Decision-making strategy for multi-agents using a probabilistic approach: application in soccer robotics

Hoang Anh Pham, Valentin Gies, Thierry Soriano

Decision-making strategy for multi-agents using a probabilistic approach: application in soccer robotics

Hoang Anh PHAM
University of Toulon
Toulon, France
hoang-anh.pham@univ-tln.fr

Valentin GIES
University of Toulon
Toulon, France
valentin.gies@univ-tln.fr

Thierry SORIANO
University of Toulon
Toulon, France
thierry.soriano@univ-tln.fr

***Abstract*—The efficient coordination of soccer robots is a complex topic because there are numerous possible scenarios in the game, and the state of the robots can change rapidly. It requires the robot to be able to analyze and make decisions in a short time. In this article, we first use Dec-POMDP to describe the actions and states of the soccer robot team. Secondly, we introduce a probabilistic approach so that the robots can quickly make decisions corresponding to specific situations. More specifically, we present a method for calculating and evaluating the expected points corresponding to each particular action for each robot. The robots then choose the actions with the highest expected points. Finally, we have developed a simulator based on the digital twin approach to verify the proposed strategies on simulation models and implement these strategies rapidly on real robots.**

***Index Terms*—Dec-POMDP, coordinated decision-making, multi-agent teams, RoboCup Soccer.**

I. Introduction

In recent years, there has been considerable interest in coordinated robots. One of the research problems is that robots have to make their own decisions in a limited time based on information received from the environment. One case study is a collaborative robot soccer competition called RoboCup [1]. The initial mission was to create a team of robots capable of winning against the human champions of the Football World Cup in 2050. In this case, the situation change in the game (i.e., environmental change) combined with robot strategies (i.e., attack or defense) is studied. Robots must make their own decisions as quickly as possible and in real-time. Currently, there are several main research directions for solving this problem, such as:

Q-Learning is a popular reinforcement learning algorithm that can be applied to coordinate multiple robots effectively. Using Q-learning, the robots can learn from their interactions with the environment and each other to make informed decisions. In [2], the team's performance in playing is enhanced by using modular Q-learning. A mediator module is employed to select the most appropriate action for a robot, considering the Q-value from each learning module. The mediator takes into account state information like the distance between the ball and the robot, as well as the angle between the robot's heading and the desired angle, along with the Q-value. In [3], a fundamentally different approach is proposed, namely Hyper-Q Learning, in which values of mixed strategies rather than base actions are learned, and in which other agents' strategies are estimated from observed actions via Bayesian inference. In [4], they proposed a self-learning cooperative strategy for a robot soccer game by using an adaptive Q-learning method which is modified from the traditional Q-learning and the fuzzy method.

Multi Q-Learning enables the robots to create a shared understanding of the environment and learn the best actions to take in various situations in multi-robot coordination. Each robot maintains its Q-table, which helps it determine the most appropriate action based on its current state and the collective knowledge of the team. In [5], they present a new algorithm called multi-Q-learning to attempt to overcome the instability seen in Q-learning. Their results show that in most cases, Multi Q-learning outperforms Q-learning, achieving average returns up to 2.5 times higher than Q-learning and having a standard deviation of state values as low as 0.58.

Deep Q-Learning is a powerful technique used to coordinate multiple robots efficiently. By employing deep neural networks, the robots can learn complex strategies and make better decisions in dynamic and unpredictable environments. In the context of multi-robot coordination, each robot maintains a Deep Q-network (DQN) that takes in environmental observations and outputs Q-values for different actions. These Q-values represent the expected rewards for taking specific actions in specific states. In [6], they seek to employ a renowned reinforcement learning algorithm, the Deep Q-Network, in the AI Soccer game in their endeavor to enhance performance. AI Soccer is a captivating 5vs5 robot soccer competition where each participant devises an algorithm to control five robots in their team, aiming to outmaneuver the opposing participant.

Deep Reinforcement Learning (DRL) is a cutting-edge approach used to coordinate multiple robots seamlessly. By leveraging deep neural networks, DRL enables the robots to learn complex behaviors and strategies in dynamic environments. Each robot is equipped with a deep reinforcement learning agent in the context of multi-robot coordination. These agents interact with the environment and receive feedback in the form of rewards based on their actions. By maximizing the cumulative rewards over time, the robots learn optimal coordination policies. In [7], DRL was applied to low-cost humanoid robots, instructing them in 1v1 soccer play.

However, the limitation of the above methods require thou-

sands of training sessions for robots. Besides, there is also a limitation in deploying these training models from the simulation to the actual model. The main contributions of this paper can be summarized as follows

- We introduce a decentralized strategy using a probabilistic approach where robots can make their own decisions. Compared with the centralized approach [8], [9], our approach allows the robot to maximize its ability to adapt to difficult conditions (such as limited communication). To the author's knowledge, in the RoboCup competition, we were the first team to introduce this approach. Our approach also allows real-time decision-making, without requiring execution on high-performance hardware.
- We have additionally designed a simulator based on the Digital Twin approach. This simulator enables us to swiftly generate and assess algorithms for strategic coordination among robots, which can then be directly applied to real robots. A demonstration video can be found at the following link https://youtu.be/YEpfklV8hRg.

The paper is organized as follows: Section 2 describes the decentralized partially observable Markov decision Processes. Section 3 discusses the decision-marking strategy for soccer robots using a probabilistic approach. Section 4 presents a Digital twin simulation software. Finally, section 5 provides conclusions and future work.

II. Decentralized Partially-Observable Markov Decision Processes

Decentralized partially observable Markov decision processes (Dec-POMDPs) are a generalization of both POMDPs and MDPs [10], [11], [12], designed to handle multi-agent environments.

A Dec-POMDP represents a team of agents that must collaborate to accomplish a task by taking individual actions based on their local observations across a series of time steps. The agents share a common reward function that defines their collective objective, but it is typically unknown during execution. The execution is decentralized because each agent must choose its own action at each time step without being aware of the actions or observations of other agents. Moreover, the problem is partially observable because although the framework assumes the presence of a Markovian state at each time step, the agents do not have access to it.

A Dec-POMDP is defined by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, O \rangle$, where $\mathcal{S}$ is a finite set of states, $\mathcal{A}$ is a finite set of actions for each agent, $\mathcal{T}$ is a state transition probability function. $\mathcal{R}$ is a reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$, that maps states and joint actions to real numbers and is used to specify the agents' goal, Ω is a finite set of observations for each agent, $\mathcal{O}$ is an observation probability function.

III. Decision-making strategy for soccer robots using a probabilistic approach

In this article, we use Dec-POMDP to be able to describe the whole relationship between the observed parameters of each robot, and the actions. The strategy for the reward points for each action depends on the specific situation.

The use of a Dec-POMDP approach in modeling a robot football team can provide various advantages, such as:

- Coordination: To attain a shared goal, robots must collaborate effectively. Dec-POMDP provides a modeling framework for capturing player interactions, allowing them to synchronize their actions and work cohesively towards a collective objective.
- Uncertainty: In a football robot game, numerous factors come into play, including the strategies of the opposing team and uncertainties in the game state. Dec-POMDP offers a means to model and handle this uncertainty, empowering the team to make decisions based on the probabilities of various outcomes.
- Partial observability: Robots possess only limited information about the game's state, such as the ball's location, their opponents' positions, and the current score. Dec-POMDP can effectively model this partial observability, allowing players to make decisions based on their own observations and those of their teammates.
- Flexibility: The football robot game is a dynamic and intricate challenge, demanding robots to adjust to ever-changing situations on the field. Dec-POMDP can effectively model this adaptability, allowing the team to modify its strategy in response to the evolving game environment.

Assumption 1: The group of soccer robots operates in a discrete time and the observed state of the system is discrete. The assumption 1 is essential for creating and analyzing the proposed model, and for designing control strategies that work well with the discrete nature of the system. Such strategies may differ from those used in continuous-time systems and require specific mathematical techniques.

Assumption 2: In our present study, we assume that the robots have a common set of observable information. Through assumption 2, the robots share their individual observations via communication and therefore can maintain the same internal state. We then can calculate the probability of successful scoring of each robot at each time k.

In this study, we have not chosen a policy associated with Bellman's equation [13] because it seemed more complex to implement in real-time, and we have proposed the expected point $\mathcal{Q}_{a_i,j}(k)$ for each robot j corresponding to each action a_i, which is defined as follows

$$\mathcal{Q}_{a_i,j}(k) = \mathcal{R}\left(s(k), a(k)\right) \times \mathcal{P}\left(s(k), a(k)\right) \tag{1}$$

where $\mathcal{R}(s(k), a(k))$ is the reward for each action of a robot. The reward depends on the following criteria:

- The robot's ability to score goals.
- Its approach to the opposing goal.
- Its role in augmenting the defense against the opponents' goal.
- The strategic choices made by the coach when the robot team is required to emphasize offensive or defensive maneuvers.

$\mathcal{P}(s(k), a(k))$ represents the probability of a goal being scored by each robot and is contingent upon the following factors:

- The robot's positioning relative to the ball.
- The likelihood of encountering contact with opponents, other robots, and the ball.
- The orientation of the robot with respect to the direction of the ball (greater alignment results in faster movement).
- The convenience of the robot's positioning for scoring.

To provide a more detailed explanation of the above components for constructing the probability $\mathcal{P}(s(k), a(k))$, we define a few component probabilities $\mathcal{P}_1(k)$, $\mathcal{P}_2(k)$, $\mathcal{P}_3(k)$, $\mathcal{P}_4(k)$, $\mathcal{P}_5(k)$, as follows:

$\mathcal{P}_1(k)$ is the probability of avoiding interception of the ball by the opposing robot if the robot with the ball passes the ball to the other robots in the team. To do this, let t^{int} be the shortest time taken by the opposing robot to reach the ball's trajectory when the robot passes the ball, and $t_{i,j}$ is the time taken for the ball to pass from the passing robot i to the receiving robot j (see Figure 1).

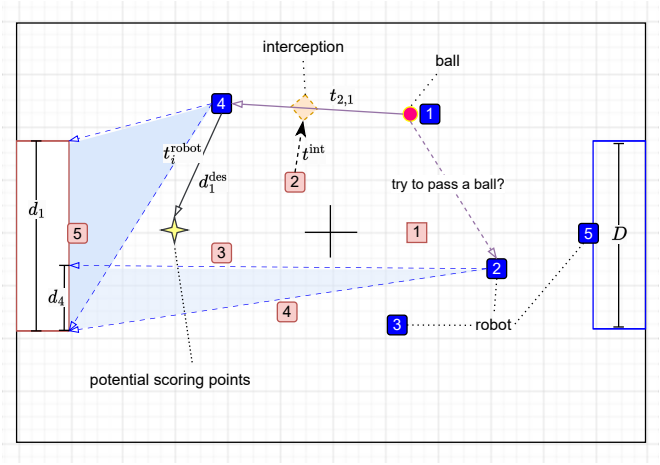

Fig. 1. Illustrating Component Probabilities to Aid Robots in Decision-Making

$\mathcal{P}_1(k)$ is defined as

$$\mathcal{P}_1(k) = \begin{cases} 1 & \text{if } t^{int} \geq 2t_{i,j} \\ 0 & \text{if } t^{int} \leq t_{i,j} \\ \dfrac{t^{int} - t_{i,j}}{t_{i,j}} & \text{if } t_{i,j} < t^{int} < 2t_{i,j} \end{cases} \tag{2}$$

$\mathcal{P}_2(k)$ is the probability of scoring a goal for each robot. This probability depends on the angle between the robot's position and the opponent's goal. Let d_i be the distance corresponding to the robot's free view of the goal, D is the length of the goal (see figure 1), $\mathcal{P}_2$ is defined as

$$\mathcal{P}_2(k) = \begin{cases} 1 & \text{if } d_i = D \\ d_i/D & \text{if } d_i < D \end{cases} \tag{3}$$

$\mathcal{P}_3(k)$ is the probability of the robot reaching the highest potential scoring position. Depending on the coach's strategy, a number of potential scoring positions can be predetermined. Therefore, probability $\mathcal{P}_3(k)$ represents the possibility of which robot in the team is able to move to that location the fastest. Let d_i^{des} be the distance of robot i to the potential scoring position (see Figure 1). Then, $\mathcal{P}_3(k)$ is defined as follows

$$\mathcal{P}_3(k) = \begin{cases} 1 & \text{if } d_i = \min(d_i^{\text{des}} | i = 1...5) \\ 0 & \text{if } d_i = \max(d_i^{\text{des}} | i = 1...5) \\ \dfrac{d_i - d_{\min}}{d_{\max} - d_{\min}} & \text{if } d_{\min} < d_i < d_{\max} \end{cases} \tag{4}$$

$\mathcal{P}_4(k)$ is the probability, that the robot has not been intercepted by the opposing robot during its movement. This probability represents the robot's ability to move towards the ball position (in case of defense or ball search) or it may also move towards a potential goal position. It is similar to the calculation of probability $\mathcal{P}_1(k)$. Let t_i^{robot} be the expected movement time of the robot i, t^{int} is the shortest time the opposing robot may intercept on the movement trajectory. Consequently, the probability $\mathcal{P}_4(k)$ is defined as follows

$$\mathcal{P}_4(k) = \begin{cases} 1 & \text{if } t^{int} \geq 2t_i^{robot} \\ 0 & \text{if } t^{int} \leq t_i^{robot} \\ \dfrac{t^{int} - t_i^{robot}}{t_i^{robot}} & \text{if } t_i^{robot} < t^{int} < 2t_i^{robot} \end{cases} \tag{5}$$

$\mathcal{P}_5(k)$ is the probability of the robot with the highest ability to reach the ball (the fastest in this study). This probability is calculated as a function of the distance between robot i and the ball d_i^{ball} and is not constrained by opposing robots at the time of calculation. This probability is calculated as follows

$$\mathcal{P}_5(k) = \begin{cases} 1 & \text{if } d_i = \min(d_i^{\text{ball}} | i = 1...5) \\ 0 & \text{if } d_i = \max(d_i^{\text{ball}} | i = 1...5) \\ \dfrac{d_i - d_{\min}}{d_{\max} - d_{\min}} & \text{if } d_{\min} < d_i < d_{\max} \end{cases} \tag{6}$$

Based on the component probabilities mentioned in the above definition and the equation 1. We then construct a method for calculating $\mathcal{Q}_{a_j,i}(k)$ points that correspond to the appropriate actions a_j for robot i at time k. The list of actions is defined as a table I. In this study, the method is built based on experiments on soccer matches and the coach's strategies.

- PlayingAction.Assist (a_2)

$$\mathcal{Q}_{a_2,i}(k) = \mathcal{R}_{a_2}(k) \times [(0.2\mathcal{P}_1(k)) + \mathcal{P}_2(k) + \mathcal{P}_3(k)] \tag{7}$$

- PlayingAction.MovingWithBall (a_3)

$$\mathcal{Q}_{a_3,i}(k) = \mathcal{R}_{a_3}(k) \times [\mathcal{P}_3(k) + \mathcal{P}_4(k) + \mathcal{P}_5(k)] \tag{8}$$

- PlayingAction.TrytoCatchBall (a_4)

$$\mathcal{Q}_{a_4,i}(k) = \mathcal{R}_{a_4}(k) \times \mathcal{P}_5(k) \tag{9}$$

- PlayingAction.TrytoPass (a_5)

$$\mathcal{Q}_{a_5,i}(k) = \mathcal{R}_{a_5}(k) \times [\mathcal{P}_1(k) + \mathcal{P}_3(k) + \mathcal{P}_4(k)] \tag{10}$$

Action		
a_1	`Stopped`	Stop all robot activities
a_2	`Assist`	Move closer to support teammates who have the ball
a_3	`MovingWithBall`	Robot tries to move with the ball
a_4	`TryToCatchBall`	Try to get the ball from the opponent robot
a_5	`TryToPassBall`	Try to pass on to teammates
a_6	`TryToShoot`	Try to kick the ball into a goal positions
a_7	`TryToDribble`	Try to dribble, don't hold the ball for more than 10 seconds
a_8	`Defend`	Robots do not have the ball and try to defend
a_9	`GoalKeeping`	Special action for robot as goalkeeper

TABLE I
LIST OF POSSIBLE ACTIONS (TO BE EXPANDED OR REDUCED)

- PlayingAction.TrytoShoot (a_6)

$$\mathcal{Q}_{a_6,i}(k) = \mathcal{R}_{a_6}(k) \times [\mathcal{P}_1(k) + \mathcal{P}_4(k)] \tag{11}$$

- PlayingAction.TrytoDribble (a_7)

$$\mathcal{Q}_{a_7,i}(k) = \mathcal{R}_{a_7}(k) \times \mathcal{P}_1(k) \tag{12}$$

- PlaygingAction.Defend (a_8)

$$\mathcal{Q}_{a_8,i}(k) = \mathcal{R}_{a_8}(k) \times [\mathcal{P}_2(k) + \mathcal{P}_4(k)] \tag{13}$$

- PlayingAction.GoalKeeping (a_9)

$$\mathcal{Q}_{a_9,i}(k) = 1 \tag{14}$$

Assumption 3: All $\mathcal{Q}_{a_j,i}$ computations for each robot will be synchronous at each iteration k.
The assumption 3 to ensure synchronization in calculating $\mathcal{Q}_{a_j,i}$ points for each robot's action is identical.

Our objective is to let the robots decide their best action according to team rules (such as one player maximum contesting a ball) and according to the other potential robot actions.

Assumption 4: if $p_{1,i}$ is the position of robot i of team 1, $p_{2,j}$ is the position of robot j of team 2, p_b is the position of the ball, we assume that if $p_{i,j} = p_b$, which proves that robot j of the team i has the ball.

By using the approach that gives the highest score based on the probability of success for each specific action, each robot can quickly make decisions based on real time. Details are presented in algorithm 1.

Moreover, this approach does not require high computing power for each robot. It is essential to note that the coefficients in the above formulas have been chosen through simulations on our software and tested with real robots. These coefficients can be adjusted based on the coach's offensive or defensive strategies.

Algorithm 1: Calculating and evaluating the reward points for each action

Input: position of robots, position of ball
while *match is ON* **do**
  determine playing situation: ATTACK, DEFENSE;
  build a list of possible actions for all teammates (PlayingAction) a_j, $j = 1...9$;
  compute $\mathcal{Q}$-Table for all actions of any teammates;
  $\mathcal{Q}_{a_j,i}(k) = \mathcal{R}_{a_j}(k) \times \mathcal{P}_{a_j}(k)$;
  determine best playing action:
  $a^*_{i,j} \leftarrow \max_{a_{i,j}}\{\mathcal{Q}_{i,j} : j = 1\ldots,9; i = 1\ldots,5\}$;
  trigger action ;
end

IV. A PROPOSED DIGITAL TWIN SIMULATOR

A. Proposed a simulator tool based on Digital twin approach

This software is a crucial tool in the simulation and analysis of robotic systems, facilitating rapid algorithm development and strategic coordination testing. Divided into three distinct components, this software exploits the power of high-level control, low-level control, and dynamic modeling to reproduce real-world scenarios (see Figure 2).

- High-level control: The high-level control component is designed to develop algorithms and strategies independent of specific robot hardware configurations. It consists of the main blocks *Perception Manager, Local World Map, Strategy Manager, Trajectory planner, IMU Processor*, and *Kalman Filter*. These algorithms are executed on industrial computers, providing the flexibility to refine and optimize robotic behaviors without being constrained by the underlying hardware. This software feature enables us to focus on the decision-making and intelligence aspects of robot control, helping to increase adaptability and innovation.
- Low-level control: On the other hand, the low-level control part of the software explores the subtleties of robot hardware configurations, such as actuator size and allocations. Custom algorithms are created to correspond to real-world robotic configurations, guaranteeing precise and efficient control of physical systems. These specialized algorithms are deployed on 32-bit processors, adapting perfectly to the practical requirements of robotic systems. This allows us to optimize the software for specific robot designs, maximizing their performance in real-world scenarios.
- Modeling of robot kinematics: The third part of the software focuses on dynamic kinematic modeling of robots. Its main objective is to accelerate the testing and evaluation of coordination strategies between several

robots. This component also exploits real-time data from physical sensors, enabling a high degree of precision and realistic simulation. By simulating the dynamic behaviors of robots and integrating sensor data, it provides a powerful insight into how their algorithms perform in diverse and complex environments, helping to advance the field of collaborative robotics.

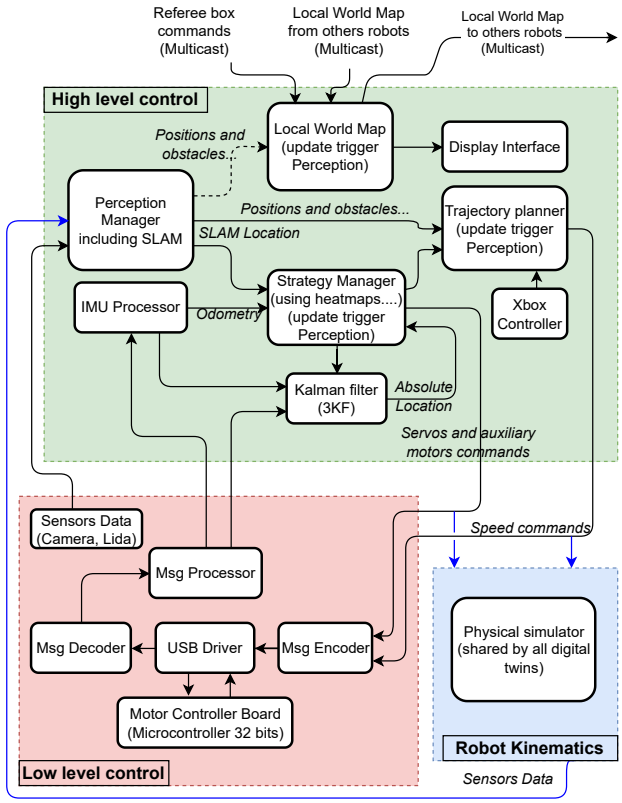

Fig. 2. An architectural design for soccer robot competition simulation software

B. An example of action evaluation on a simulator

To be able to evaluate actions, we used a simulator, which is shown above. The size of the simulation field is $23m \times 15m$. The origin point $O(0,0)$ is chosen to coincide with the center point of the field. The present scenario entails two football teams, distinguishable by the red dot with a white circle; the red dot white circle, and the red border outside (a red team), respectively. In this example, the reward value $\mathcal{R}_{a_i}(k)$ is predetermined. We fixed robot 5 as the goalkeeper.

The goal is to find the best action for a robot of a red team. We then compute $\mathcal{Q}$-Point for each action by using the formulas 7, 8, 9, 10, 11, 12, 13, 14, respectively. Table II shows an example of the $\mathcal{Q}$-Point for each action corresponding to each robot at time k. It can be seen that actions a_{5-4} (Try to Pass Ball to robot 4) of robot 1, and a_2 (Assist) of robot $2, 3, 4$ have respectively the highest scores. This means that these actions are highly selective.

Figure 3 shows an example of the state of the robots at time k. Robot number 1 requires an appropriate action decision.

-	$\mathcal{Q}_{a_j,1}(k)$	$\mathcal{Q}_{a_j,2}(k)$	$\mathcal{Q}_{a_j,3}(k)$	$\mathcal{Q}_{a_j,4}(k)$	$\mathcal{Q}_{a_j,5}(k)$
a_1	0	0	0	0	0
a_2	0	1.6	2.4	3.0	0
a_3	3	0.8	0.6	1.4	0
a_4	3	0	1.6	1	0
a_{5-1}	0	0.2	1.2	1.5	0
a_{5-2}	3	0	0	1.5	0
a_{5-3}	1.5	0.2	1.2	1.5	0
a_{5-4}	4.5	0.2	1.2	0	0
a_{5-5}	0	0	0	0	0
a_6	0	0	2	0	0
a_7	0	0	0	0	0
a_8	0	0.3	1.5	1.5	0
a_9	0	0	0	0	1.0

TABLE II
AN EXAMPLE OF THE Q POINT FOR EVERY ACTION a_j CORRESPONDS TO EACH ROBOT i AT TIME k

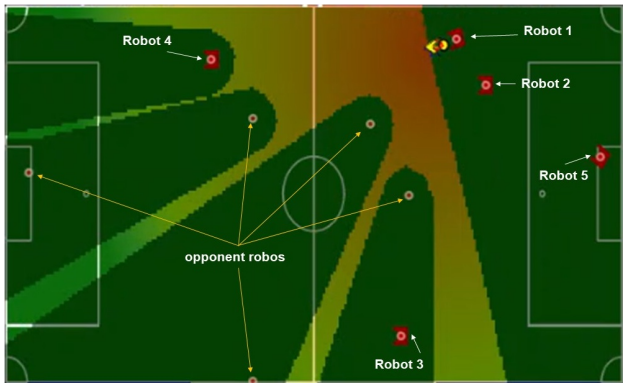

Fig. 3. States of the robots on the field at time k (in the present state, Robot 1 needs to make a decision based on the Q-table's value)

According to the values in table II, figure 4 shows that robot 1 tries to pass a ball to robot 4 at time $k+1$, because robot 4 has the highest probability of scoring a goal.

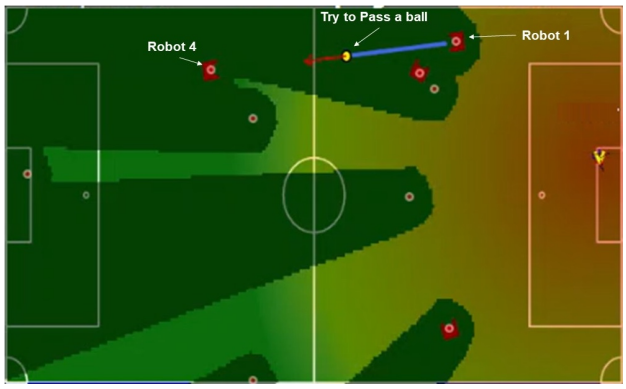

Fig. 4. States of the robots on the field at time $k+1$ (Robot 1 has decided to Pass the ball to robot 4)

A demonstration video can be found at the following link https://youtu.be/YEpfklV8hRg. We have introduced a variety of scenarios in our simulation that require the robot to make decisions using the component probabilities described above.

C. Implement the strategy on real robots

We also implemented these strategies during the 2023 RoboCup competition in Bordeaux, France [14]. These robots have length $\times$ width $\times$ height dimensions of $0.8m \times 0.8m \times 0.8m$. It uses four omnidirectional wheels, enabling a robot to maneuver in any direction. It is also equipped with a Lidar sensor and four cameras to determine its position, other robots' positions, and the ball's position on the pitch. The robots also have an on-board computer i7 - 10810 and a 32-bit microprocessor control board (see Figure 5).

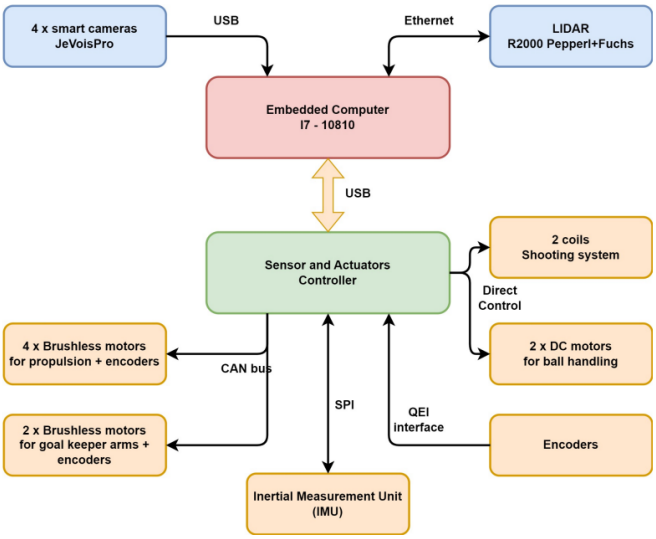

Fig. 5. Hardware architecture of the robot

Figure 6 shows an example of a robot that is required to automatically make the decision to pass the ball to the robot with the highest probability of scoring.

Fig. 6. An example of a robot making automatic action decisions

It is also helpful to note that, although the robots are able to make decisions about passing the ball, the real positioning errors of the robots actually affect their decision-making ability.

V. Conclusion

This study presents a novel real-time strategy for selecting specific actions for a team of soccer robots. The proposed approach involves evaluating and assigning points to the actions of each robot at each moment. The allocation of bonus points considers the robots' current state and the probability of scoring a goal during the calculation process. Thanks to these decentralized approaches, robots can make their own decisions, potentially extending flexibility under complex conditions.

In practical terms, when we tested these strategies on real robots in the RoboCup competition, in some cases, the robots could not decide to proceed because they had identical reward points. The reason is due to the influence of positioning errors of the robot and the ball in the field. Therefore, we aim to improve the effectiveness of reward point methods for decision-making in situations where the robot has limited information on environmental parameters in future research. In addition, it improves the decision-making ability of individual robots in conditions characterized by environmental uncertainty.

Acknowledgment

This work was funded by the French ANR/AID agency under the RoboSCo project and University of Toulon.

References

[1] RoboCup, *RoboCup-97: Robot Soccer World Cup I.* Springer Berlin, Heidelberg, 1997.

[2] K.-H. Park, Y. J. Kim, and J. H. Kim, "Modular q-learning and based multi-agent cooperation for robot soccer," in *Robotics and Autonomous Systems*, 2001.

[3] G. Tesauro, "Extending q-learning to general adaptive multi-agent systems," in *Proceedings of the 16th International Conference on Neural Information Processing Systems*, NIPS'03, (Cambridge, MA, USA), p. 871–878, MIT Press, 2003.

[4] K.-S. Hwang, S.-W. Tan, and C.-C. Chen, "Cooperative strategy based on adaptive q-learning for robot soccer systems," *IEEE Transactions on Fuzzy Systems*, 2004.

[5] E. Duryea, M. Ganger, and W. Hu, "Exploring deep reinforcement learning with multi q-learning," *Intelligent Control and Automation*, vol. 07, no. 04, pp. 129–144, 2016.

[6] C. Kim, Y. Hwang, and J.-H. Kim, "Deep q-network for ai soccer," in *arXiv*, 2022.

[7] T. Haarnoja, B. Moran, G. Lever, S. H. Huang, D. Tirumala, Markus, Wulfmeier, J. Humplik, S. Tunyasuvunakool, N. Y. Siegel, R. Hafner, M. Bloesch, K. Hartikainen, A. Byravan, L. Hasenclever, Y. Tassa, F. Sadeghi, N. Batchelor, F. Casarini, S. Saliceti, C. Game, N. Sreendra, K. Patel, M. Gwira, A. Huber, N. Hurley, F. Nori, R. Hadsell, and N. Heess, "Learning agile and soccer skills and for a bipedal and robot with deep and reinforcement learning," in *arXiv*, 2023.

[8] K. Yasui, K. Kobayashi, K. Murakami, and T. Naruse, "Analyzing and learning an opponent's strategies in the robocup small size league," in *RoboCup 2013: Robot World Cup XVII* (S. Behnke, M. Veloso, A. Visser, and R. Xiong, eds.), (Berlin, Heidelberg), pp. 159–170, Springer Berlin Heidelberg, 2014.

[9] A. J. R. Neves, F. Amaral, R. Dias, J. Silva, and N. Lau, "A new approach for dynamic strategic positioning in robocup middle-size league," in *Progress in Artificial Intelligence* (F. Pereira, P. Machado, E. Costa, and A. Cardoso, eds.), (Cham), pp. 433–444, Springer International Publishing, 2015.

[10] F. A. Oliehoek and C. Amato, *A Concise and Introduction to and Decentralized POMDPs.* Springer, 2015.

[11] Busoniu, Babuska, and D. Schutter, "Multi-agent reinforcement learning: An overview," *Chapter 7 in Innovations in Multi-Agent Systems and Applications – 1, vol. 310 of Studies in Computational Intelligence, Berlin, Germany Springer*, no. 10-003, 2010.

[12] R. Lowe, Y. Wu, A. Tamar, M. University, U. Berkeley, and U. Berkeley, "Multi-agent actor-critic for mixed cooperative-competitive environments," *arXiv*, 2020.

[13] S. Tiomkin and N. Tishby, "A unified bellman equation for causal information and value in markov decision processes," *arXiv*, 2018.

[14] RoboCup, "Robocup middle size league competition," *Bordeaux, France*, 2023.