Hindawi
Wireless Communications and Mobile Computing
Volume 2023, Article ID 9830654, 1 page
https://doi.org/10.1155/2023/9830654

Retraction

Retracted: Cooperation Mode of Soccer Robot Game Based on Improved SARSA Algorithm

Wireless Communications and Mobile Computing

Received 19 September 2023; Accepted 19 September 2023; Published 20 September 2023

This article has been retracted by Hindawi following an investigation undertaken by the publisher [1]. This investigation has uncovered evidence of one or more of the following indicators of systematic manipulation of the publication process:

(1) Discrepancies in scope
(2) Discrepancies in the description of the research reported
(3) Discrepancies between the availability of data and the research described
(4) Inappropriate citations
(5) Incoherent, meaningless and/or irrelevant content included in the article
(6) Peer-review manipulation

The presence of these indicators undermines our confidence in the integrity of the article's content and we cannot, therefore, vouch for its reliability. Please note that this notice is intended solely to alert readers that the content of this article is unreliable. We have not investigated whether authors were aware of or involved in the systematic manipulation of the publication process.

Wiley and Hindawi regrets that the usual quality checks did not identify these issues before publication and have since put additional measures in place to safeguard research integrity.

We wish to credit our own Research Integrity and Research Publishing teams and anonymous and named external researchers and research integrity experts for contributing to this investigation.

The corresponding author, as the representative of all authors, has been given the opportunity to register their agreement or disagreement to this retraction. We have kept a record of any response received.

References

[1] W. Zhan and S. Qu, "Cooperation Mode of Soccer Robot Game Based on Improved SARSA Algorithm," *Wireless Communications and Mobile Computing*, vol. 2022, Article ID 9190687, 11 pages, 2022.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 9190687, 11 pages
https://doi.org/10.1155/2022/9190687

Research Article

Cooperation Mode of Soccer Robot Game Based on Improved SARSA Algorithm

Wei Zhan and Shengqing Qu

International College of Football, Tongji University, Shanghai 200092, China

Correspondence should be addressed to Wei Zhan; 93762@tongji.edu.cn

Received 25 February 2022; Revised 8 March 2022; Accepted 12 March 2022; Published 1 April 2022

Academic Editor: Kalidoss Rajakani

Football is one of the most popular sports in the world, and its competition has received increasing people's attention. With the increasing number of robot football competitions, more strategic planning is needed for robot football matches. In a tournament, each player has their own task and must have the skills to complete it. In this paper, we use the RoboCup2D platform to give details of the server and client roles, introduce the agent model in RoboCup2D, and compare the plan design and scheme design presented in the current study using the SARSA algorithm, one of the augmented methods classified as TD learning metrics. In addition, heuristic information was introduced and implemented to enhance learning through the sharing of Q values between participants and reinforcement learning. A comparative analysis of the feasibility of the SARSA algorithm in the context of its application in RoboCup2D was carried out, and the experimental results proved that our algorithm was effective in improving the team's offensive and defensive capabilities.

1. Introduction

The discipline of artificial intelligence is dedicated to the study of how machines can be used to simulate humans, that is, the comprehensive and analytical study of computational intelligence (agents) that can generate intelligent behaviour [1]. Over the past 50 years, research in AI has been closely integrated with application areas. Artificial intelligence has been widely used in areas such as expert systems that simulate the problem-solving thinking of human experts, machine learning that studies the mechanisms of human learning and the way the human brain thinks, pattern recognition that concentrates on graphic recognition and speech recognition, and artificial neural networks that use neuronal structures to simulate the workings of the human brain's nervous system [2, 3].

The aim of machine learning is to investigate how computers can simulate or fully implement human learning behaviour [3]. Reinforcement learning (RL), also known as augmented learning or reactive learning [4], is characterized by an intelligent learning without the need for teacher signals, based on environmental and reward signals in the expectation of maximizing the expected benefit.

In this process, the intelligence changes the next state of the environment based on its behaviour in the current state while receiving certain reward signals. Intelligence learns through fully autonomous interaction with the environment and gradually improves during this iterative trial-and-error interaction. The ultimate goal is to maximise the long-term future reward for the actions of the agent, which learns to form a strategy that maximises the benefits, i.e., the optimal strategy.

The Robot World Cup [5] was originally established to promote the development and research in the field of artificial intelligence and robotics, and as more and more scholars have delved deeper, RoboCup competitions have become platforms for research such as distributed artificial intelligence and multiple intelligences. In particular, in the RoboCup2D project, intelligences simulate real football players in a standard computer environment to build a 2D simulation of a football robot for a real-time match, with various participants and research scholars applying methods such as

multi-intelligence-related techniques and machine learning to the development and learning of the team [6]. In the RoboCup2D platform, players are intelligence with independent thinking brains that can make actions and collaborative decisions in a real-time, asynchronous, noisy confrontation environment. Researchers can focus on multi-intelligence collaboration, real-time decision-making, formation, and other functions, and research methods can use machine learning, data mining, neural networks, and other relevant technologies, thus providing an important impetus to the development of artificial intelligence and research significance [7].

The CK, FK, and PK in football are the most important in the game. According to the World Cup "OPTA" held in 2018, there were 169 goals in the tournament with 73 goals and a total gain of about 43%, and every team has designed and studied some defensive knowledge in the tournament. In RoboCup, there was a lot of research on turning paths or spending a lot of points on attacks, but Paul researched taking and protecting targets not as much as attacks. In addition, the scenario has even less research studies in the tournament, so it is an area worth studying [8].

2. Related Work

The original idea of robotic football was formally proposed in [9] and saw the first proposals and request. In [10], create the international RoboCup competition was first proposed and called for [9]. At the same time, scholars and researchers from all over the world began to participate in the RoboCup research room, using robotic football as a research topic. For example, An et al. [11] from the Electronics Technology Laboratory in Japan focus on multi-intelligence collaborative systems in robotic football. Haobin et al. [12] from Carnegie Mellon University in the USA have carried out work on the collaboration of intelligence using reinforcement learning, neural networks, and other methods.

Reference [13] proposed a hierarchical approach to learning strategies using BP neural networks to train players to intercept the ball in different scenarios. Training results of the interceptions were then used to learn pass success judgments and construct decision trees for player actions, decisions, and collaboration and finally to learn pass objects online [14]. Reference [15] used predictive memory to update the world model of the intelligence to achieve more flexible role changes for dynamic formations and fixed response strategies for set pieces.

Reference [16] has developed a mathematical model of the underlying actions of an agent and trained the decision-making system of the agent using a neural network approach. Reference [17] used reinforcement learning to improve the overall capabilities of players and teams [13] by giving the team a long-term goal of "winning the game," under which the players learn autonomously, selecting appropriate actions during the learning process and ultimately obtaining optimized team decisions. The HELIOS team at Fukuoka University in Japan conducted an in-depth study on 2D formation design and passing strategy [18] searches

The term reinforcement learning was first introduced by Minsky in 1954 in his Ph.D. thesis [19], and in his dissertation, he addressed issues related to the application domain of reinforcement learning. The Q learning algorithm was proposed in [20], where Q learning allowed reinforcement learning to find optimal action strategies without relying on a problem model. Reference [21] proposed a confidence ceiling tree algorithm to add reinforcement learning to Go applications. Reference [22] proposed a feedback control adaptive dynamic planning algorithm, and [23] proposed a deterministic policy gradient algorithm.

3. Background Knowledge

3.1. Reinforcement Learning. The reinforcement learning process is simply the process by which an agent learns to select the optimal action to reach its goal. The specific process is that the intelligence selects an action while perceiving the current state of the environment, at which point the environmental state migrates to a new state; accordingly, the new state generates a reinforcement signal, and the intelligence selects the next action based on the current environmental information and the reinforcement signal. The basic model is shown in Figure 1.

In the reinforcement learning process, the intelligence can continuously try to choose an action. This process is also known as trial-and-error learning, where the action is evaluated by the reinforcement signal (often also called the reward and punishment value) provided by the environment, and the intelligence learns by relying only on its own experience during the learning process, adjusting the evaluation value of the action through the reward and punishment value, so that the intelligence can eventually obtain the optimal strategy [24].

Intelligence must explore the environment by executing the action and perceiving the outcome of the action in terms of the impact on the environment and the reward obtained. The only feedback the agent receives is the reward, but the agent does not receive any information about the correct action. At some point in time, the intelligence has a strategy with a specific performance. In order to improve the strategy, the intelligence must try various actions and check the results of these actions. Because some actions may be worse than the current strategy, but without trying, the intelligence will never find an improved strategy. Furthermore, because the environment is dynamically changing, intelligence must constantly explore to keep up-to-date with its strategies. With the limited feedback signals provided by the environment, the intelligence must work hard to evaluate and improve the action.

3.2. RoboCup2D Platform. RoboCup2D tournaments are similar to human football tournaments and are a tool that enables learning and research for multiagent systems and related technologies. RoboCup2D is conducted in a standard computer environment and allows for the complete study of intelligence in high-level decision-making without regard to hardware issues. The RoboCup2D platform allows for the testing of various theories, algorithms, and player

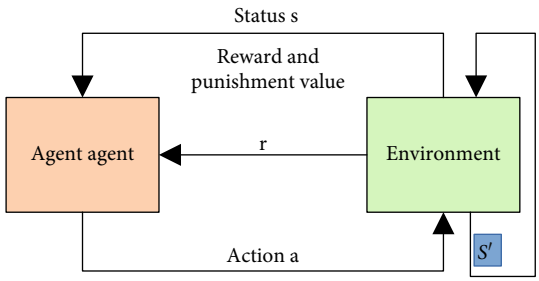


Figure 1: Basic model of reinforcement learning.

intelligentsia architecture in a real-time asynchronous, noisy adversarial environment to study cooperative multi-intelligence adversarial problems [25]. The RoboCup2D platform architecture is shown in Figure 2, and the competition uses a client/server architecture, with the server and client communicating via the UDP/IP protocol

Developed and used by the RoboCup committee. Soccer Server is divided into two parts: Soccer Server and Soccer Monitor. Soccer Server provides a virtual pitch to simulate football and player movement, position changes, etc. During a match, Soccer Server calculates and updates the position, movement, and status of all objects on the pitch; sends messages to players; and accepts and executes player commands [26]. The Soccer Monitor is a visualisation tool that allows us to observe the game and record information about the score, team names, players, and ball positions.

The client is written by the participating teams, and each client is the brain of the player, simulating the player's mind and directing the player's movement. The client sends commands to control the corresponding player and simultaneously receives information back from the server [27]. Each client is a separate process, which can only control one player individually, so a team of 11 players and one coach requires 12 separate processes to run.

3.3. Agent Model. According to the function of the server, agent models in RoboCup2D can be divided into three categories: agent perception models, motion models, and movement models. Among them, agent perception models are subdivided into auditory perception models, visual perception models, and body perception models, which are modelled on the three human senses of hearing, vision, and touch.

3.3.1. Agent Perception Model. Intelligence can receive three different types of perceptual information from the server: auditory, visual, and body perception information. Auditory perception is modelled on the human ear, which can hear messages from other intelligence. Visual perception is similar to the eye and can observe information about objects within a certain range. Body perception detects the player's own current state. All three types of perception models are modelled on the perception systems of real football players and provide the intelligence with various types of information on the field.

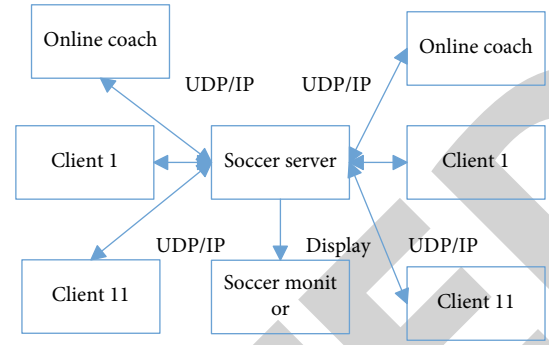


Figure 2: RoboCup2D platform architecture.

3.3.2. Agent Movement Model. In the game, the coordinate system is divided into a stadium global coordinate system and a player relative coordinate system [3]. The stadium global coordinate system, using a plane right angle coordinate system, faces the enemy goal as the positive direction of the x-axis and the x-axis turns clockwise 090 as the positive direction of the y-axis. The player relative coordinate system, using a planar polar coordinate system, uses the player himself as the pole, with the player facing the polar axis and the clockwise direction as the positive direction of the angle, which ranges from -1800 to 1800. The movement of the object in each simulation cycle is shown in the following equation:

Acceleration

$$\left(u_x^{t+1}, u_y^{t+1}\right) = \left(v_x^t, v_y^t\right) + \left(a_x^t, a_y^t\right). \tag{1}$$

Mobile

$$\left(p_x^{t+1}, p_y^{t+1}\right) = \left(p_x^t, p_y^t\right) + \left(u_x^{t+1}, u_y^{t+1}\right). \tag{2}$$

Decay rate

$$\left(v_x^{t+1}, v_y^{t+1}\right) = \text{decay} \times \left(u_x^{t+1}, u_y^{t+1}\right). \tag{3}$$

Reset acceleration

$$\left(a_x^{t+1}, a_y^{t+1}\right) = (0, 0), \tag{4}$$

where (v_x^t, v_y^t), (a_x^t, a_y^t), and (p_x^t, p_y^t) represent the coordinate position, acceleration, and velocity of the object at time t. Decay is a parameter that affects the decay of the object's velocity and is controlled by ball_decay and player_decay.

The player's movement model simulates the movement of a football player in a real environment; for example, the amount of wind can directly hinder the player's acceleration and speed, and the intensity of sunlight can interfere with the player's field of vision. In the simulation of a football match, only one environmental disturbance is added, namely, wind interference. In the motion model of the agent, the server adds wind interference directly to the acceleration equation, which is expressed as follows:

$$\left(u_x^{t+1}, u_y^{t+1}\right) = \left(v_x^t, v_y^t\right) + \left(a_x^t, a_y^t\right) + (\tilde{r}_{r\ \max}, \tilde{r}_{r\ \max}), \quad (5)$$

where $\tilde{r}_{r\ \max} \in [-r\ \max, r\ \max]$ is a random number. The value of r maxis proportional to the player's current speed.

3.3.3. Agent Movement Model. The intelligence in RoboCup2D competitions is all two-dimensional, there is no concept of height, the intelligence is represented by circles, and therefore, the action model does not include the movements of the feet.

The player's basic action model is divided into mutually exclusive and compatible actions, depending on whether they can be performed simultaneously in the same cycle. Mutually exclusive actions, also known as active actions, are actions that can be sent to the server in the same cycle and include six actions: dash, kick, tackle, turn, catch, and move. Compatible actions, also known as collateral actions, are multiple actions that can be selected in the same simulation cycle, including chronic, conceive, attention to, say, and score. In theory, any number of compatible actions can be sent in the same cycle, but in practice, multiple compatible actions need to be carefully chosen to avoid blocking the server's communication.

3.4. RoboCup2D Subtasks. RoboCup2D competitions, on the one hand, model the intelligence in real football matches as realistically as possible and, on the other hand, make RoboCup2D problems more difficult to learn. Trying to solve RoboCup2D problems as a whole in one go becomes increasingly difficult, so various experts have proposed RoboCup2D-based subtasks. In RoboCup2D subtasks, only certain problems are trained for players, e.g., ball control training for intelligence in the Keepaway subtask for multi-intelligence collaboration.

The Keepaway subtask is played in a certain size area where two teams engage in ball possession-stealing confrontation type of training [7]. In this task, the ball-handling player tries to keep possession of the ball for as long as possible in a limited area, while the ball-stealing player of the other team tries to gain possession. In the Keepaway subtask platform, the area and the number of players on both sides are arbitrarily set, as showed in Figure 3 for a classic 20 ∗ 20 3v2 Keepaway platform, consisting of 3 ball handlers and 2 ball stealers. The square area is the field, the white hollow circles are the balls, the yellow players are the ball handlers, and the blue players are the ball stealers.

In each learning task, the ball carrier needs to keep possession of the ball for a long time; the ball carrier needs to hold the ball alone and also needs to learn to pass the ball or follow a suitable passing route with the ball, while the goal of the ball carrier needs to steal the ball to get possession or force the ball carrier to kick the ball out of bounds.

4. Reinforcement Learning Methods

4.1. SARSA. Here, we describe the algorithm used in this paper, the SARSA method [3]. SARSA is a reinforcement learning method classified as TD learning on measures, which stands for State, Action, Reward, State (next), and

Figure 3: Keepaway platform.

Action (next). The effectiveness of taking $Q(s, a)$ an at states is assessed by using the action value function $Q(s, a)$. Also, $Q(s, a)$ used here is called the Q value and is used as a basis for comparing the merits of choosing $Q(s, a)$ under state s.

At the transition of an agent in state s_t at time t to state s through action a, Q-learning updates the Q value of the following formula:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha\left[r + \gamma \max_{a' \in A(s')} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right]. \quad (6)$$

4.2. On the Method of Action Selection. In the above-mentioned reinforcement learning algorithms, the action selection method itself needs to be specified when selecting actions. In SARSA, we often use the e-greedy method for reinforcement learning. The e-greedy method is an action selection method that randomly selects an action with probability e and chooses the action with maximum Q value in the state with other probabilities. The e-greedy method is an action selection method that randomly selects an action with probability e and chooses the action with maximum Q value in state t at time t state s and selects the action with the maximum Q value in state s. If the action selection method selects only the action a with the maximum action value function max $Q(s-,a)$ in state s at time t, it cannot learn the payoffs obtained from the other actions. Therefore, the method of random selection of actions with probability c is generally used. In addition, e is set to $0 < c \leq 1$. There are many cases of fixed c and cases where e decreases as learning proceeds, and the method of setting varies depending on the learning conditions.

4.3. Heuristic Accelerated SARSA (HAQL). Reference [11] investigated a method called Heuristic Accelerated SARSA (HAQL), which is based on SARSA and applies heuristics. As learning progresses and Q increases, the impact of Q becomes greater than the impact of domain knowledge. To achieve these goals, the method imposes a new condition on the e-greedy method. e-greedy is an action selection method that randomly selects an action with probability e and chooses the action with the maximum Q value over states with other probabilities. $H(s_t, a)$ is the maximum of $Q(s, a) + sH(s_t, a_t)$, where s is the state at time t, a is the

possible action, and k is the effect of the heuristic. $H(s_t, a)$ is obtained by the following formula, where the action a is considered the better action in state s_t. For other actions, $H(s-,a) = 0$.

$$H_t(s_t, a_t) = \max_a Q(s_t, a) - Q(s_t, a_t) + \eta. \quad (7)$$

In this context, action a is the action with the maximum value of Q among the actions considered in a given state s, which was considered to be $n = 0.01$ in the study by Haobin et al. [12].

Thus, by using heuristics in the action selection method, learning is faster if the developer's domain knowledge is correct and the conditions are applied properly and slower if the conditions are applied improperly.

5. Learning Programme

5.1. Realisation Process. Before describing the reinforcement learning program in this paper, we would like to mention the structure of the program in this study, which is based on SARSA. In this program, the procedure is based on SARSA. The updating of the Q value is done by the intelligence's own program, since the intelligence decides the reward from the information it gets. In RoboCup, the agent does not get the full state of the field because it contains errors in recognising distant objects, and if it cannot see the ball, it cannot know its position. Therefore, in order to keep the time and reward values constant for the whole team, the agent itself temporarily stores the state, actions, and rewards obtained for each time and updates the Q value at the end of the episode. The Q value is stored as data in an array of 1296 state representations and 4 action sets, as described below. In addition, as described below, the program selects actions based on predetermined macro actions (e.g., moving towards the ball). If there is a change in the observed state, the program selects an action via the c-greedy method, and if there is no change in the observed state, the program selects an action that is one time ahead. The algorithm implemented in this program is shown below.

(1) Initialize all Q value

(2) Observe state s and save the state

(3) Select $Q(s, a)$ in the e-greedy method if the state is different from the previous one, or select the same action as the previous one if the state is the same as the previous one. If the state is the same as the previous one, the same action as the previous one is selected (the initial action is selected as active a: the e-greedy method)

(4) Perform action a, accept reward r, and store the action and reward at that time

(5) Repeat (2) to (4), terminating the iteration when the episode (trial) ends

(6) Use equation (1) to update and store Q values, storing data for state s_t, action, and reward r each time t

(7) Repeat (2) to (5) until the end of learning, terminating the iteration when the end-of-learning condition is reached

5.2. Status Representation. In the guarding task described above, state is represented by the distance between each player, the distance between other players, the distance between friends and enemies, and the angle between enemies, and path routes at each step. In this study, we used the following distances to represent state: distance between the goal and the ball; distance between the finish and yourself; distance between the ball and yourself; and distance between the marked enemy and yourself.

Each of these is divided into 6 states, each of which is further divided into $64 = 1296$ states. Furthermore, as described in Section 2, in RoboCup, the states are determined by the perceptual information obtained by the agent itself by shaking its head. In this program, the states are determined by the information obtained by the intelligence itself, and there is a situation where the coordinates of the poles and the enemy are not available. In this procedure, there is a situation where the agent cannot know the coordinates of the pole or the enemy due to the distance between the goal and the ball, between the goal and itself, and between the ball and itself. Therefore, we divide the distance between the marked enemy and ourselves into 6 states and the distances of the other state variables into 5 states, plus the states where the coordinates of the object cannot be known, expressing 6 states in one state variable.

5.3. Reward Design. Reward design is very important because it provides the machine with material to learn from. In this paper, at the end of the match, we give a +50 bonus when the team kicks the ball and a -15 bonus when the opponent scores a goal. To balance the rewards, we designed the rewards for successful episodes to be heavier than the rewards for unsuccessful episodes.

5.4. Skill Improvement. In this study, two models were implemented. The first model was a team that changed only the reward design from the previous study and programmed it using SARSA. The second model is where a team programs the reward design from the first model and adds the following two features. We will compare the effects of the reward design in the first model, while examining the effects of other factors on learning in the second model. In this section, we will describe the features of the team project for the second model.

The first feature is that we divide the intelligence into regions based on their initial positions and share Q values between the regions so that the intelligence can learn not only their own learning but also the learning of other intelligence. In football, each position has a different responsibility. For example, the FW player closest to the opponent's goal may stay up front rather than defending, depending on the conditions, and the priority action varies from region to region, but the priority action is broadly determined. In front of his own goal, the best course of action might be to abandon his marker and move to block the shot, while on

the touchline, he might prioritise his marker and try not to chase too many players, even if the sticks are close. Therefore, we share the Q of each area not by position but by the initial position to improve learning.

The second feature is the introduction of heuristics into reinforcement learning. By introducing heuristics, we can expect to speed up reinforcement learning if the best conditions can be proposed. In the case of cooperative behaviour such as football, if we randomly select behaviour in the early stages of learning, the team will not function as a team and learning will suffer. If, on the other hand, we specify conditions that include knowledge of football, learning can proceed smoothly. The structure of the program with shared Q values, mentioned in the first objective, is also expected to improve the speed of learning.

6. Simulation

6.1. Implementation Results. For the purpose of this study, teams using the SARSA algorithm in their reward design are referred to as SARSA teams, while teams using the SARSA algorithm and the suggested rewarding design are referred to as teams. The team using the SARSA algorithm and the suggested rewarding design is referred to as team suggestion A, while the team using the heuristic to share Q values is referred to as team suggestion B, as a comparison considering factors other than reward. Defensive time starts when a corner kick is taken and each intelligence is assigned the number of the enemy to be marked in advance, so that the intelligence starts with the marked enemy in its initial state. In team proposal B, the Q is updated for each area and the Q is shared between the six side players closest to the corner arc, with the other four field players being the central players. The initial arrangement is shown in Figure 4, where the blue team is the implemented team and the yellow team is HELIOS 2016.

As a condition of implementation, all teams have the same state representation and action set. Action sets are assumed to be identical. We also assume that the episode ends when the ball goes out of play (e.g., optics, kick-in), when play is interrupted by an opponent of foul, when one of the team's intelligence kicks the bar, or when the opponent scores a goal. In the case of team SARSA, the decision is made to compare whether a goal was scored, the time it took to score, and the number of time the defence took more than 100 steps. In the case of team SARSA, the learning rate $a = 1.25$ is the same as in the case of [12]. Also, in team proposal B, the parameters of the heuristic were set to =0.5 and $n = 0.01$, and the results of the implementation were compared.

The results of the implementation are shown in Figure 5. In Figure 5, we show the number of successful defences per 100 events. As mentioned above, an episode ends when the ball goes out of play, when play is interrupted, when our team's intelligences kick the ball, or when the opponent scores a goal, and the episode is considered successfully defended if the opponent does not score. The vertical axis is the number of successful defences per 100 events, and the horizontal axis is the number of events.

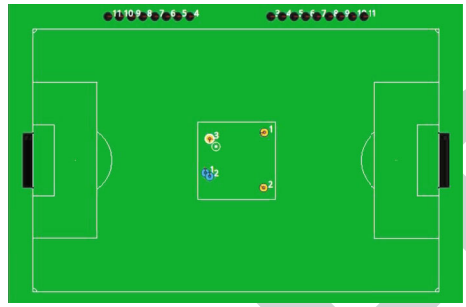


Figure 4: Initial configuration of the agent.

Next, the graph shows the defensive time in the case of a defensive failure. Of course, in football, it is not the score that matters most, but in this paper, it is used as an indicator for comparison. In Figure 6, we show the number of times the defence took more than 100 steps in every 100 episodes of defensive failure. The vertical axis is the number of failed defensive attempts in which the defence took more than 100 steps per 100 episodes, and the horizontal axis is the number of episodes that have passed.

In Figure 7, we show the average time spent defending in each of the 20 episodes. The vertical axis is the average time spent unsuccessfully defending per 20 episodes when a goal was scored, and the horizontal axis is the number of episodes elapsed. Here, we only use the times excluding successful defending.

6.2. Team Match Analysis

6.2.1. Offensive and Defensive Collaboration. The SARSA algorithm was applied to the collaborative learning of intelligence for the Apollo team, and Figures 8(a) and 8(b) show the live matches played by the learned Apollo against the prelearning Apollo and HfutEngine, respectively. Table 1 shows the statistics of 50 matches played by Apollo with prelearning Apollo and HfutEngine, respectively, after adding the algorithm SARSA algorithm learning, 10 minutes a match, for a total of 500 minutes. The Apollo_base team is the original team, and Apollo is the team that joined the SARSA algorithm for reinforcement learning. Experiments prove that the SARSA algorithm improves the players' offensive and defensive collaboration.

The experimental data show that Apollo with the SARSA algorithm added to the learning has a 98% win rate compared to the original Apollo_base team, and Apollo with reinforcement learning training has a 92% win rate compared to the HfutEngine team. Apollo with the addition of the SARSA algorithm effectively improves team collaboration by training on various moves and selecting high-yielding moves at move decision time to converge towards an optimal strategy as quickly as possible. It has been experimentally demonstrated that reinforcement learning based on the SARSA algorithm can effectively improve the offensive and defensive collaboration of an agent.

Figure 9 shows a single Apollo_base vs. Apollo real-time match. The blue player is Apollo with the SARSA

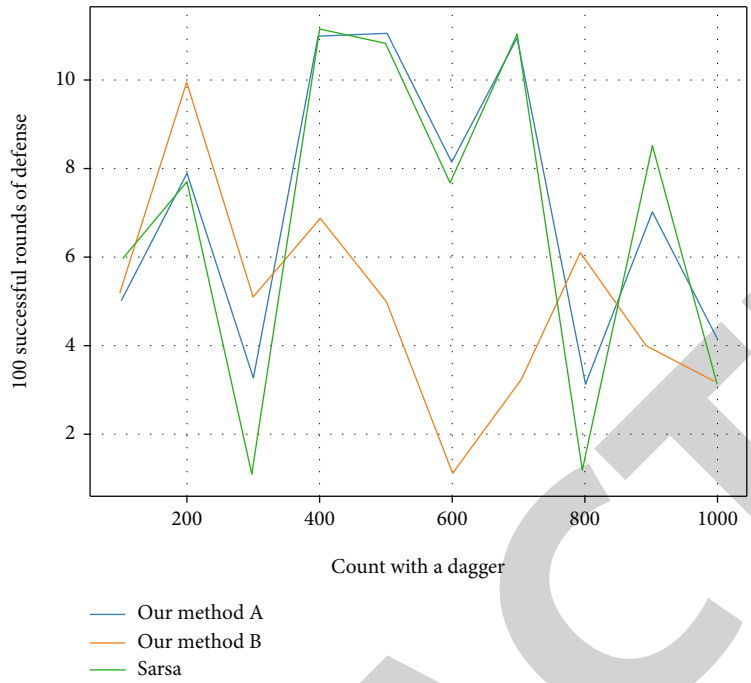


Figure 5: Number of defensive successes.

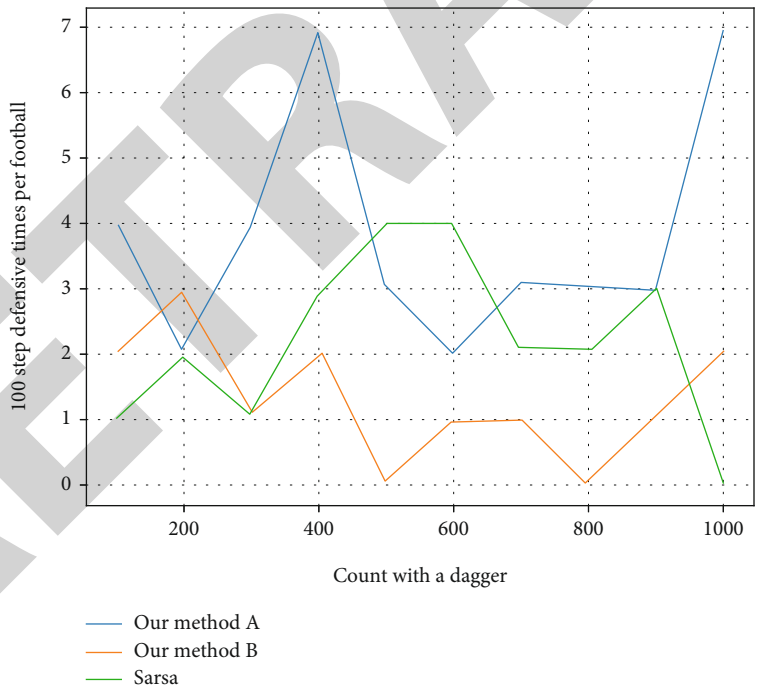


Figure 6: 100 number of guards above 100 steps per episode.

reinforcement learning algorithm. The yellow player is the original Apollo_base team, and the green player is the Apollo_base goalkeeper.

In the single game between Apollo_base and Apollo, Figures 9(a)–9(c) show a continuous scene before the player learns to stabilise, and Figure 9(d) shows an attacking scene after the player learns to stabilise. In Figure 9(a), our player 10 is attacking with the ball, with players 6-9 and 11 as assistants, with player 11 being the nearest assistant; in Figure 9(b), player 10 chooses to pass the ball to 11 as he approaches the opponent's goal with the ball; in Figure 9(c), it shows a successful shovel by the opponent's player 7. In this scenario, player 10 chooses to pass the ball to the nearest assisting player, number 11, when there is

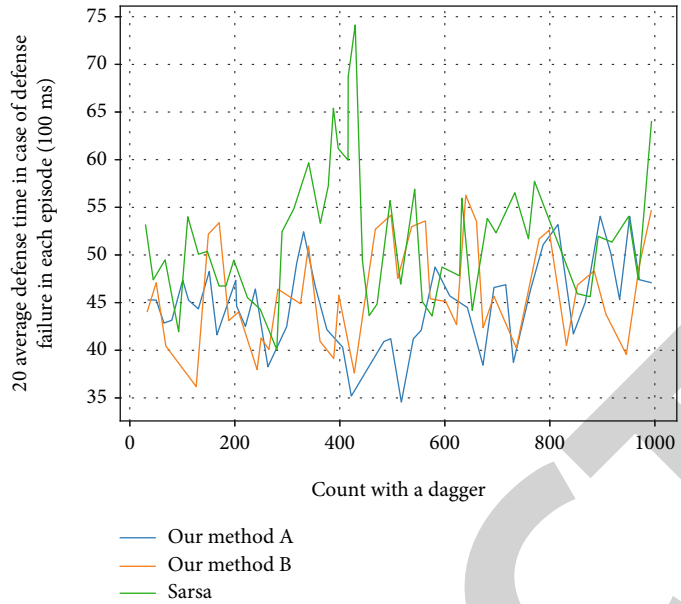
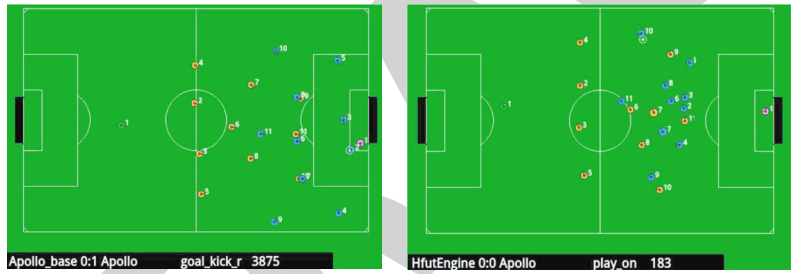


Figure 7: 20 average times spent defending per episode when defending failed.



(a) Apollo_base vs. Apollo (b) HfutEngine vs. Apollo

Figure 8: Live game.

no suitable opportunity to shoot, but as the player is still in the learning stage, the pass is not very successful and he eventually loses possession of the ball. In Figure 9(d), our player 11 holds the ball. Comparing and analysing the offensive scenarios before and after learning to stabilise, it can be concluded that the players choose the appropriate action according to their current state and reward and punishment values after learning to stabilise by the SARSA algorithm, the passing routes are more reasonable, the passing success rate has improved, and the offensive collaboration efficiency is also improved.

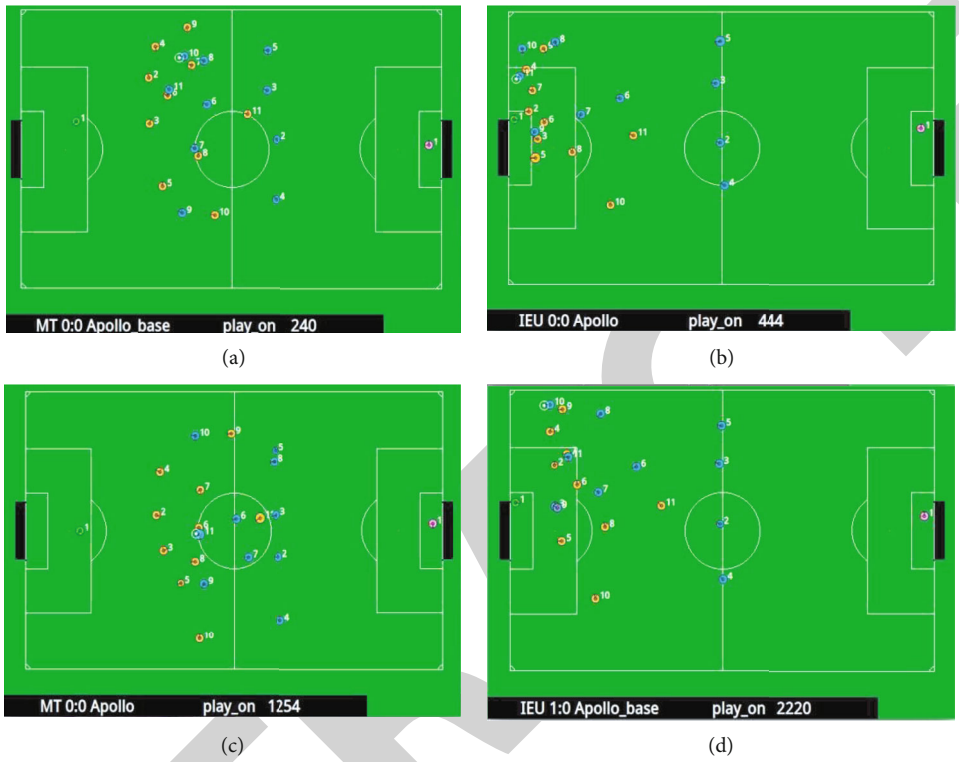
6.2.2. Offensive Collaboration. For the problem of reinforcement learning intelligence, the goal of learning with the SARSA algorithm is to have teams of player intelligence start with a random selection of collaborative strategies and gradually acquire a stable collaborative strategy through learning. As learning takes place in a real competition, the choice of the opponent affects the efficiency of learning. In order to solve the reinforcement learning problem under different agent collaboration strategies, different scenario training is required. Therefore, in this experiment, agents' offensive collaborative strategy training selects teams with stronger offensive capabilities, while defensive collaborative strategy training selects teams with stronger defensive capabilities, so that agents' learning is fully reflected in different collaborations, thus improving the efficiency of both offensive and defensive.

In the experiment, we modified the game script to set the game to start with our team holding the ball and the opponent holding the ball, i.e., setting the teams to be in an offensive state and a defensive state, respectively. Figure 10 shows Apollo's match with each team before and after adding the SARSA algorithm to learn when we have the ball. Table 2 shows the 50 games played by Apollo against each team before and after the SARSA algorithm was added.

The data in Table 2 above shows that the reinforcement learning approach using the SARSA algorithm improved the team's offensive efficiency. Apollo team's pass success rates against MT, Alice, IEU, and NanQiang improved by 9%, 8%, 3%, and 4%, respectively, compared to the original team Apollo_base. The Apollo team scored more goals and conceded fewer goals than the original team during the match. The Apollo team scored more goals and conceded fewer goals than the original team, increasing their winning

Table 1: Apollo compared to the teams.

Team	Number of goals	Number of successful goals	Goal success rate (%)	Win : draw : lose	Total score
Apollo_base vs. Apollo	434	371	85	1 : 1 : 48	23 : 155
HfutEngine vs. Apollo	425	353	83	2 : 2 : 46	25 : 161



(a) (b) (c) (d)

Figure 9: Apollo_base vs. Apollo race live.

Figure 10: Attacking form match live.

percentage. The experiments show that the SARSA algorithm can enhance the team's ability to collaborate when they are in an attacking situation, ultimately improving the players' decision-making ability in attacking collaboration [28].

6.2.3. Defensive Collaboration. Apollo played against each team before and after the SARSA algorithm was added in the opponent holding mode as shown in Figure 11. Table 3 shows the statistics of the 50 games played by Apollo against each team before and after the SARSA algorithm was added.

The data in Table 3 shows that the algorithm-based reinforcement learning method can effectively improve the team's defensive collaboration when there is a certain gap in strength with the opponent, but the learning effect is not obvious when the gap in strength with the opponent is too large. Two teams, Yushan and Miracle, were the winner and runner-up of the RoboCup2D China tournament, respectively, and it can be seen that in the Apollo learning, in the games against Yushan and Miracle before and after Apollo learning, the interception success rate was basically unchanged and there was no significant difference between

Table 2: Comparison of offensive state Apollo learning before and after with each team.

	Apollo_base			Apollo		
	Pass success rate (%)	Number of victories	Total score	Pass success rate (%)	Number of victories	Total score
MT	61	12	71 : 121	70	21	89 : 113
Alice	63	14	82 : 141	71	25	112 : 109
IEU	69	23	91 : 103	72	27	121 : 110
NanQiang	71	24	95 : 104	75	31	133 : 101

Figure 11: Defensive state game live.

Table 3: Comparison of defensive state Apollo learning before and after with each team.

	Apollo_base			Apollo		
	Interception success rate (%)	Number of victories	Total score	Interception success rate (%)	Number of victories	Total score
Yushan	26	0	2 : 331	27	0	3 : 337
Miracle	33	2	5 : 294	32	3	6 : 289
NewForces	31	14	91 : 126	46	24	101 : 104

the number of goals scored and the total score. When playing against NewForces, Apollo's interception success rate increased from 31% to 46% after learning, a 14% increase, and the number of goals scored and the winning percentage also increased significantly. This is due to the fact that the intelligences were given few, if any, learning opportunities during the learning process, when the opponent chosen was stronger, and the number of learning opportunities was reduced accordingly; when the opponent team chosen was not very different, the intelligences were given sufficient learning, thus improving the efficiency of defensive collaboration.

7. Conclusions

In this paper, we use SARSA for defensive behaviour of sets in RoboCup2D simulation and implement the reward design from previous research, omitting the penalty for being kicked by the opponent. As a result of the implementation, we found that allowing the ball to be touched more often by the opponent did not have a direct effect on corner defending. To further improve defensive behaviour, we need to be able to give simple rewards that are indicators of good team-wide functioning.

Data Availability

The dataset used in this paper are available from the corresponding author upon request.

Conflicts of Interest

The authors declared that they have no conflicts of interest regarding this work.

References

[1] A. N. Fitriana, K. Mutijarsa, and W. Adiprawita, "Color-based segmentation and feature detection for ball and goal post on mobile soccer robot game field," in *2016 International Conference on Information Technology Systems and Innovation (ICITSI)*, pp. 1–4, Bandung, Indonesia, 2016.

[2] E. Rudiawan, R. Analia, P. D. Sutopo, and H. Soebakti, "The deep learning development for real-time ball and goal detection of Barelang-FC," in *2017 International Electronics Symposium on Engineering Technology and Applications (IES-ETA)*, pp. 146–151, Surabaya, Indonesia, 2017.

[3] D. Zhao, H. Wang, K. Shao, and Y. Zhu, "Deep reinforcement learning with experience replay based on SARSA," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–6, Athens, 2016.

[4] R. Dikairono, T. A. Sardjono, and D. Purwanto, "Visual ball tracking and prediction with unique segmented area on soccer robot," in *2017 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, pp. 362–367, Surabaya, Indonesia, 2017.

[5] N. Setyawan, N. Mardiyah, K. Hidayat, and Z. Has, "Object detection of omnidirectional vision using PSO-neural network for soccer robot," in *2018 5th International Conference on*

Electrical Engineering, Computer Science and Informatics (EECSI), pp. 117–121, Malang, Indonesia, 2018.

[6] H. Jiang, R. Gui, Z. Chen, L. Wu, J. Dang, and J. Zhou, "An improved Sarsa (λ) reinforcement learning algorithm for wireless communication systems," *IEEE Access*, vol. 7, pp. 115418–115427, 2019.

[7] D. B. Kusumawardhana and K. Mutijarsa, "Object recognition using multidirectional vision system on soccer robot," in *2017 International Conference on Information Technology Systems and Innovation (ICITSI)*, pp. 183–187, Bandung, Indonesia, 2017.

[8] S. B. Germi, A. Zamanian, M. A. Arzati, M. A. Khosravi, and R. FesharakiFard, "Estimation of moving obstacle dynamics with mobile RGB-D camera," in *2017 5th RSI International Conference on Robotics and Mechatronics (ICRoM)*, pp. 156–161, Tehran, Iran, 2017.

[9] N. A. Mardiyah, N. Setyawan, and M. N. Achmadiyah, "Autonomous mobile soccer robot localization using particle filter through Omni-vision.," *IOP Conference Series: Materials Science and Engineering*, vol. 732, no. 1, article 012096, 2020.

[10] E. Antonioni, V. Suriani, F. Riccio, and D. Nardi, "Game strategies for physical robot soccer players: a survey," *IEEE Transactions on Games*, vol. 13, no. 4, pp. 342–357, 2021.

[11] P. An, Z. Wang, and C. Zhang, "Ensemble unsupervised autoencoders and Gaussian mixture model for cyberattack detection," *Information Processing & Management*, vol. 59, no. 2, article 102844, 2022.

[12] S. Haobin, Z. Lin, P. Wei, and W. Shichao, "Robot soccer confrontation decision-making technology based on MOGM: multi-objective game model," *Journal of Intelligent & Fuzzy Systems*, vol. 28, no. 2, pp. 713–724, 2015.

[13] J. Suh and T. Tanaka, "SARSA (0) reinforcement learning over fully homomorphic encryption," in *2021 SICE International Symposium on Control Systems (SICE ISCS)*, pp. 1–7, Tokyo, Japan, 2021.

[14] X. Chen and P. Gao, "Path planning and control of soccer robot based on genetic algorithm," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 12, pp. 6177–6186, 2020.

[15] R. H. Abiyev, I. Günsel, N. Akkaya, E. Aytac, A. Çağman, and S. Abizada, "Robot soccer control using behaviour trees and fuzzy logic," *Procedia Computer Science*, vol. 102, pp. 477–484, 2016.

[16] R. A. Pamungkas, A. Maulana, D. P. Sya'ban, R. Ramdani, A. Musafa, and I. Riyanto, "WiFi data communication system design for wheeled soccer robot controller," *Advanced Science Letters*, vol. 24, no. 11, pp. 8782–8786, 2018.

[17] M. Abreu, L. P. Reis, and N. Lau, "Learning to run faster in a humanoid robot soccer environment through reinforcement learning," in *Robot World Cup*, pp. 3–15, Springer, Cham, 2019.

[18] X. Liu, "Retracted: Research on decision-making strategy of soccer robot based on multi-agent reinforcement learning," *International Journal of Advanced Robotic Systems*, vol. 17, no. 3, p. 172988142091696, 2020.

[19] H. Shi, Z. Lin, K. S. 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.

[20] D. Willemsen, H. Baier, and M. Kaisers, "Value targets in off-policy AlphaZero: a new greedy backup," *Neural Computing and Applications*, vol. 34, no. 3, pp. 1801–1814, 2022.

[21] M. Abreu, N. Lau, A. Sousa, and L. P. Reis, "Learning low level skills from scratch for humanoid robot soccer using deep reinforcement learning," in *2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pp. 1–8, Porto, Portugal, 2019.

[22] V. G. F. Barbosa, R. F. de Oliveira Neto, and R. V. G. Rodrigues, "A baseline approach for goalkeeper strategy using Sarsa with tile coding on the half field offense environment," in *2020 19th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*, pp. 1–8, Recife, Brazil, 2020.

[23] C. Xuanyu, L. Zhiwei, Y. Yongyi et al., "Multi-robot collaboration based on Markov decision process in Robocup3D soccer simulation game," in *The 27th Chinese Control and Decision Conference (2015 CCDC)*, pp. 4345–4349, Qingdao, China, 2015.

[24] Z. Chen, Z. He, H. Du et al., "Multi-stage decision-making skill learning for soccer robot," in *2021 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pp. 608–613, Xining, China, 2021.

[25] X. Tao, C. J. Zhang, and Y. J. Xu, "Collaborative parameter update based on average variance reduction of historical gradients," *Journal of Electronics and Information Technology*, vol. 43, no. 4, pp. 956–964, 2021.

[26] C. Celemin and J. Ruiz-del-Solar, "Interactive learning of continuous actions from corrective advice communicated by humans," in *Robot soccer world cup*, pp. 16–27, Springer, Cham, 2015.

[27] M. Dalgaard, F. Motzoi, J. J. Sørensen, and J. Sherson, "Global optimization of quantum dynamics with AlphaZero deep exploration," *Information*, vol. 6, no. 1, pp. 1–9, 2020.

[28] D. Speck, P. Barros, C. Weber, and S. Wermter, "Ball localization for RoboCup soccer using convolutional neural networks," in *Robot World Cup*, pp. 19–30, Springer, Cham, 2017.