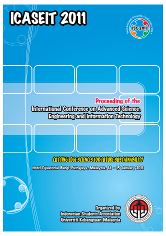
International Scientific Conference 2011
ISC 2011

Proceeding of the International Conference on Advanced Science, Engineering and Information Technology 2011

Hotel Equatorial Bangi-Putrajaya, Malaysia, 14 - 15 January 2011

ISBN 978-983-42366-4-9



Coordination Algorithm for Multi Robot Collaboration in Soccer Game

Awang Hendrianto Pratomo[1, 2], Anton Satria Prabuwono[1], Mohamad Shanudin Zakaria[1],
Siti Norul Huda Sheikh Abdullah [1], Khairuddin Omar[1], and Md Jan Noordin[1]

[1)]Centre for Artificial Intelligence Technology, Faculty of Information Science and Technology
Universiti Kebangsaan Malaysia, 43600 UKM Bangi, Selangor D.E., Malaysia.
Tel : +60 38921 6090

[2)] Department of Informatics Engineering, Faculty of Industrial Engineering, University Pembangunan Nasional
“Veteran” Yogyakarta, Jl. Babarsari No.2. Jogjakarta 55281, Indonesia
Tel : +62 274 485323

awang.upn@gmail.com, {msz, antonsatria, ko, mimi}@ftsm.ukm.my

***Abstract* - Robot Soccer is a rich domain for the study in artificial intelligence. Teams of players must work together in order to put the ball in the opposing goal. Learning is essential in this task since the dynamics of the system can change as the opponents' behaviours change. The players must be able to adapt to new situations. In this paper, we create a passing, avoiding obstacle and shooting strategy for robot soccer coordination. Based on a scenario in robot soccer, we stimulate a mini case study which involves two robots and ball.This research proposes coordination algorithm for robots collaboration in soccer game. The method is based on role, act, and behaviour of the robots. The actions of each robot depend on the created situation. The simulation result shows its probability to be applied in the real robot soccer game.**

***Keywords* - Robot soccer strategy, obstacle avoidance, shooting strategy, passing strategy.**

I. Introduction

Robot soccer is a game similar to football match with only one significant difference: the players aren't humans but robots. Normally, the robots are miniature wheeled mobile vehicles because of their simple mechanical construction and control [1]. Each team consists of five players; one of them is the goal keeper, two robots as defender and two robots as striker. Robots must be fully autonomous and controlled by the computer. The CCD camera mounted overhead the field and connected to the computer is used for gaining the information about the position of individual players and the ball. The game between two teams (each having its own camera and computer) is running without the human intervention, with the exception of foul or stalemate situation, when the game is interrupted and a free kick or penalty kick is called. The game is judged by the human referee.

Robot Soccer is a rich domain for the study in artificial intelligence. The robots players must work together in order to find the ball their own goals and create goals in the opponent goal [2]. Learning is essential in this task since the dynamics of the system can change as the opponent's behaviours change. The players must be able to adapt to new situations. Not only must the robots players learn to adapt to the behaviours of different opponents, but they must learn to work together. Soon after beginning to play, novice soccer players learn that they cannot do everything on their own: they must work together as a team to winning the game. However, before robots players can learn any collaborative or adversarial techniques, they must first acquire some low-level skills that allow them to manipulate the ball.

Basic skills for robot soccer strategy such as: dribbling and controlling the ball though entirely individual in nature are as critical to the development of a robot soccer player such as passing and receiving passes, which are more collaborative in nature. Several research studies have been done to develop robots soccer strategy that can play [RoboCup Research & Education] [2]. However, because of the rich and dynamic condition of the environment in robot soccer game, directly programming of the robots soccer strategy to play soccer is impractical, if not impossible. Therefore, the main focus has been shifted to treat the robots as an intelligent agent that can learn how to play soccer under with different environments and conditions [3].

The objective of our research which is to create algorithm for coordination about two robots to passing and moving with obstacle avoidance for shoot the ball into the goal. Besides that, we need to set and adjust the coordinate of the robot to make sure shooting properly. However, we also need to synchronize the timing between two robots, the purpose of the synchronize are make sure the robot shooting the ball into the goal area, at the same time, another robot moved with obstacle avoidance to the shoot the ball from the goal into goal.

II. Methodology

A mobile robot, which is equipped with a camera mounted overhead in the field, is sketched in the lower right area of the image. The data image reads from an image-processing module that must try to detect an object. The object is analyzed and reconstructed after a successful detection. In our research in Robot Soccer programming we used Visual C++ for created dll strategy files.

Robot Soccer game is a demanding application for the robots have a very fast and the entire thinking and decision making. The process must be finished after a few milliseconds. The intense time pressure is the one reason for the centralized robot team organization with a computer for processing the strategy [4].

The main problem is the definition of the behaviour of football match, included opponents model and strategy. MiroSot system usually uses hard coded sets of if-then rule that prevent easy updates or changes. Robot soccer is fully autonomous control structure, the planning and activities of the robot in the team controlled by the computer systems. [5].

The control architecture of robot soccer implements a process where simple task achieving behaviours are added as required. Each process is behaviour producing in its own strategy, although it may rely on the presence and operation of players [6].

The role level in control architecture is related to **who**. It refers to choose a role and area of manoeuvre in the playground for each robot such as: defender, midfielder, attacker, and goalkeeper. In the act level that related to **what** issue, that refer to selected an appreciate action for each team robot according to assigned the role. Such as: dribbling, passing, shooting, blocking, etc. behaviour level is refer to the issue of **how**. That is refers to condition checking for the robot condition such as the process of the robot for moved the robot with or without obstacle avoidance. The execution level is refer to motion of the robot for execute that condition [7]. The execution level refers to hardware control for controlling the motor for the robots movements.

III. Discussion

In our research the robot 2 (R_2) has to pass the ball to shooting area and then at the same time, the robot 1(R_1) has move with avoid opponent defender (R_D) and move to the R'$_1$ position and shoot the ball to goal in shooting area. In our data, 1 pixel represents 2.78 mm. Position and moving path of the R_1, the R_2, and the R_D are shown in Fig.1.

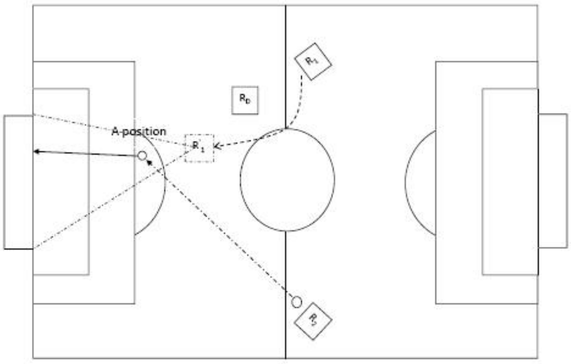


Fig 1: Position and moving path of the R_1, R_2, R_D and R'$_1$

Coding below is the source codes for the R_1 robot and the R_2 robot. Fig.2 is the algorithms that explain how these source codes works.

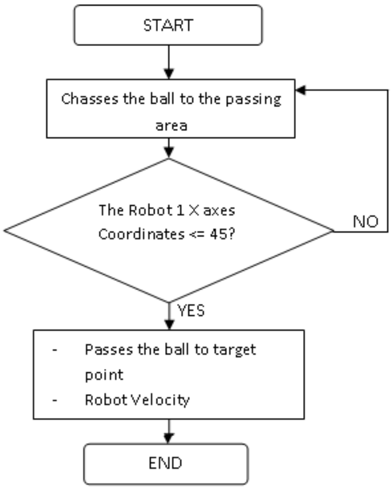

START
Chasses the ball to the passing area
The Robot 1 X axes Coordinates <= 45?
NO
YES
- Passes the ball to target point
- Robot Velocity
END

Fig 2: Algorithm passing of the R_2 [8]

Alorithm 1 : The proposed passing algorithm

Input : Current robot positions and the ball position.

Output : Robot velocity .

Step 1 : Assume each robot position , *R* is the robot, the ball as b , and

Step 2 : Estimate the ball to the target position .

```
        dx1 = x_t - x_b;                      (1)
        dy1 = y_t - y_b;                      (2)
```

Step 3 : Calculate distance, from the robot to the ball

```
        dx2 = x_b - robot->pos.x;             (3)
        dy2 = y_b - robot->pos.y;             (4)
```

Step 4: Estimate the rotation angle of the ball to the target, and rotation angle of the robot to the ball.

Calculate desired shooting angle

```
if (dx2 == 0 && dy2 == 0)
        phi2 = 90;
else
        phi2 = (int)(180. / PI *
        atan2((double)(dy2),(double)(dx2)));(5)
```

Step 5 : Calculate the angle error as the following.

```
theta_d = 2 * phi2 - phi1;            (6)
theta_e = theta_d - (int) robot->rotation;(7)
```

Step 6 : Set the kinematic value or propotional gain,

```
if (abs(theta_e) > 50)
       Ka =0.16;
else if (abs(theta_e) > 40)
       Ka =0.18;
else if (abs(theta_e) > 30)
       Ka =0.2;
else if (abs(theta_e) > 20)
       Ka =0.22;
else Ka =0.24;
```

Step 7 : **C**alculate robot velocity of the robot movement,

```
vl = (int) (Vc - Ka * (double) theta_e); (8)
vr = (int) (Vc + Ka * (double) theta_e); (9)
```

Step 8 : End.

Different algorithms for the R_1 and R_2 are built and run simultaneously. Algorithm for the R_2 is shown in Fig.2.

In this case, the R_2 will be placed at bottom of the field, as shown in Fig.1. The R_1 will be placed at the top of the field. An opponent defender will be placed in front of the R_1 to block its way as shown in Fig.3.

Fig.4 shows the actual position of the robot 3 and the desired position.

i. Some of the method needs to code or program it, to make sure the robot shoots the ball into the desired location.

ii. Set the current position of the ball and position of the robot.

Position of the robot in X-axes and Y-axes as shown in the following code:

```
double x1 = (&env->home [n])->pos.x; (10)
double y1 = (&env->home [n])->pos.y; (11)
```

The codes for passing the ball are shown in the following Algorithm:

Input : Current possition of all the robots and the ball, and the subsequent position of the robots.
Output : Robot velocity
Step 1 : Assume the distance of the robot to the obstacle, length of the free area from the robot position, to the obstacle position, the difference angle between the robots to the obstacle and the angle of the robots.

Distance = $\sqrt{(O_x - x)^2 + (y - O_y)^2}$ (12)
length = fabs($(O_x - x)$*sin(θ_d)+($y - O_y$)*cos(θ_d))(13)
angle = atan2($O_y - y$, $O_x - x$) (14)
diff_angle = θ_d $-$ angle (15)
where :
Distance = distance from obstacle to the robot
O_x, O_y = Obstacle x, y coordinate
X, y = Robot x, y coordinate
θ_d = Robot angle

Step 2 : Predict the subsequent action based on the condition of obstacle and target point, calculate the safety margin for collision avoidance and the radius of the obstacle.

Step 3 : If the action prediction is 'move and avoid', we calculate the distance from the centre gravity of the obstacle .

Step 4 : Calculate the radius of the robot and the relative position to the obstacle.

Step 5 : Estimate the desired position of the robot, the bearing or the orientation of the robot.

$T_x = B_x - Dh . \cos (\theta_t)$ (16)
$T_y = B_y - Dh . \sin (\theta_t)$ (17)

Where :
T_x, T_y is the shooting position.
B_x , B_y is the ball position.
Dh is the desired distance behind the ball.
θ_t is an angle from the ball towards the desired direction (in radians).

Step 6 : Calculate direction of the robot behind the ball for moving towards a shooting position to the goal.

$\sqrt{(B_X - R_X)^2 + (B_Y - R_Y)} > D_f$ (18)
or
$| \Phi2 - \Phi1 | < 90^o$
Where :
Bx , By is the ball position.
Rx, Ry is the robot position.
Df is the desired distance behind the ball.
Φ2 is an angle of ball to the target point (TP).
Φ1 is an angle of robot to the ball.

Step 7 : Use algorithm 1 to calculate the velocity of the robots.

Step 8 : End.

To avoid the obstacle, we used control statement to control the different condition. For example when the robot position reach the x-axes larger than 45 and if robot Y detect obstacle then it will avoid clash with the obstacle.

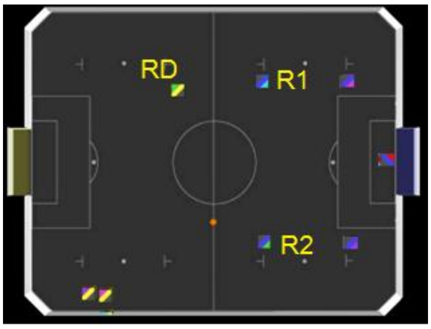


Fig 3: Position of the R_1, R_2 and R_D

When the robot moves to the shooting area which is less than or equal to the 45 at x-axes, the shooting function will be operate. The robot will shoot the ball into the goal. The control statement was used to control the robot for shoot the ball into the shooting area. When the robot reach area more than 20 in the y axes or the robot position at the x-axes between 50 and 60, then robot will stop which is velocity left and right set as zero.

The R_1 will keep detecting the distance between itself and the ball while it is moving. If the distance between the ball and R_1 is less than 45, R_1 will charge the ball into the goal as shown in Fig.5.

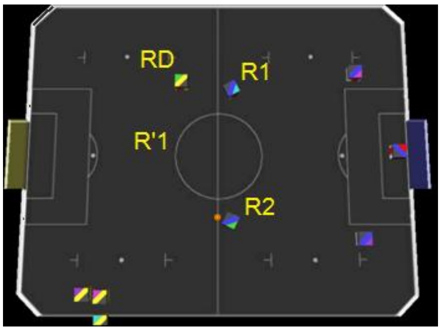


Fig 4: The R_1 avoid R_D by turning to left side while it is moving to R'_1

IV. Conclusions

From this research we propose some algorithm for robots coordination and collaboration in soccer game. We need to set and adjust the coordinate of the robot position and the obstacle to make sure the robot can move with obstacle avoidance, passes and shooting the ball properly. The synchronization for passing time and robots movements are made the robot shoots the ball at the shooting area, at the same time. Our strategy had been tested on robot soccer simulator. The success of the robot for shooting the ball was highly depending on the speed of the ball and the robot 1 movement.

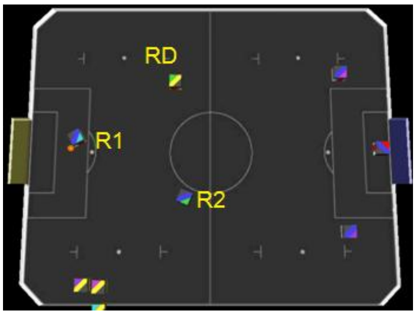


Fig 5: The R_1 successes to charge the ball into the goal

We aim to improve the stability and velocity of the robot's behaviour to kick the ball to the goal in advance. Hence, we correct the erroneous occurred in the previous algorithm and understand it as well. From the strategy algorithm that we have implemented for improvements, we can clearly distinguish the different type of functions we have experimented and the effect of it. In our research we need to set and adjust the ordinate of the robot to make sure shooting properly. And we also need to synchronize the timing between two robots, the purpose of the synchronize are make sure the robot passes the ball at the shooting area, at the same time, another robot move with obstacle avoidance and the shoot the ball from the shooting area into goal.

Acknowledgment

The researchers would like to thanks Anton Heryanto, Yon Hendri, Muhammad Nuruddin Sudin, Muhamad Syafiq Shohaimi, and Dwi Yanuar Panji Tresna for their contribution in this research, University Kebangsaan Malaysia for providing facilities and financial support under Operational University Research Grant of Pattern Recognition Research Group No. UKM-OUP-ICT-26-186/2010, Arus Perdana No. UKM-AP-ICT-17-2009 and Fundamental Research Grant Scheme No. FRGS UKM-TT-03-FRGS0131-2010.

References

[1] Minoru Asada, Hiroaki Kitano, "Robotics and Autonomous System", The RoboCup Challenge [A], pp.3-12, 1999[C].

[2] Kim J H, Shim H S, A cooperative multi-agent system and real time application to robot soccer [A]. Proc of IEEE IntConf on Robotics and Automation[C]. Albuquerque, pp. 638-643, 1997.

[3] Kim, H.R., Hwang, J.H., Kwon, D.S.: Co-operative Strategy for an Interactive Robot Soccer System by Reinforcement. Int. J. Control, Automation, and Systems, pp. 236--242 (2003)

[4] Uwe Egly, Gregor Novak, and Daniel Weber, Decision Making For Mirosot Soccer Playing Robots, 1st CLAWAR/EURON/IARP Workshop on Robots in Entertainment, Leisure and Hobby, 2005, pp: 69 – 72.

[5] R. Alami, F. Ingrand and S. Qutub. A scheme for coordinating multi-robot planning activities and plans execution. In 13th European Conference on Artificial Intelligence, pages 617–621, 1998.

[6] HAN Xue-dong, HONG Bing-rong, MENG Wei, "Distributed Control for Generating Arbitrary Formation of Multiple Robots", ROBOT, Vol 25, No. 4, pp. 66-72, Jan. 2003.

[7] Huang, L.: Velocity Planning for a Mobile Robot to Track a Moving Target a Potential Field Approach. J. Robotics and Autonomous Systems 57, pp. 55--63 (2009)

[8] Pratomo, A.H., Prabuwono, A.S., Zakaria, M.S., Omar, K., Nordin, M.J., Sahran, S., Abdullah, S.N.H.S., Heryanto, A.: Position and Obstacle Avoidance Algorithm in Robot Soccer. Int. J. Computer Science 6 (2), pp. 173--179 (2010)