right| > \theta_{robot} \right) \end{cases} \tag{4}$$

where θ_{robot} is the limit angle for controlling the ball with the robot's dribbler device, which was set to 90 degrees. The evaluation values of the robot's ball domination E_{robot} is,

$$E_{robot} = E_L \times E_\theta \tag{5}$$

The evaluation value of the team's ball domination E_{team} is set to the maximum ball domination value of team's robots.

$$E_{team} = max \left(E_{robot} \right) \tag{6}$$

where E_{team} is E_{friend} or $E_{opponent}$.

The ball possession finder estimates the ball dominating team by comparing the team's ball domination ratings.

$$\begin{cases} Friend's \left(E_{friend} > E_{opponent} + E_{th} \right) \\ Opponent's \left(E_{opponent} > E_{friend} + E_{th} \right) \\ Ambiguous \left(\left| E_{friend} - E_{opponent} \right| \leq E_{th} \right) \end{cases} \tag{7}$$

where E_{th} is the minimum difference and was experimentally set to 0.005. If the difference between the team's ratings of the ball domination is equal to or less than E_{th}, the estimation will be "ambiguous".

If the average ball speed at time n V_n is larger than the speed threshold V_{stop}, and the last ball speed V_{n-1} is equal to or lower than V_{stop},

$$\begin{cases} V_n > V_{stop} \\ V_{n-1} \leq V_{stop} \end{cases} \tag{8}$$

the finder considers that the ball speed is varied by a kick. At that moment, the finder retains the previous estimation until the ball goes out the field or the ball speed becomes lower than V_{stop}.

2.3 Man-to-man defense strategy

Until last year, our AI calculated the position of team's robot P_{friend} from the position of the designated opponent robot $P_{opponent}$ (Figure 4). The program used the polar coordinate system with its origin on the center of the team's goal. The position of the designated opponent robot $P_{opponent}$ is,

$$P_{opponent} \equiv r_{opponent} \angle \theta_{opponent} \tag{9}$$

The x component of $P_{opponent}$ is,

$$x_{opponent} = r_{opponent} \cos \theta_{opponent} \tag{10}$$

The program set the distance between the team robot and the opponent Δr using $x_{opponent}$,

$$\Delta r = k \times x_{opponent} + l \tag{11}$$

where *k* is the proportional constant, was set to 0.077, and *l* is the distance offset, was set to 530 millimeters experimentally. The calculated location of team robot P_{friend} will be,

$$P_{friend} = (r_{opponent} - \Delta r) \angle \theta_{opponent} \tag{12}$$

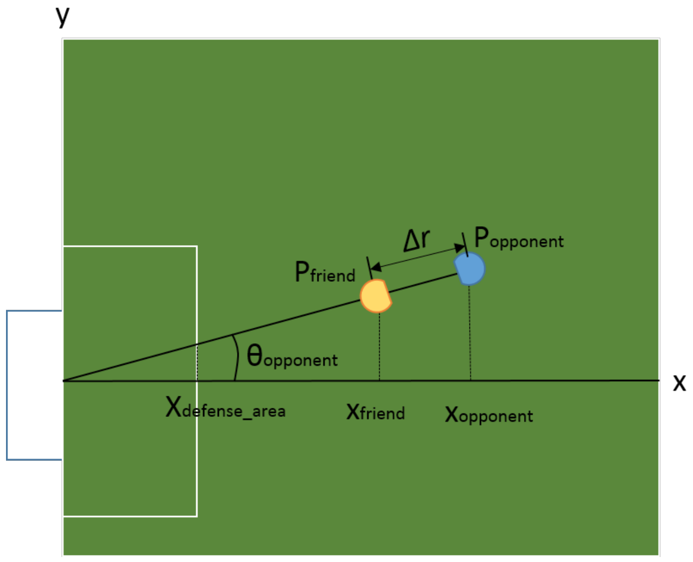


Fig. 4. The position of the team robot during man-to-man system

Our man-to-man system had two major defects. First, we could not attack by combining multiple robots. The robot closest to the ball was programmed to go and get the ball. However, even if the robot got the ball, the ball owning robot could not pass the ball to a friend robot since friends continued tracking opponent robots. So we developed the ball possession finder. At the moment a team robot gets the ball, Super_strategy will be able to change a defense strategy to an offensive strategy by the estimation of the ball possession finder.

Another defect of the man-to-man system was the slow response. The position of team robot was calculated from the position of opponent's, however, the opponent robot went somewhere before the team robot arrived. Therefore, we added the linear prediction using the velocity and the acceleration of the opponent robot.

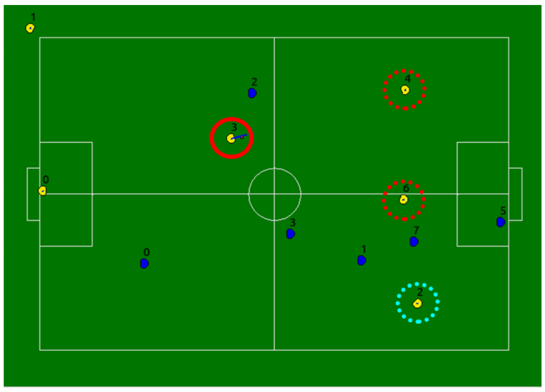
The team robot position is calculated not from the present opponent's position $P_{opponent}$, but from the predicted position after *t* seconds $P'_{opponent}$.

$$P'_{opponent} = P_{opponent} + V_{opponent} \cdot t + \frac{1}{2} a_{opponent} \cdot t^2 \qquad (13)$$

where $V_{opponent}$ is the present velocity of the robot and $a_{opponent}$ is the present acceleration of the robot.

3 Simulations

Figure 5 shows a simulation result of *PassShoot Strategy*. A robot (red circle) holding the ball was scanning pass receivers (Fig.5(a)). There were three receiver candidates, two of them (red dotted circle) were free, while another robot (blue dotted circle) had two opponent robots on the ball passing line. A short line on the robot in the red circle indicates the kick direction, which shows that the strategy chose the upper robot as the receiver. After the pass, the upper robot (red circle) held the ball and the short line indicates that the robot would shoot the ball toward the opponent goal (Fig.5(b)).



(a) Scanning the pass target

(b) After receiving the ball

Fig. 5. A simulation result of *PassShoot Strategy*

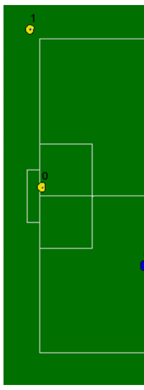
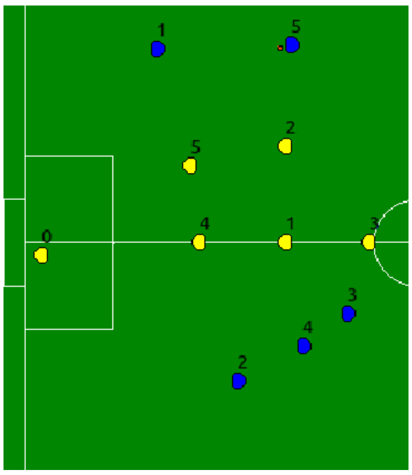
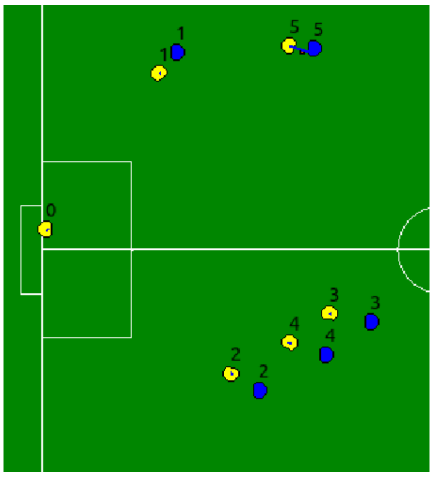


Figure 6 shows a simulation result of *Man2Man Strategy*. The Robots of the yellow team went next to the opponent robot waring the same number. The yellow robot waring number five went to the ball to draw it. Although we cannot indicate the result of moving target, the response of the opponent's tracking seems to be better with the linear prediction.



(a) Start



(b) Man-2-man formation

Fig. 6. A simulation result of *Man2Man Strategy*

4 Conclusion

Our new AI and revised man-to-man system are now under development and have been tested only through simulations. We will try them in Japan Open games before participating in RoboCup 2020 Bordeaux.

References

1. Koki Inoue, Naoki Kanyama, Shohei Degawa, Yuki Kurosaki, Toshiyuki Beppu. MCT Susano Logics 2019 Team Description, https://ssl.robocup.org/wp-content/uploads/2019/03/2019_TDP_MCT_Susano_Logics.pdf
2. Christian Lobmeier, Peter Blank, Jonas Buehlmeyer, Daniel Burk, Michael Eischer, Adrian Hauck, Markus Hoffmann, Stefan Kronberger, Markus Lieret, Bjoern M. Eskofier, ER-Force Extended Team Description Paper RoboCup 2016, https://ssl.robocup.org/wp-content/uploads/2019/01/2016_ETDP_ER-Force.pdf