Coach Application for Soccer Robot

Hendawan Soebhakti[1], Mochamad Rizal Fauzi[2], Tajdar Hal Ata[3], and Wanda Eka Kurniawan[4]

{hendawan@polibatam.ac.id[1], fauzimochamadrizal2@gmail.com[2], tajdarhalata23@gmail.com[3], and scoeg12@gmail.com[4]}

Barelang Robotics Artificial and Intellegence Lab (BRAIL), Department of Electrical Engineering, Politeknik Negeri Batam, Batam, Indonesia[1,2,3,4]

Abstract. The RoboCup Middle-size League (MSL) competition involves a game of soccer played by five robots, with four strikers and one goalkeeper. The competition is held annually by the RoboCup Federation and is fully automated, with no human intervention allowed. To monitor and control the robots on the field, an application called "basestation" is used. This application must be able to process data from the robots, the referee, and the environment to regulate the behavior of each robot. The application must have a system of intelligence to process data and send commands to each robot, using a behavior tree system to regulate behavior on the field. The communication system between the application and the robots uses the UDP method and runs on the IEEE 802.11 standard, with network access through ports. The application must also be able to prepare tasks such as determining signals from the referee and specifying team colors, as well as monitoring the state of the game and each robot's actions.

Keywords: Coach Application, Basestation, and midle size league

1 Introduction

RoboCup [1] is a branch of an annual international competition event held by the RoboCup Federation. The competition has various branches, one of which is the Middle-Size League (MSL) or soccer robots with wheels. In this branch, five robots compete in a field-sized 22 x 14 meters, with four striker robots and one goalkeeper robot. The robots need to make decisions on the field based on their current position, the enemy robots' position, the goal's location, the ball's position, and other factors. Therefor this research goal is to make a software for monitoring and controlling the robots on the field, and the application named "basestation". Basestation is needed to act as their coach [2 - 3]. This application must have the ability to control all the robots on the field and monitor everything that is happening with the robots, such as their current position, the position of their allies, the ball position, the goal position, and the behavior of the robots themselves [4]. This application is crucial because the robots must operate autonomously and without human intervention during the competition.

To make sure that the basestation can communicate with the robots, several studies were conducted to find solutions to the requirements that the system must meet. The first step was to establish a communication system with the assistant referee's app, also known as the referee [5

ICAE 2023, November 07, Batam, Indonesia
Copyright © 2024 EAI
DOI 10.4108/eai.7-11-2023.2342934

- 6]. This system allows the referee to communicate with the robots by giving orders on the field and having the assistant referee press the command button in the basestation application. This communication system uses TCP as its method [7]. The next challenge was to find a suitable communication method to use with the robots, considering that more than one robot is involved and this could cause network instability. After comparing various methods, the team chose to use the UDP method [8 - 9] as it can communicate simultaneously with many clients using just one data packet delivery [10 - 11]. All communication systems run with the IEEE 802.11 standard and require network access through ports.

Each robot in the competition has its own role [12], and the basestation must consider the behavior of each robot to ensure they work together effectively, as coordination is crucial in playing soccer [13]. The basestation must have an intelligent system to process data, including data sent by the assistant referee and data from every robot on the field. This data is processed through a method called the behavior tree [14], which is the core of the data processing process and regulates the behavior of each robot on the field. The behavior tree process is responsible for sending commands to each robot using a switch for each task in progress. This method must be able to complete several preparation tasks, such as determining signals from the referee, including start and stop signs, specifying the team's colors, and the opposing side of the goal [15]. Additionally, this method must be able to complete several tasks during the game, such as monitoring the current state of the game, the position of the ball and the robots, whether a robot is holding the ball or not, the action that the robots are currently executing, the team color, and the actual goal position.

The implementation of this system requires a thorough understanding of the robots' hardware and software components, as well as the communication systems used. The behavior tree method also requires a deep understanding of artificial intelligence techniques and their applications in robotics [16]. All of these elements must work together seamlessly to provide the robots with the support they need to succeed in the competition and showcase their capabilities to the world.

2 Method

In working on this application, several methods need to be completed to run correctly, and some optimizations are needed to make the app run properly.

2.1 Referee Communication

The RoboCup Middle Size League competition committee has developed an application named "Referee Box" [7] to improve the communication system between the Referee and the participating teams' basestations. This aims to ensure clarity and consistency that could occur due to varying communication methods used by each basestation . The application utilizes TCP (Transmission Control Protocol) [5 – 6] as its data transmission method, which creates a virtual connection between the sender and receiver devices [11]. This connection is established by the client first sending a synchronization request to the server, ensuring a stable and reliable communication link.

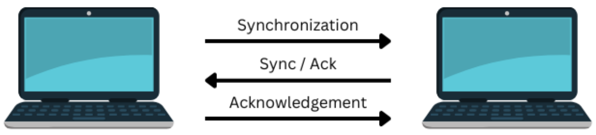
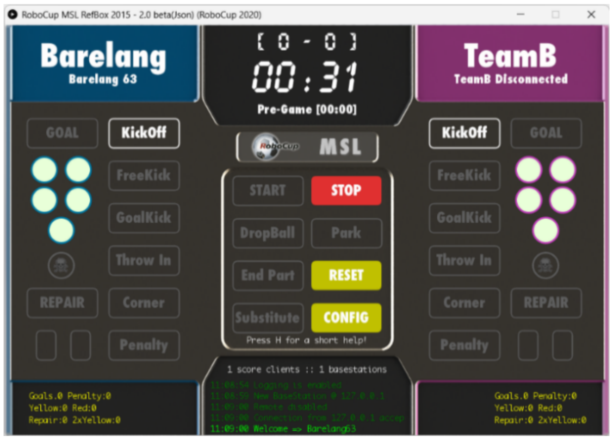


Fig. 1. TCP synchronization process

To maintain the quality of communication and prevent disruptions, the system operates through a local router connected directly to the field via a LAN cable. This eliminates wireless communication, which could compromise the communication between the Referee Box and the basestations. Additionally, the Basestation application must be capable of processing data from two versions of the Referee Box [3 – 4]. The first version sends a single character command, while the latest version sends commands in JSON format with comprehensive information for both basestations. This enhances the efficiency and effectiveness of the communication system, ensuring smooth and seamless coordination during the competition.



```
1 score clients :: 1 basestations
11:08:54 Logging is enabled
11:08:59 New BaseStation @ 127.0.0.1
11:09:00 Remote disabled
11:09:00 Connection from 127.0.0.1 accep
11:09:00 Welcome => Barelang63
```

```
Goals.0 Penalty:0
Yellow:0 Red:0
Repair:0 2xYellow:0
```

```
Goals.0 Penalty:0
Yellow:0 Red:0
Repair:0 2xYellow:0
```

Fig. 2. Referee box application

2.2 Robot Communication

In the system for communicating with robots, the socket method is utilized [5]. However, the protocol used differs from that used for communicating with referees. Coordinating communication between a group of robots can be challenging [8] [10] due to the high volume of data traffic and the need for two-way communication between the robot and the basestation [9].

Wireless communication between the basestation and the robots is maintained continuously with a 50 milliseconds delay for each new data transmission. The socket programming method is still used, but the UDP (User Datagram Protocol) method is employed. To accommodate the heavy network traffic, multiple robots can communicate with the basestation simultaneously, sending and receiving data in large amounts.

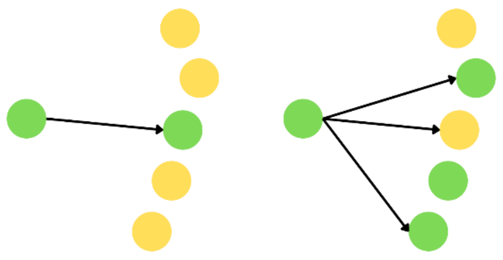


Fig. 3. Representation of sending data packets

As shown in the illustration, two methods for sending data packets exist [11]. The first method involves sending data packets to a single device without involving other devices, as seen on the left. The second method involves sending data packets directly to multiple devices, as seen on the right, but each device must first register its original IP address to join the multicast socket address, which is the same as the socket address used by the data packet sender.

2.3 Strategy System

Controlling multiple robots simultaneously is complex [17 - 18]. Each action and command must be executed appropriately to prevent collisions and enable robots to work together [12 – 13] effectively. The behavior tree [14] provides an alternative task-switching system in an automated agent, which has many benefits, such as reusable code, understandable diagrams, and flexible reactions to code changes [15][16][19]. However, the behavior tree has drawbacks, such as complex implementation and time-consuming condition checks.

The behavior tree algorithm starts by executing each node in sequence [20 – 21], known as the control flow node [22 – 25], determined by parent and child relationships. The signal to start execution is given to the root node, with the frequency determined as necessary. Upon execution, the node returns a running value. If the execution successfully reaches its destination, it returns a success value; otherwise, it fails [14][20].

The behavior tree uses two control flow nodes, sequence and fallback, followed by two execution nodes, action, and condition. Sequence nodes are marked with "->" and start by executing child nodes from left to right. Execution continues with a failure or running value depending on the steps taken and succeeds if all child nodes return a success value, similar to a logic gate AND.

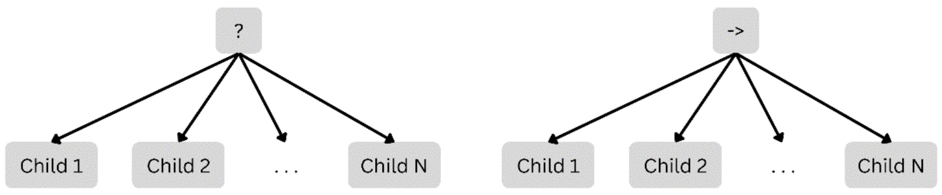


Fig. 4. Fallback (left) and sequence (right) node graphical representation

The figure on the right depicts the sequence node. The action node is on the left, indicated by a "?" mark. Like the sequence node, the action node begins by executing its child nodes from left to right. If a running value is returned, execution continues. If a failure value is returned, the next child node is attempted until one returns a success value, allowing the node to finish. It operates like an OR logic gate [14][20].

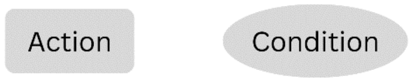


Fig. 5. Action and Condition node graphical representation

The figure depicts the two types of child nodes in use [21 – 22]. The action node executes commands and returns success, running, or failure values. The condition node checks conditions and returns either true or false. Both child nodes work together to create a condition check that can be executed in an open or closed loop.

3 Results and Discussion

Testing systems and communication strategies will be carried out separately to obtain detailed data for each system.

3.1 Robot Communication

Communication testing involves evaluating the performance of a communication system to determine its efficiency and effectiveness in delivering data. This analysis is performed by examining various Quality of Service (QoS) metrics such as Throughput, Delay, Jitter.

Table 1. Data communication testing result

Testing	Throughput	Delay	Jitter
1	71 kb	7.34 ms	7.26 ms
2	70 kb	7.51 ms	-2.52 ms
3	67 kb	7.81 ms	-1.85 ms
4	68 kb	7.67 ms	1.14 ms
5	68 kb	7.74 ms	5.86 ms

This analysis was performed during an online competition, with data from five tests, each lasting 3 minutes. The results show that the average Throughput is 68.8 kb with a standard deviation of 1.48 kb, indicating stability and no significant fluctuations in Throughput. A stable Throughput is crucial for a seamless network experience. The average delay was 7.54 ms with a standard deviation of 0.19 ms, suggesting stability in delay values with no significant variations. This is essential for a consistent network experience. The average Jitter was 1.66 ms with a standard deviation of 4.14 ms, indicating significant variability in Jitter with values ranging from -2.52 ms to 7.26 ms. Jitter is a crucial metric for real-time applications where even minor delays can affect the user experience. This variability in Jitter suggests potential network congestion or other performance issues.

In conclusion, the analysis reveals that the network has a stable Throughput and Delay but experiences variability in Jitter. To improve network performance, addressing the sources of Jitter variability may be necessary. Regular monitoring of network performance and tracking changes in Throughput, Delay, and Jitter can provide valuable insights into the network's overall health.

3.2 Strategy System

A diagram is crucial in creating a successful and well-structured strategy, mainly when there are many conditions to be taken into account in the field and with the robots. A behavior tree diagram is especially beneficial in this regard, providing a clear visual representation of the systems employed in the behavior tree to ensure all conditions are thoroughly checked. The process of designing a strategy using a behavior tree involves creating a flow chart to outline the execution order of each node in the behavior tree. This flow chart acts as the foundation for the behavior tree and provides a clear guide for executing each node.

Before executing each node in the behavior tree, it is vital to check the referee's commands to ensure that all actions taken by the robots are in line with the rules of the game. The flow chart then dictates the execution of strategies, from checking conditions in the field to monitoring the robots' performance. With a clear visual representation of the systems and nodes to be executed, the behavior tree becomes a powerful tool for creating an effective strategy and achieving success in various settings.

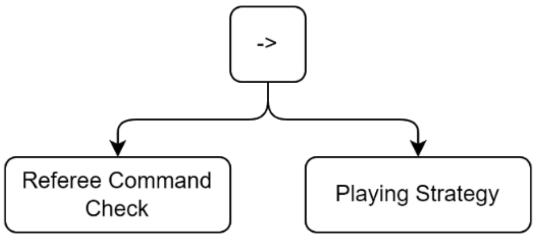


Fig. 6. Behavior tree summary diagram design

The figure shown above summarizes all the diagrams that have been produced. The system depicted in this diagram is designed to receive every command sent from the referee box and is a crucial component of the overall strategy. To ensure its effectiveness, a behavior execution test is conducted to verify that the system functions as intended and that all necessary conditions are being considered. The test results and this diagram can be used to refine and optimize the strategy.

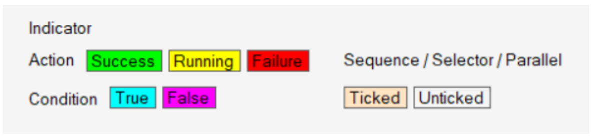


Fig. 7. Execution indicator on debugging page

The figure above show behavior tree execution indicators for each node. It shows the kick-off process, from stop to completion, and helps understand the flow and execution of systems in the behavior tree. The diagram is vital for refining the overall strategy and ensuring successful execution by considering all necessary conditions.

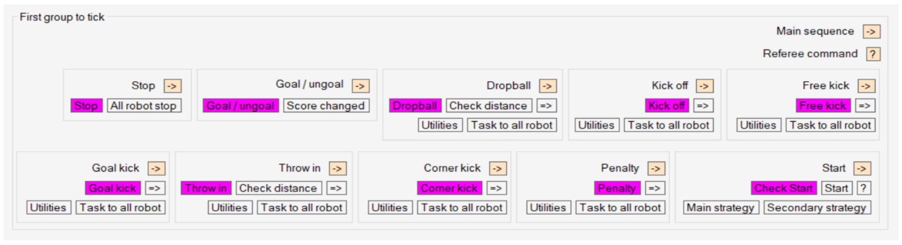


Fig. 8. System did not find the appropriate command

The figure above shows the behavior of executing each node in the parent node command check with the condition that the data from the referee does not meet the node condition check or the referee data still does not exist.

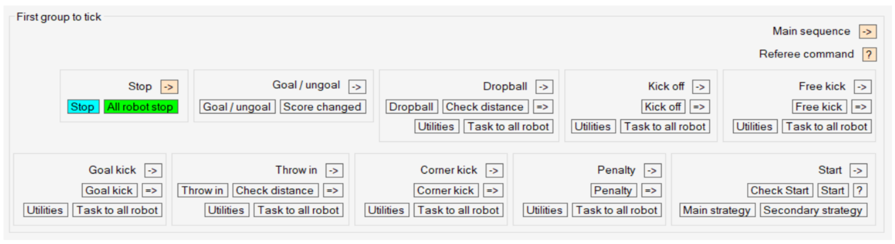


Fig. 9. The system finds a matching command

The figure above shows the behavior condition when finding a matching command from the node condition check and immediately executing the next node due to the conditions of its parent, the sequence node.

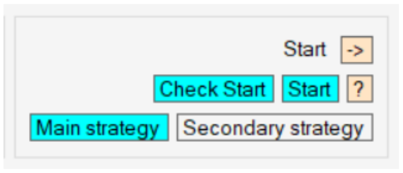


Fig. 10. The system executes the start command

When the behavior executes the start command, it will start executing the second group, the continuation of the parent main sequence node.

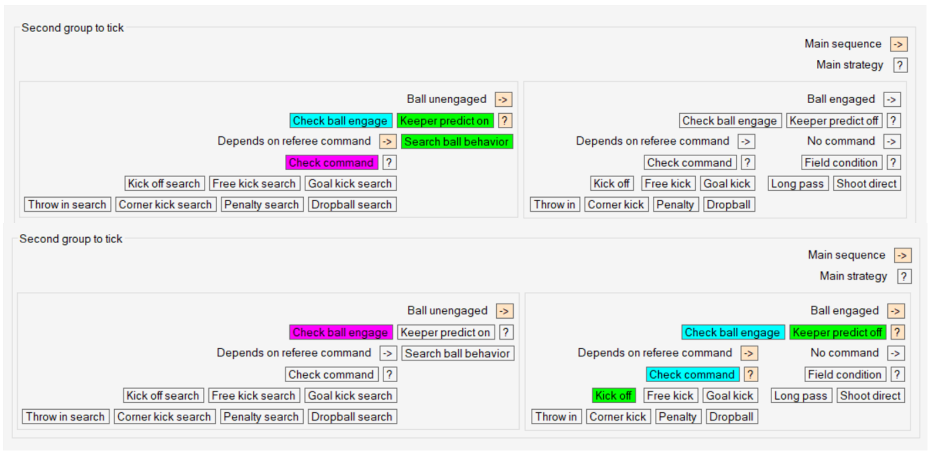


Fig. 11. The system has executed the strategy to find the ball and has obtained the ball

The figure above displays that the kick-off strategy has been successfully executed. The robot runs in coordination and will continue the game until it gets a new command. The following is a table of periodic testing with data received from the referee box because the referee box uses a direct cable communication system.

As seen in Table 2, the football robot strategy system underwent comprehensive testing with nine scenarios in an offline competition, revealing a robust overall performance. Successful outcomes were observed in kick-off positioning, accurate ball passes to front teammates, strategic ball chasing without possession, and precise stopping of robot movement upon command. However, challenges arose when attempting longer passes to the farthest teammate and encountering opponent blockages during goal kicks. Despite minor setbacks, the system effectively positioned itself during goal kicks and skillfully defended against opponents' attempts. The testing provided valuable insights into the system's strengths and weaknesses, paving the way for future enhancements. Further refinements are aimed at optimizing these aspects to ensure consistent and improved performance in upcoming competitions.

Table 2. Behavior tree system test result

Test Case	Referee Box Command	Field Condition	Expected Outcome	Robots Action	Status	Interference (if any)
1	Kick off	Ball not in possession, see the ball	Robot is positioned correctly for kick off	Positioning between the balls	Success	
2	Kick Off, start	Ball in possession and teammate is nearby	Ball is successfully passed to a teammate	Pass to front teammate	Success	
3	Start	Ball in possession, position in team territory and team is in enemy territory	Pass the ball to the farthest teammate	Pass to farthest teammate	Failure	The ball has been taken by the opposing team
4	Start	Ball not in possession, see the ball	Robot searching for the ball and the other stand guard	Chasing the ball, positioning	Success	Blocked by the opposing team
5	Stop	Ball not in possession, see the ball	Robot stop	All robot stop	Success	
6	Goal kick	Ball not in possession, see the ball	Robot is positioned correctly for goal kick	Positioning near the ball and in the center of the field	Success	
7	Goal kick, start	Ball in possession, position in team territory	The robot passes the ball to the teammate in the center	Robot kicks the ball towards the center of the field	Success	Blocked by the opposing team, the ball left the field
8	Enemy goal kick	Ball not in possession, see the ball	Robot blocks opponents from looking forward	Robots block opponents and guard in front of the goal	Success	
9	Enemy goal kick, start	Ball not in possession, see the ball	Robot searching for the ball and the other stand guard	Chasing the ball, positioning	Success	

The football robot strategy system underwent comprehensive testing with nine scenarios in an offline competition, revealing a robust overall performance. Successful outcomes were observed in kick-off positioning, accurate ball passes to front teammates, strategic ball chasing without possession, and precise stopping of robot movement upon command. However, challenges arose when attempting longer passes to the farthest teammate and encountering opponent blockages during goal kicks. Despite minor setbacks, the system effectively positioned itself during goal kicks and skillfully defended against opponents' attempts. The testing provided valuable insights into the system's strengths and weaknesses, paving the way for future enhancements. Further refinements are aimed at optimizing these aspects to ensure consistent and improved performance in upcoming competitions.

4 Conclusion

In conclusion, the software results is the Communication Testing show a stable Throughput and Delay but with variability in Jitter, suggesting potential network congestion. Regular monitoring is necessary to track changes in network performance. The football robot strategy system performed well in most test cases, with successful scenarios in the kick-off, stop, goal kick, and enemy goal kick. However, room for improvement remains in scenarios where the robot had to pass the ball to a teammate or position itself to guard the goal. The testing provided valuable insights into the system's strengths and weaknesses, which will be helpful in refining and improving the system's performance.

References

[1] M. Asada *et al.*, “Middle Size Robot League Rules and Regulations for 2022,” 2021

[2] M. A. Haq, I. K. Wibowo, B. S. B. Dewantara, M. M. Bachtiar, and K. Anwar, “The base station application of ERSOW team for communication between robots,” in *Proceedings of the 2020 27th International Conference on Telecommunications, ICT 2020*, Oct. 2020. doi: 10.1109/ICT49546.2020.9239551.

[3] S. Arya and C. G. / 165114029, “Visualization Of Robot And Ball’s Position With The Soccer Robot’s Strategy.”

[4] N. M. Figueiredo, A. J. R. Neves, N. Lau, A. Pereira, and G. Corrente, “Control and monitoring of a robotic soccer team: The base station application,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2009, vol. 5816 LNAI, pp. 299–309. doi: 10.1007/978-3-642-04686-5_25.

[5] D. Arie Widhining Kusumastutie and F. Alif Fiolana, “Design of Wheeled Football Robot Coordination System at Base Station Using TCP / IP Perancangan Sistem Koordinasi Robot Sepak Bola Beroda Pada Base Station Menggunakan TCP / IP,” 2020.

[6] W. Stallings, *Data and computer communications*. Pearson/Prentice Hall, 2007.

[7] T. Ardhiansyah, I. Syarifuddin, M. R. Naufal, Y. Pramono, and O. T. Hartatik, *Pergerakan Otomatis Robot Sepak Bola Beroda Melalui Komunikasi dengan Referee Box Menggunakan Base Station*. 2017.

[8] Y. D. Santoso, S. Nugroho, and H. K. Wardana, “Komunikasi Antar Robot Sepakbola Beroda Menggunakan Udp Multicast Communication Between Wheeled Soccer Robot Using Udp Multicast.”

[9] T. Vedavathi*, R. Karthick, R. S. Selvan, and P. Meenalochini, "Data Communication and Networking Concepts in User Datagram Protocol (UDP)," *Int. J. Recent Technol. Eng.*, vol. 8, no. 5, pp. 2765–2765, Jan. 2020, doi: 10.35940/ijrte.D8758.018520.
[10] I. Damayanti, S. Siregar, and M. I. Sani, "UDP Protocol for multi-task assignment in 'void loop' robot soccer," *Telkomnika (Telecommunication Comput. Electron. Control.*, vol. 17, no. 2, pp. 986–994, 2019, doi:10.12928/TELKOMNIKA.V17I2.11782.
[11] A. Makinun Amin, J. Sahertian, and A. Sanjaya, "Perancangan Sistem Komunikasi Data Robot Sepak Bola Dalam Kontes Robot Sepak Bola Indonesia Beroda (KRSBI) Oleh: Dibimbing oleh : 1," 2019.
[12] N. Lau, L. S. Lopes, G. Corrente, N. Filipe, and R. Sequeira, "Robot team coordination using dynamic role and positioning assignment and role based setplays," *Mechatronics*, vol. 21, no. 2, pp. 445–454, 2011, doi: 10.1016/j.mechatronics.2010.05.010.
[13] N. Lau, S. Lopes, N. Filipe, and G. Corrente, "Roles, Positionings and Set Plays to Coordinate a RoboCup MSL Team," 2009.
[14] M. Colledanchise and P. Ögren, *Behavior Trees in Robotics and AI*. CRC Press, 2018. doi: 10.1201/9780429489105.
[15] *2019 2nd China Symposium on Cognitive Computing and Hybrid Intelligence (CCHI)*. IEEE, 2019.
[16] M. Iovino, E. Scukins, J. Styrud, P. Ögren, and C. Smith, "A Survey of Behavior Trees in Robotics and AI," May 2020, [Online]. Available: http://arxiv.org/abs/2005.05842.
[17] R. A. Agis, S. Gottifredi, and A. J. García, "An event-driven behavior trees extension to facilitate non-player multi-agent coordination in video games," *Expert Syst. Appl.*, vol. 155, Oct. 2020, doi: 10.1016/j.eswa.2020.113457.
[18] 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 *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018, vol. 11175 LNAI, pp. 277–289. doi: 10.1007/978-3-030-00308-1_23.
[19] M. Colledanchise, R. Parasuraman, and P. Ögren, "Learning of behavior trees for autonomous agents," *IEEE Trans. Games*, vol. 11, no. 2, pp. 183–189, Jun. 2019, doi: 10.1109/TG.2018.2816806.
[20] R. Ghzouli, T. Berger, E. B. Johnsen, S. Dragule, and A. Wasowski, "Behavior trees in action: A study of robotics applications," in *SLE 2020 - Proceedings of the 13th ACM SIGPLAN International Conference on Software Language Engineering, Co-located with SPLASH 2020*, Nov. 2020, pp. 196–209. doi: 10.1145/3426425.3426942.
[21] J. Marcello, "Penggunaan Behavior Tree untuk Menentukan Aksi NPC di dalam Gim."
[22] R. H. Abiyev, I. Günsel, N. Akkaya, E. Aytac, A. Çağman, and S. Abizada, "Robot Soccer Control Using Behaviour Trees and Fuzzy Logic," in *Procedia Computer Science*, 2016, vol. 102, pp. 477–484. doi: 10.1016/j.procs.2016.09.430.
[23] ArvinAgah, "Robots Playing to Win: Evolutionary Soccer Strategies," 1997.
[24] Sarvasiddhi Sabitha, "A Simple Yet Effective Failure Resilient Robotic Soccer Strategy," 2015.
[25] C. Floriana Pana, G. Bizdoaca, I. C. Rescanu, and M. Niculescu, "Strategy Planning For Mirosot Soccer's Robot."