Jurnal Ilmiah KURSOR
Menuju Solusi Teknologi Informasi

Vol. 11, No. 1, Juli 2021

ISSN 0216 – 0544
e-ISSN 2301– 6914

ADVANCE ATTACK AND DEFENSE STRATEGY ALGORITHM WITH DYNAMIC ROLE ASSIGNMENT FOR WHEELED SOCCER ROBOT

[a]Rudy Dikairono, [b]Setiawardhana, [c]Fajar Budiman, [d]Djoko Purwanto, [e]Tri Arief Sardjono

[a,b,c,d]Department of Electrical Engineering, Institut Teknologi Sepuluh Nopember (ITS), Surabaya, Gedung A, B, C dan AJ, Kampus ITS, Sukolilo, Surabaya Indonesia 60111
[e]Department of Biomedical Engineering, Institut Teknologi Sepuluh Nopember, Sukolilo Surabaya, Gedung A, B, C dan AJ, Kampus ITS, Sukolilo, Surabaya Indonesia 60111
E-mail: rudydikairono@ee.its.ac.id, setiawardhana16@mhs.ee.its.ac.id, fajarbudiman@ee.its.ac.id, djoko@ee.its.ac.id, sardjono@bme.its.ac.id

Abstract

Game strategy is one of the most critical parts of winning a soccer robot match and cannot be separated from the cooperation among robots in making movements to score goals. In this paper, a wheeled soccer robot game strategy called advance attack and defense has been developed. The strategy is combined with dynamic role assignment, in which robot can change from an attacker to a defender and vice versa. Defender robots are not only based on defensive area but will always block opposing attacker to score goal. The attack strategy performs a rotational trajectory for attacker robot to overpass opponent robot. This strategy has been proven to increase defense and attack effectiveness. Test results using soccer robot gameplay environment simulator developed by Institut Teknologi Sepuluh Nopember Robot with Intelligent System (IRIS) team show that the advance strategies are superior compared with basic strategies. In 30 matches, the advance dynamic strategy won 80%, drew 6.7%, and obtained the highest goal difference, 85 goals. The test was then verified with the implementation in the IRIS robots and showed the same performance. The developed game algorithms were tested in 2019 Indonesian wheeled soccer robot contest (KRSBI-B) and the IRIS team won the title.

Key words: Advance attack strategy, Advance defense strategy, Dynamic role assignment, IRIS Robot, KRSBI-B, Wheeled soccer robot, Wheeled soccer robot simulator.

INTRODUCTION

The development of soccer robots has involved many researchers around the world. An annual soccer competition has been held since 1997 until now, the beginning and the vision of the competition can be seen in [1]. There are two main categories in RoboCup for soccer, the humanoid soccer robot category, and the wheeled soccer robot category. Some works on humanoid soccer robot developments have been carried out in [2]–[4]. The developments focus on developing the robot motions and internal movements such as balance and robot dynamics, starting from kid size, teen size, and adult size. Movement optimization has been proposed in [5] to get the best moves in humanoid robot soccer games.

Compared with a humanoid soccer robot, a wheeled soccer robot uses wheels as its main drive so it is more stable. The development includes robot mechanics and the algorithm of the strategy of the soccer game. There are two subcategories in wheeled soccer robots, the first is Small Size League (SSL) and the second is Middle Size League (MSL). SSL uses a small robot and monitors the robot's positions using a global camera. This sub-category focuses more on developing game algorithms. MSL has a larger robot size and uses onboard sensors. This sub-category focuses on processing data from sensors and turning it into perceptions that robots can understand. This perception is then used to make decisions in implementing strategies in robot soccer games. Self-localization perception using Convolution Neural Network has been developed using Omni vision image and gyro-compass orientation as the input of the system [6]. A cooperative game for the roles assignment has been also developed by combining static and dynamic roles assignment [7]. This cooperative role assignment game provides optimization for the deployment of robots in the field but still requires additional strategies to win matches. Related research on artificial intelligence has been also carried out for ball position mapping [8] and intelligent navigation [9]. The robot can map and navigate the ball in the goal and this technique has been implemented on the IRIS goalkeeper robot.

Dynamic role assignment with strategy selection using reinforced learning also demonstrates a good result in SSL robot soccer for balanced attack and defense [10]. In MSL, strategy with dynamic role assignment on the soccer robot has also been demonstrated using a finite state machine of attack, defend, and intercept [11]. The strategy commonly used for centrally controlled in SSL robot soccer such as Skills, Tactics, and Plays (STP) has been integrated into MSL robot soccer for distributed control of the robot, performing strategy, and dynamic role assignment [12]. This is also proven to increase the number of goals significantly. Dynamic role assignment and non-hierarchical cooperation and hierarchical cooperation in MSL have been also implemented in [13]. This study develops advanced attack and defense algorithms for MSL soccer robot with dynamic role assignment and robot capability consideration. This means that each robot has been allotted the same capabilities to perform either as an attacker or defender. With these two algorithms, the team can increase the number of goals against the opponent and reduce the opponent's goals.

The match strategy implementation for the MSL soccer robot can be performed using a simulator. However, verification still has to be carried on with a robot tested in a real match. A simulator for the vision system and match strategy has been developed for the MSL soccer robot [14]. A simulator based on ROS and Gazebo has also been developed for RoboCup MSL [15]. Robot movement simulator has been developed by [16], the simulator is developed based on real robots developed previously. Overall, a specific simulator is required to meet the needs of a particular MSL soccer robot. The simulator also has to be developed specifically according to the real robots used in the competition.

This paper focuses on developing a strategy for wheeled soccer robot game using the man to man marking strategy as it has been applied in the human soccer game strategy. This strategy had been tried out in some matches with other strategies. The trials were carried out using a simulator developed by the IRIS team and then verified through matches with other robots in the Indonesian wheeled soccer robot contest. The highlighted contributions of this paper are as follows:

- Proposing a new attack strategy by passing the opposing defender in a circular motion. This strategy can

increase the effectiveness of attacks in wheeled soccer robot matches.

- Proposing a new defense strategy based on the defender's robot movement in blocking the direction of the opponent's attack by estimating the direction of the opponent's robot to kick the ball into the goal. This strategy is proven to reduce the number of opponents' goals.
- Proposing a new method for dynamic role assignment in wheeled soccer robot games. The use of this method is able to increase the number of goals in wheeled soccer robot matches.
- Proposing a soccer robot gameplay simulator environment based on IRIS robot for MSL RoboCup. This simulator can be used for the initial testing of a new strategy in the wheeled soccer robot game. The use of this simulator is proven to be able to speed up the development time of the strategy for the wheeled soccer robot game.

The next sections of this paper are organized as follows: Section 2 describes the development of a soccer robot gameplay simulator. Section 3 discusses the proposed game strategy in the form of advance man to man marking which the testing results are presented in section 4. Section 5 presents the conclusion and summary of the paper.

WHEELED SOCCER ROBOT ENVIRONMENT SIMULATOR

A soccer robot gameplay simulator environment had been developed to test the match strategy which would later be implemented into a real wheeled soccer robot competition. The simulator was created by the IRIS team based on the mechanical system and the robot software platform of IRIS[17]. The wheeled soccer robot matches were performed using the rules issued by RoboCup, which the latest rules can be seen in [18]. Furthermore, this paper used a rulebook issued by the Indonesian Wheeled Football Robot Contest as seen in [19], which was mostly based on the RoboCup rule.

In the simulator system, the objects of the robot, ball, and 3D field were prepared through 3D-based modeling software. The size of each object was made according to the rules in the Indonesian National Robot Contest competition. The simulation used the Open Dynamic Engine to simulate collisions between robots and ball dynamics. The simulator block diagram is depicted in Figure 1. The main part of the simulator is the Game Rule Engine which functions as a controller of all objects in the simulator. The other parts of this simulator are the Game Play Algorithm of Team A and Team B, Referee Box, Ball, Soccer Field, and 3D Visualization. The explanation of each block will be presented in the next section.

Game Rule Engine

Game Rule Engine is the most important part of this soccer robot gameplay simulator. The gameplay engine plays a role in calculating the physical behavior of every object in the simulator. The game rule engine takes the output data from each object and translates it into the physical movement of objects in the simulator.

The gameplay engine plays a role in controlling all the interactions of objects in the simulator. This gameplay engine gives an effect so that the simulator can run closer to real conditions. Parameters in the gameplay engine are hardcoded, such as robot weight, robot size, ball weight, ball size, wheel friction force, and other laws of physics that occur in the simulator. With this engine, the simulator can simulate friction, collision, as a real condition of the robots, ball, and field being used.

The input and output diagram of the rule engine game system can be seen in Figure 2. There are 6 incoming data lines and 6 outgoing data lines. The $\mathbf{R_{An}G}$ incoming data path pairs with the $\mathbf{GR_{An}}$ outgoing data path. This data path is connected to the *n* robot object of Team A. $\mathbf{R_{Bn}G}$ and $\mathbf{GR_{Bn}}$ are one pair of input and output data from the *n* robot of team B. The next pair of data is **ReG** and **GRe** which connect the game rule engine with the Referee Box. The ball object interacts with the game rule engine through the **BG** and **GB** data pair. **SG** and **GS** are data pair from the Soccer Field object. **SG** contains the size of the field and the features in the field. The next data pair is **VG** and **GV**, which is the data pair leading to the 3D Visualization object. The game rule engine calculates all visual positions of objects in the simulator. The results of this object calculation are then sent to the 3D visualization of the object to be visualized in the simulator view.

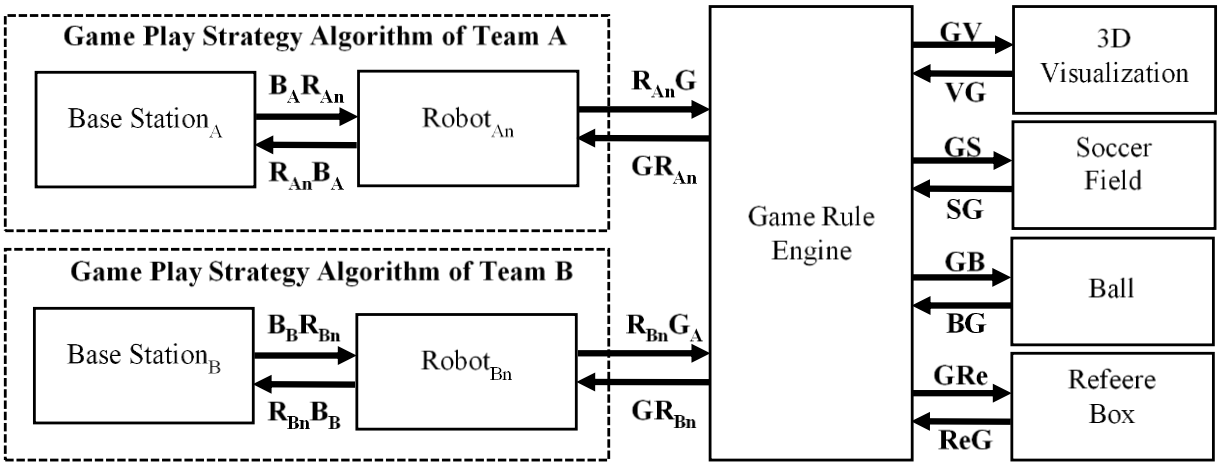

Fig 1. Functional diagram of Soccer Robot Game Play Simulator Environment; $\mathbf{B_AR_{An}}$ is the data from Base Station$_A$ to Robot$_{An}$: $_A$ is the the team name, $_n$ is the robot number; $\mathbf{R_{An}B_A}$ is the data from Robot$_{An}$ to Base Station$_A$; $\mathbf{R_{An}G}$ is the data from Robot$_{An}$ to Game Rule Engine; $\mathbf{GR_{An}}$ is the data from Game Rule Engine to Robot$_{An}$; $\mathbf{B_BR_{Bn}}$ is the data from Base Station$_B$ to Robot$_{Bn}$: $_B$ is the the team name; $\mathbf{R_{Bn}B_B}$ is the data from Robot$_{Bn}$ to Base Station$_B$; $\mathbf{R_{Bn}G}$ is the data from Robot$_{Bn}$ to Game Rule Engine; $\mathbf{GR_{Bn}}$ is the data from Game Rule Engine to Robot$_{Bn}$; **GV** is data from Game Rule Engine to 3D Visualization object; **VG** is the data from 3D Visualization object to Game Rule Engine; **GS** is data from Game Rule Engine to Soccer Field object; **SG** is the data from Soccer Field object to Game Rule Engine; **GB** is data from Game Rule Engine to Ball object; **BG** is the data from Ball object to Game Rule Engine; GRe is data from Game Rule Engine to Referee Box object; **ReG** is the data from Referee Box object to Game Rule Engine.

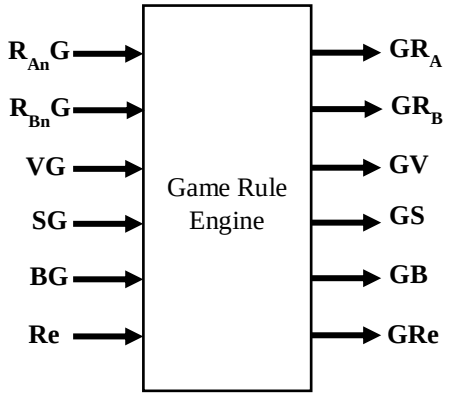

Fig 2. Game rule engine input and output

Robot Object

The robot object is a model made from the physical dimensions and functions of the IRIS team wheeled soccer robot. These physical dimensions include the length, width, height, and weight of the robots. The functions of the robot object are the robot's ability to catch, dribble, and kick the ball. Furthermore, for one match in the simulator, there are at least 2 player robot objects and 4 player robot objects in maximum.

The robot objects for team A are connected to the game rule engine via $\mathbf{R_{An}G}$ and $\mathbf{GR_{An}}$ data pair. The $\mathbf{R_{An}G}$ contains $[\dot{x}_{An} \quad \dot{y}_{An} \quad \dot{\theta}_{An}]^T$ that is a command of the robot's movement, where $\dot{x}_{An}$ is the velocity towards the x-axis, $\dot{y}_{An}$is the velocity towards the y-axis, and $\dot{\theta}_{An}$ is the velocity of rotation of the robot towards the robot's center. The $\mathbf{GR_{An}}$ is the output of the game rule engine which is calculated using the following formula:

$$\mathbf{GR_{An}} = [\dot{x}_{An} \quad \dot{y}_{An} \quad \dot{\theta}_{An}]^T \Delta t$$
$$= [\Delta x_{An} \quad \Delta y_{An} \quad \Delta \theta_{An}]^T \quad (1)$$

Δt is the time frame for one iteration in the simulator system. Δx_{An} and Δy_{An} are the distance movement of the robot in x and y Cartesian coordinate on the field in the simulator. $\Delta \theta_{An}$ is the rotational angle movement of the robot on the field in the

simulator. The robot object for team B is connected to the game rule engine via $\mathbf{R_{Bn}G}$ and $\mathbf{GR_{Bn}}$. The calculation of this data pair is the same as for $\mathbf{R_{An}G}$ and $\mathbf{GR_{An}}$.

Base Station Object

The base station object plays a role in the coordination system for the robots from each team. Team A's base station object is connected to each robot of team A via $\mathbf{B_AR_{An}}$ and $\mathbf{R_{An}B_A}$ data pairs. The base station object for Team B is connected to each robot of Team B through the $\mathbf{B_BR_{Bn}}$ and $\mathbf{R_{Bn}B_B}$ data pairs. Each robot sends its position, detected friendly robot position, detected opponent robot position, detected ball position, ball status, and role assignment status. Meanwhile, the base station object sends data to each robot in the form of data on all friendly robots' position, the position of all opposing robots, ball positions, ball status, role assignment of each friendly robot, and game strategies.

All data that goes into the base station is used to determine the strategy that will be used by the team. The role assignment for each robot is also determined by the base station based on each robot's position data in a team. The base station decision is then sent to each robot to be executed as a strategy played in the match.

Referee Box Object

The timer and orders in the match are carried out by the referee box. Referee boxes used in RoboCup matches can be seen in [18] . In real matches according to the rules of the Indonesian wheeled soccer robot contest [18], the referee box is controlled by the referees determined by the committee. There are 3 referees in one match, the main referee, the line referee, and the referee box operator referee. The time used in one match is 2 x 5 minutes, and there is an interval of 5 minutes. In the second half, each team changes positions on the field.

In this paper, the referee box is connected to the rule engine game through the **ReG** and **GRe** data pairs. **ReG** contains commands issued by the referee box consistin of: command data, status data, and match data. Meanwhile, **GRe** contains the status of the ball in the field which consists of: kick off ball, throw in ball, corner kick ball, goal kick ball, and goal ball.

Ball Object

The ball object is equipment in the wheeled soccer robot match which is a major part of the wheeled soccer robot's task to make goals. The ball is contested by both teams, and the team that manages to enter the ball into the opponent's goal will score a goal. The ball object is connected to the game rule engine through the **BG** and **GB** data pairs. **BG** contains $[\dot{x}_{ball} \quad \dot{y}_{ball} \quad \dot{z}_{ball}]^T$ where $\dot{x}_{ball}$ is the velocity of the ball in the direction of the x-axis, $\dot{y}_{ball}$ is the velocity of the ball in the direction of the y-axis, and $\dot{z}_{ball}$ is the velocity of the ball along the vertical z-axis. The game rule engine calculates the new ball position and sends it back to the ball object via **GB**. Calculation of the **GB** can be seen in the following formula:

$$\mathbf{GB} = [\dot{x}_{ball} \quad \dot{y}_{ball} \quad \dot{z}_{ball}]^T \Delta t$$
$$= [\Delta x_{ball} \quad \Delta y_{ball} \quad \Delta z_{ball}]^T \quad (2)$$

The Δx_{ball} is the distance movement of the ball object on the x-axis, the Δy_{ball} is the movement distance of the ball object on the y-axis, and the Δz_{ball} is the movement distance of the ball object on the z-axis. The ball object on the soccer field coordinate system is shown in Figure 3.

Fig 3. Soccer robot game play simulator environment: $\mathbf{R}_{A0}$ is goalkeeper robot of team A; $\mathbf{R}_{A1}$ is robot 1 of team A; $\mathbf{R}_{A2}$ is robot 2 of team A; $\mathbf{R}_{B0}$ is goalkeeper robot of team B; $\mathbf{R}_{B1}$ is robot 1 of team B; $\mathbf{R}_{B2}$ is robot 2 of team B; Ball is the ball object; x_f, y_f and z_f are coordinates axis of soccer field object.

The initial velocity of the ball comes from the touch of the ball with the robot or the robot's kick to the ball. This initial velocity

will move the ball and will experience a slowdown and or change of direction when touching field objects or robot objects. The calculation of the physical phenomenon of this ball is carried out by the rule engine game using the Open Dynamics Engine (ODE) library that has been created by Russell Smith [20].

Soccer Field Object

The soccer field object provides dimensional information of all the features in the field. These features include field lines, midfield points, penalty points, corner kick points, and goals. The size of the playing field is 8 m x 12 m. The goal is 2 m x 1 m in white paint. The field floor uses green carpet and the field white lines. On the edge of the field, there is a 1 meter wide empty area which is used for robot maneuvers in the event of a throw-in. The soccer field object is connected to the game rule engine via the SG and GS data lines. SG contains field feature data as mentioned above. Meanwhile, GS contains changes in field features such as damage to the goal and the fence if it is hit by a robot.

3D Visualization object

3D Visualization object serves to visually display all objects in the soccer robot gameplay simulator. Objects that are displayed visually include robots, balls, and soccer fields. The 3D visualization object was built using the Gazebo platform in the Robot Operating System (ROS) [21]. The visualization of the gameplay simulator is shown in Figure 3.

PROPOSED SOCCER ROBOT MATCH STRATEGY ALGORITHM

The competition rules in the proposed algorithm are based on the Indonesian Wheeled Soccer Robot Contest rule. The developed algorithm is limited to the number of robots that can be played, which is 3 robots including the goalkeeper. Although the number of robots is limited, it is possible that the number of players can be increased. This is important because the number of robots that can be played will increase according to the rules of RoboCup. The algorithm developed by the IRIS team has been tested in the 2019 Indonesian Wheeled Soccer Robot Contest and has won in regional and national competitions. The algorithm also received the best algorithm award in both events. The next section explains the IRIS robot, IRIS base station, and the advanced IRIS match strategy algorithm.

IRIS Robot

The soccer robot used in this paper was the IRIS robot. A 3D model of the IRIS robot was developed to be used in the simulator. The physical form of the IRIS robot and its 3D model is depicted in Figure 4. The IRIS robot weighs 37 kg and dimensions of 45 cm long, 45 cm wide, and 70 cm high. The robot uses 4 Omni wheels so that it can move in any direction.

(a) (b)

Fig 4. IRIS Robot: (a) Physical; (b) 3D model

The motion speed of the robot in the simulator is adjusted to the speed of the real robot in the field. This adjustment uses the direct measurement method. This adjustment is required to obtain the most similar results between the simulator and the real conditions. The maximum speed of the robot is 5 m/s with an acceleration of 2 m/s^2.

The calculation of the angular speed of each wheel to the linear speed of each wheel is according to the following formula:

$$\begin{bmatrix} \dot{\phi}_1 \\ \dot{\phi}_2 \\ \dot{\phi}_3 \\ \dot{\phi}_4 \end{bmatrix} = \begin{bmatrix} v_1/r_1 \\ v_2/r_2 \\ v_3/r_3 \\ v_4/r_4 \end{bmatrix} \quad (3)$$

where $\dot{\phi}_1, \dot{\phi}_2, \dot{\phi}_3$, and $\dot{\phi}_4$ are the angular speed of wheels 1,2,3 and 4 in rad/s; v_1, v_2, v_3, and v_4 are the linear speed of wheels 1, 2, 3, and 4 in meter/second. r_1, r_2, r_3,and r_4 are the radius of wheels 1, 2, 3,

and 4 in meters. The linear speed of each wheel is calculated in the following formula:

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} = \begin{bmatrix} -\sin(\alpha_1) & \cos(\alpha_1) & l \\ -\sin(\alpha_2) & \cos(\alpha_2) & l \\ -\sin(\alpha_3) & \cos(\alpha_3) & l \\ -\sin(\alpha_4) & \cos(\alpha_4) & l \end{bmatrix} \begin{bmatrix} \dot{x}_{An} \\ \dot{y}_{An} \\ \dot{\theta}_{An} \end{bmatrix} \quad (4)$$

where v_1, v_2, v_3, and v_4 are the linear speed of wheels 1, 2, 3, and 4 in m/s; α_1, α_2, α_3, and α_4 are the angle between x_r to the axis of wheels 1, 2, 3, and 4 in rad; l is the length of the wheels to the center of the robot.

The values of $\dot{x}_{An}$, $\dot{y}_{An}$, and $\dot{\theta}_{An}$ are generated by the robot navigation system. In a real robot the values of $\dot{x}_{An}$, $\dot{y}_{An}$, and $\dot{\theta}_{An}$ will be sent to the Omni- directional wheel driver system which then drives the Omni-wheels to move the robot on the soccer field. In the simulator the values of $\dot{x}_{An}$, $\dot{y}_{An}$, and $\dot{\theta}_{An}$ are sent to the Game Rule Engine which then moves the robot object according to the formula (1) which represents the movement of the robot in the simulator environment.

The robot uses a camera to find out its position on the field called self-localization, the position of the ball, and the position of other robots. Self-localization was obtained using CNN as seen in [6]. The CNN system provides an approximate position of the robot which is then combined with the robot's odometry system. The position of the ball is obtained by regressing the distance of the robot from the camera frame to the ball with the distance of the robot on the field to the ball. The regression results can be either exponential or polynomial equations, the selection of this equation considers the minimum error value determined by trial. The position of the other robots is obtained in the same way as for getting the ball position.

In the simulator, camera data is manipulated by providing direct output in the form of the robot's positions in the field, the ball's position, and other robots' positions. To perform this manipulation, each robot object is given a 360-degree LIDAR-like sensor. Therefore, the resulting output is similar to the actual condition. The robot is assumed to be in ideal conditions, where the mechanical slip that occurs can be solved using automatic control on each mechanical wheel and has been done and implemented in real robot [16].

IRIS Base Station

The base station is connected to the robots in a team using a Wi-Fi network. The base station is also connected to the referee box to get status and game commands in wheeled soccer matches. The robots are connected to the Base Station via $\mathbf{R_{An}B_A}$ and $\mathbf{B_AR_{An}}$ containing the following data:

$$\mathbf{R}_{An}\mathbf{B}_A = \{\mathbf{P}_{An}, \mathbf{P}_{Bn}, \mathbf{P}_{ball}\}$$

$$\mathbf{B}_A\mathbf{R}_{An} = \{\mathbf{P}_{\boldsymbol{An}}, \mathbf{P}_{\boldsymbol{Bn}}, \mathbf{P}_{ball}, \mathbf{C}, \mathbf{S}\}$$

$$\mathbf{P}_{An} = \{[x_{A1}, y_{A1}, \theta_{A1}], [x_{A2}, y_{A2}, \theta_{A2}], \dots\}$$

$$\mathbf{P}_{Bn} = \{[x_{B1}, y_{B1}, \theta_{B1}], [x_{B2}, y_{B2}, \theta_{B2}], \dots\}$$

$$\mathbf{P}_{ball} = \{[x_{ball}, y_{ball}, z_{ball}]\}$$

$$\mathbf{C} = \begin{Bmatrix} kickoff, goalkick, \\ freekick, cornerkick, \dots \end{Bmatrix}$$

$$\mathbf{S} = \begin{Bmatrix} Strategy_1, Strategy_2, \\ Strategy_3 \dots \end{Bmatrix} \quad (5)$$

where $\mathbf{P}_{An}$ is the position of the n-th robot from Team A, $\mathbf{P}_{Bn}$ is the position of the n-th robot from Team B. $[x_{A1}, y_{A1}, \theta_{A1}]$ is the coordinate of robot 1 from team A in the field, $[x_{A2}, y_{A2}, \theta_{A2}]$ is the coordinate of robot 2 from team A in the field. $[x_{B1}, y_{B1}, \theta_{B1}]$ is the coordinate of robot 1 from team B in the field, $[x_{B2}, y_{B2}, \theta_{B2}]$ is the coordinate of robot 2 from team B in the field. $\mathbf{P}_{ball}$ is the position of the ball on the field with the coordinate $[x_{ball}, y_{ball}, z_{ball}]$.

C is the command given by the Base Station to the robots for the match status. This command is obtained from the referee box. S is the strategy currently being played by the Team. The base station provides strategic decisions based on match conditions. $Strategy_1$ is a strategy for normal play when the score is equal at the beginning of the match, $Strategy_2$ is an offensive strategy when the score is losing, and $Strategy_3$ is a defensive strategy when the score is winning.

IRIS Role Assignment

The game strategy developed by the IRIS divides the roles of the robot into 2, the goalkeeper robot and the outfield robot. Outfield robots are divided into defender robots and attacker robots. This paper discusses the role of outfield robots. The defender robot has an area in front of the goal up to 0.5 meters in front of the centerline of

the soccer field. The attacker robots have an area 0.5 meters behind the centerline of the soccer field to the opponent's goal. There is an intersection area as wide as 1 meter in the middle of the field. This area is needed to provide wider coverage for the defender robots and the attacker robots to do their job. Changes in the role of robots have also occurred in this area. The division of the defensive and attack area is shown in Figure 5. The red rectangle is the defensive area and the yellow rectangle is the attack area. If the number of outfield robots is n, then the division of the number of defender and attacker robots follows the following formula:

$$n_d = \lceil {}^{n}/_{2} \rceil$$

$$n_a = n - n_d \quad (6)$$

where n_d is the number of defender players, n_a is the number of attackers and n is the number of outfield players. For more than one number of n_d, the defensive area will be divided by the number of n_d with a line in the direction of the y axis of the field. Likewise, if the number of n_a is more than one, the attack area will be divided by the number of n_a.

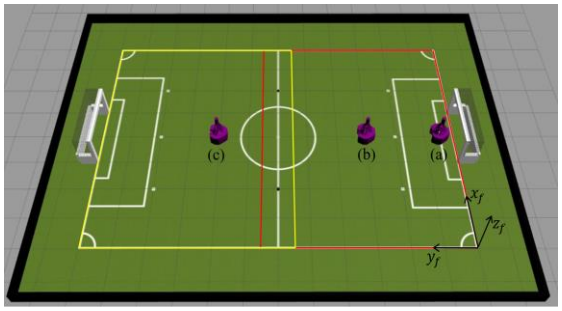

Fig 5. Soccer field area for robot: (a) goal keeper robot; (b) defender robot on defensive area; (c) attacker robot on attack area.

In the IRIS robot game algorithm, the role assignment of each robot can be made static or dynamic. In the static role assignment, a robot will have a fixed assignment from the beginning to the end of the game. In the dynamic role assignment, a defender robot can turn into an attacker robot and vice versa. The dynamic role assignment used in the IRIS robot is determined from the robot's position and ball control conditions of the robot. The defender robot that manages to get the ball in the attack area will turn into an attacker robot. Then, the position will be replaced by the attacker robot closest to the robot. Meanwhile, the attacker robot that gets the ball in the defensive area will turn into a defender robot and its position will be replaced by the defender robot closest to the robot.

The change in the role assignment of the IRIS team robot can be seen in Figure 6. There are 4 state role assignments for the robot, i.e. State 1 is the Attacking Attacker, which means the robot acts as an attacker and the team is in possession of the ball; State 2 is Attacking Defender which means the robot acts as the defender and the team controls the ball; State 3 is the Defending Defender which defends the ball, and this means the robot acts as a defender and the team is losing the ball; State 4 is Defending Attacker, which means the robot is acting as an attacker and the team is losing the ball. There are 4 inputs that affect the state change, i.e. a – the input to change the role of defender robot to attacker robot; b – the input for telling that the team gets the ball and the robot will change its mode to attacking; $\overline{b}$ – the input to indicate that the team loses the ball and the robot will change its mode from attacking to defending; and d – the input to change the role of the attacker robot to become defender robot. The four inputs are sent from the base-station to the robots based on conditions determined by the strategy algorithm in the base-station.

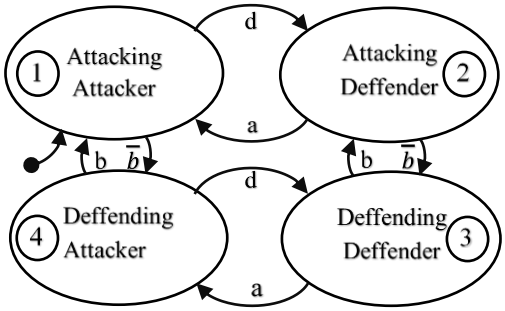

Fig 6. State machine diagram of the robot's role assignment

Even though each robot has a different role assignment, they have the same basic tasks. The basic task of every wheeled soccer robot is to pick up the ball, feed the ball to a friend, and kick the ball into the opponent's goal. In addition to these main tasks, the wheeled soccer robot must also be able to find attacking positions and finding defensive positions. The attacking position is very

important so that the robot is able to kick the ball into the opponent's goal without being hindered by the opposing robot. Meanwhile, the defensive position is very important to prevent opponents from kicking the ball into our goal. This paper discusses the two main strategies of the IRIS team, namely the attack strategy and the defensive strategy. The attacking strategy is used when the team gets the ball while the defensive strategy is used when the team loses the ball.

Basic Strategy Algorithm

The basic strategy algorithm is created as an opponent in testing the proposed algorithm. This general algorithm only has the basic functions of picking up the ball, dribbling the ball, and kicking the ball into the goal [11]. The speed of movement of the robot and the kick on goal is made the same for all the algorithms tested. When dribbling, this algorithm looks for a position with a certain distance to the goal before kicking the ball. This algorithm was created by imitating the match algorithms from other teams in the Indonesian wheeled soccer robot contest. The basic algorithm is designed to find out how effective the proposed algorithm is with the algorithm in general matches. The basic strategy algorithm used for comparison is divided into a static basic strategy algorithm and a dynamic basic strategy algorithm. The static basic strategy algorithm uses a static role assignment, where the role assignment of the soccer robot is fixed from the start to the end of the match. While the dynamic basic strategy algorithm uses a dynamic role assignment, where the role assignment of the robot can change during the game.

Advance Attack Strategy

There are two game modes for the IRIS robot either attacking mode or defending mode. When one of the team's robots gets the ball, the team changes the game mode to attacking, using equation (5). If the defender robot is holding the ball, the defender robot will feed the ball to the attacker robot. The attacker robot holding the ball will try to find the opponent's goal and kick the ball to score a goal. In a simple strategy like this, the kicks on goal often become inaccurate because the distance between the robot and the opponent's goal is too far. In addition, often the opposing robot will block the kick towards the goal so that there is no goal. The advance attack strategy developed by the IRIS team uses two parameters that determine when the robot will kick into the goal. The first parameter is the distance of the robot to the goal and the second parameter is the presence of other robots blocking the kick.

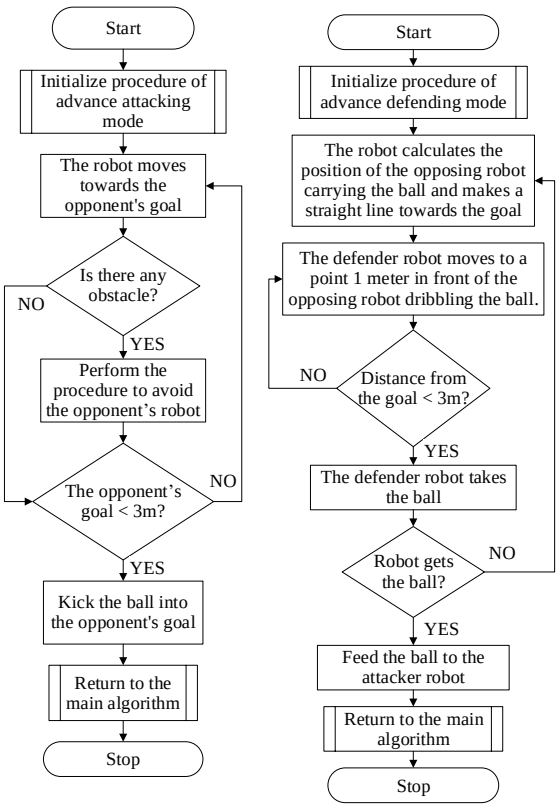

Fig 7. Flowcharts of (a) Advance attack strategy and (b) Advance defense strategy

The attack algorithm by the attacker robot is carried out with the following attack algorithm described in flowchart as seen in Figure 7(a). The advantage of this attack strategy compared to the basic strategy is in detecting obstacle and avoiding opponent's robot. The robot detects the opponent's whereabouts when moving towards the opponent's goal. When it detects an opponent, the robot sets a procedure to avoid the opponent. In the procedure, the robot will turn its back to the opponent. This is intended so that the opponent cannot grab the dribbled ball. Then the robot will move right or left based on its position in the field using equation (1). If the robot is on the left side of the field, the robot will move to the right and vice versa. This is so that the final result of the

position of the robot in the field will be closer to the opponent's goal. After the opponent has successfully passed, the robot will return to move toward the opponent's goal to verify the distance of the robot to the opponent's goal. After the shooting distance is met, the robot will kick into the goal.

Advance Defense Strategy

When all the robots lose the ball the team is in defending mode. In a simple strategy, the robot will try to catch the ball and try to grab the ball from the opposing team. This becomes difficult when we are dealing with a team that is able to move quickly to aim the ball at the goal and kick the ball. The defender will be too late to anticipate the opponent's kick. In this paper, we propose a new defensive strategy by positioning the defender on the kick line between the opposing player and our goal. The attack algorithm by the defender robot is carried out with the following defending algorithm described in flowchart as seen in Figure 7(b). The robot will calculate the estimated kick line between the opposing robot and the goal. Then the robot will make a movement trajectory to block the opposing robot on the line. The defender robot will always keep its distance from the opponent robot dribbling the ball as far as 1 meter. This distance is the ideal distance that will prevent the opposing robot from kicking the ball directly into the goal. If the opposing robot continues to move forward, the defender robot will retreat to follow the opponent's robot until it is a radius of 3 meters from the center of the goal. After reaching this line, the defender robot no longer backs up but follows the direction of the ball dribbled by the opposing robot. If the ball's distance is less than 0.5 m, the defender robot will take the ball. If the ball is successfully taken, the team mode will become attacking mode. This algorithm has been tested in the simulator and in real-world matches and has shown good results in blocking enemy attacks. The number of opponents' goals can be reduced significantly.

EXPERIMENT AND RESULT

This section discusses the tests carried out using an algorithm that has been developed in comparison with the basic algorithm used by other teams in robot soccer competitions. The test began with the simulator setup, then a basic gameplay algorithm that would be used for opponents was prepared. The match was conducted in line with the rules for the Indonesian wheeled soccer robot contest 2019.

Simulator Setup

The platform used in the IRIS robot software uses the Robot Operating System (ROS). The simulator uses a Gazebo that is compatible with ROS. The compatibility between the robot and simulator software system ensures easy implementation from robot to simulator and vice versa. The environments contained in the simulator include lighting, camera poses, physics engine, and others. There are several configurations of the simulator world used in this study, i.e. Physics engine uses ODE (Open Dynamics Engine), Max Step Size is equal to 0.004s, Real time update rate is equal to 250, and gravity is equal to -9.81 m/s^2.

An automatic program is created to run the simulator to test the match between the proposed algorithm and the comparison algorithm. There are 4 algorithms that were tested in the competition, i.e. basic static role assignments, basic dynamic role assignments, advanced static role assignments, and advanced dynamic role assignments. The first two algorithms are comparison algorithms and the next two are proposed algorithms.

Soccer Robot Match Testing

The simulator was run based on the standard time used in the Indonesian Wheeled Soccer Robot Competition, which is 2 times 5 minutes. Four algorithms were compared in combination to obtain 6 match combinations. Each match combination is tested 10 times. The results of the goals for each match are seen in Table 1. To facilitate the analysis, Table 2 is created which presents the statistics of each algorithm. These data statistics includes win, draw, loss, goal for, goal against, goal difference, and point. The point is calculated like a real soccer match, which is 3 points for a win and 1 point for a draw.

Based on the results of the match, the Advance Dynamic algorithm gets the highest point, followed by Advance Static, Basic Dynamic and the last one is Basic Static. This proves that the proposed advance strategy is able to outperform the basic strategy. The proposed algorithm of dynamic robot's role assignment can also increase the number of

goal differences as can be seen in Figure 8. The used of Dynamic robot's role assignment algorithm results a goal difference that is greater than the used of static robot's role assignment algorithm, this can be seen in the AD goal difference that is greater than the AS goal difference and the BD goal difference which is greater than the BS goal difference.

Table 1. Match Result for Advance Dynamic Algorithm (**AD**) vs Advance Static Algorithm (**AS**), **AD** vs Basic Dynamic Algorithm (**BD**), **AS** vs Basic Dynamic Algorithm (**BD**), **AS** vs Basic Static Algorithm (**BS**), **BD** vs **BS**.

No	**AD** vs **AS** (goals)		**AD** vs **BD** (goals)		**AD** vs **BS** (goals)		**AS** vs **BD** (goals)		**AS** vs **BS** (goals)		**BD** vs **BS** (goals)	
	AD	**AS**	**AD**	**BD**	**AD**	**BS**	**AS**	**BD**	**AS**	**BS**	**BD**	**BS**
1	1	2	1	0	5	0	2	0	5	0	4	4
2	1	1	1	0	1	1	3	1	5	0	6	4
3	2	1	1	1	7	0	3	0	5	1	4	4
4	0	1	4	1	10	0	7	1	4	2	2	4
5	3	2	4	0	3	1	7	1	5	0	7	4
6	4	1	4	0	4	0	4	3	7	0	3	2
7	1	3	5	1	6	0	2	1	5	0	4	5
8	0	3	2	0	7	0	6	1	5	3	5	4
9	5	3	5	1	5	0	6	1	9	1	5	2
10	3	0	8	0	7	2	4	0	7	2	6	7

Table 2. All Team Statistics. Win (**W**), Draw (**D**), Loss (**L**), Goal For (**GF**), Goal Against (**GA**), Goal Difference (**GD**), and Point (**P**).

Team	W	D	L	GF	GA	GD	P
AD	**24**	**2**	**4**	**110**	**25**	**85**	**74**
AS	**24**	**1**	**5**	**118**	**38**	**80**	**73**
BD	**5**	**2**	**23**	**59**	**119**	**-60**	**17**
BS	**3**	**3**	**24**	**53**	**158**	**-105**	**12**

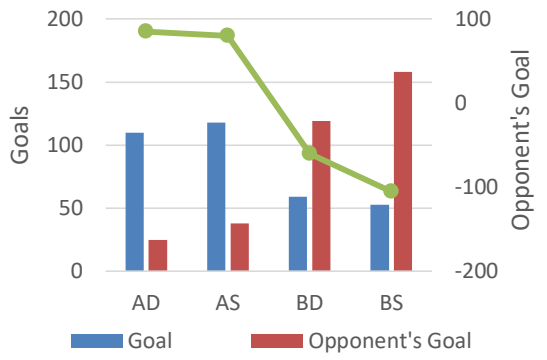

Fig 8. Graph of goal difference from match testing for four algorithms that are competed.

Besides, the coverage distance of each robot in the match is also recorded. Figure 9 shows the average coverage distance of robot 1 and robot 2 and their ratios in each match. It can be seen that the dynamic assignment algorithm has a coverage distance ratio of robot 1 compared to robot 2 which is close to the value 1, namely in AD and BD. This shows that the dynamic assignment of robot 1 and robot 2 workloads can be more balanced. By balancing the workload of this robot, each robot can work more optimally. In the assignment of the role of a static robot, the ratio of the mileage of robot 1 to robot 2 is more than 1.5, namely the AS and BS. This shows that robot 1 which functions as an attack robot has more workload than robot 2 as a defensive robot.

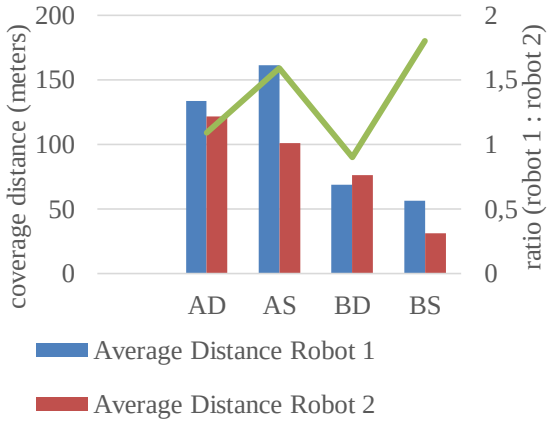

Fig 9. Comparison of the average coverage distance of robot 1 and robot 2 in each match.

To demonstrate the merit of the proposed algorithm, we employed the algorithm to the real IRIS Soccer Robot Team to compete in 2019 Indonesian wheeled soccer robot contest (KRSBI-B). Total of five real matches had been passed in the contest to become a champion. Two preliminary round matches where the IRIS team had never lost with score of 5-0 and 4-1, and three knockout matches. The knockout system with a duration of 2 x 5 minutes were the quarter-finals, semifinals and finals which had the results of 4-3, 3-0 and 4-0 respectively. Both in the simulation and the real matches yielded good results, showing more goals scored than other strategies.

CONCLUSION

Advance attack and defense strategy algorithm has been tested with a basic strategy algorithm and has shown its superiority. This algorithm is able to win almost all matches

against the basic strategy. The use of an advance defense strategy algorithm can increase the effectiveness of defense in wheeled soccer robot matches. The use of an advance attack strategy algorithm is able to increase the effectiveness of attacks in wheeled soccer robot matches. Changes in the role assignment in wheeled soccer robot matches give positive results on the overall match results. Dynamic role assignment algorithm gives better goal different against static role assignment. It also makes the player robot's workload more balanced. The Advance Dynamic (AD) algorithm, which is a combination of new algorithms with dynamic role assignment, shows the most optimal results compared to other algorithms. Although AD has fewer goals than the AS, it has a higher goal difference and a more balanced ratio of the mileage of robot 1 and robot 2. The simulator developed by the IRIS team has demonstrated its ability to test algorithms on wheeled soccer robot matches. This simulator is proven to be able to accelerate the process of developing match algorithms for wheeled soccer robots. The test results in the simulator have been verified by real matches and show similar results

ACKNOWLEGMENT

Acknowledgment is given to LPDP (Indonesia Endowment Fund for Education) with the BUDI DN scheme [PRJ-5461 /LPDP.3/2016], which has provided a scholarship for the author's continued education at Department of Electrical Engineering, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia.

REFERENCES

[1] M. Asada, H. Kitano, I. Noda, and M. Veloso, "RoboCup: Today and tomorrow—What we have learned," *Artif. Intell.*, vol. 110, no. 2, Art. no. 2, Jun. 1999, doi: 10.1016/S0004-3702(99)00024-7.

[2] Muhtadin, M. R. Arrazi, S. Ali, T. Pratama, D. S. Wicaksono, A. H. P. Putra, I. M. P. Ari, A. Maulana, O. P. Bramastyo, S. Q. Asshakina, M. Attamimi, M. Arifin, M. H. Purnomo, and D. Purwanto, "Ichiro Robots Winning RoboCup 2018 Humanoid TeenSize Soccer Competitions," in *RoboCup 2018: Robot World Cup XXII*, vol. 11374, D. Holz, K. Genter, M. Saad, and O. von Stryk, Eds. Cham: Springer International Publishing, 2019, pp. 425–435. doi: 10.1007/978-3-030-27544-0_35.

[3] G. Ficht, H. Farazi, A. Brandenburger, D. Rodriguez, D. Pavlichenko, P. Allgeuer, M. Hosseini, and S. Behnke, "NimbRo-OP2X: Adult-Sized Open-Source 3D Printed Humanoid Robot," in *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, Nov. 2018, pp. 1–9. doi: 10.1109/HUMANOIDS.2018.8625038.

[4] H. He, Z. Liang, Y. Lu, C. Xu, B. Yang, and F. Fang, "Dynamic Kick Optimization Of Humanoid Robot Based on Options Framework," in *2019 Chinese Control And Decision Conference (CCDC)*, Jun. 2019, pp. 5176–5181. doi: 10.1109/CCDC.2019.8833269.

[5] Y. Lu, Z. Liang, H. He, C. Xu, B. Yang, and F. Fang, "3D Humanoid Robot Multi-gait Switching and Optimization," in *2019 Chinese Control And Decision Conference (CCDC)*, Jun. 2019, pp. 4196–4201. doi: 10.1109/CCDC.2019.8832817.

[6] R. Dikairono, S. Setiawardhana, D. Purwanto, and T. Sardjono, "CNN-Based Self Localization Using Visual Modelling of a Gyrocompass Line Mark and Omni-Vision Image for a Wheeled Soccer Robot Application," *Int. J. Intell. Eng.*

Syst., vol. 13, no. 6, pp. 442–453, Dec. 2020, doi: 10.22266/ijies2020.1231.39.

[7] W. Na and L. Mingyong, "Cooperative Game for the Roles Assignment of the Multi-agent Robot System," in *2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, Oct. 2018, pp. 954–957. doi: 10.1109/IAEAC.2018.8577890.

[8] S. Setiawardhana, R. Dikairono, D. Purwanto, and T. A. Sardjono, "Ball Position Estimation in Goal Keeper Robots using Neural Network," *Int. Rev. Autom. Control IREACO*, vol. 13, no. 1, 2020.

[9] Setiawardhana, R. Dikairono, D. Purwanto, and T. A. Sardjono, "Navigasi Robot Penjaga Gawang Berdasarkan Prediksi Posisi dan Waktu Kedatangan Bola," *J. Nas. Tek. Elektro Dan Teknol. Inf.*, vol. 9, no. 3, Art. no. 3, Aug. 2020, doi: 10.22146/.v9i3.295.

[10] H. Shi, Z. Lin, K. Hwang, S. Yang, and J. Chen, "An Adaptive Strategy Selection Method With Reinforcement Learning for Robotic Soccer Games," *IEEE Access*, vol. 6, pp. 8376–8386, 2018, doi: 10.1109/ACCESS.2018.2808266.

[11] S. Hamidreza Kasaei, S. Mohammadreza Kasaei, S. Alireza Kasaei, and S. Amirhassan Monadjemi, "Dynamic role engine and formation control for cooperating agents with robust decision-making algorithm," *Ind. Robot Int. J.*, vol. 38, no. 2, Art. no. 2, Mar. 2011, doi: 10.1108/01439911111106363.

[12] L. de Koning, J. P. Mendoza, M. Veloso, and R. van de Molengraft, "Skills, Tactics and Plays for Distributed Multi-robot Control in Adversarial Environments," in *RoboCup 2017: Robot World Cup XXI*, Cham, 2018, pp. 277–289. doi: 10.1007/978-3-030-00308-1_23.

[13] D. Xiong, J. Xiao, H. Lu, Z. Zeng, Q. Yu, K. Huang, X. Yi, and Z. Zheng, "The design of an intelligent soccer-playing robot," *Ind. Robot Int. J.*, vol. 43, no. 1, Art. no. 1, Jan. 2016, doi: 10.1108/IR-05-2015-0092.

[14] M.-L. Wang, J.-R. Wu, L.-W. Kao, and H.-Y. Lin, "Development of a vision system and a strategy simulator for middle size soccer robot," in *2013 International Conference on Advanced Robotics and Intelligent Systems*, May 2013, pp. 54–58. doi: 10.1109/ARIS.2013.6573534.

[15] W. Yao, W. Dai, J. Xiao, H. Lu, and Z. Zheng, "A simulation system based on ROS and Gazebo for RoboCup Middle Size League," in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Dec. 2015, pp. 54–59. doi: 10.1109/ROBIO.2015.7414623.

[16] R. Dikairono, A. A. Rachman, Setiawardhana, T. A. Sardjono, and D. Purwanto, "Motion planning simulator for holonomic robot soccer platform," in *2017 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, Aug. 2017, pp. 368–371. doi: 10.1109/ISITIA.2017.8124111.

[17] A. R. Tinkar, F. K. Adianto, I. J. Kusumo, H. I. Alamsyah, R. S. Fimansyah, D. A. Samhan, M. R. Khairullah, D. Febrianto, D. Ma'ruf, and R. Dikairono, "Team Description Paper: IRIS Team 2020," *https://msl.robocup.org/*, p. 8.

[18] M. Asada *et al.*, "MSL Technical Committee 1997–2020." [Online]. Available: https://www.robocup.org

[19] R. DIKTI, “Indonesian Wheeled Soccer Robot Rulebook 2019.” RISTEKDIKTI, Dec. 2018. [Online]. Available: https://kontesrobotindonesia.id/

[20] R. Smith, “OPEN DYNAMICS ENGINE.” [Online]. Available: https://www.ode.org/

[21] “Gazebo.” http://gazebosim.org/ (accessed Oct. 22, 2020).

.