

WrightOcean Team Description Paper^{*}

Qi Zhang^[0000–1111–2222–3333], Xinkai Xie^[1111–2222–3333–4444], Peng Wang^[2222–3333–4444–5555], Xiaoli Zhang^[0000–0001–7084–6798], Dunrui Liu^[0000–0001–8521–550X], and Fei Liu^[0000–0002–1541–501X]^{**}

Lab of Robotics, School of Information and Electrical Engineering, Ludong University, Yantai, Shandong, China

liufeildu@163.com

<http://www.wrightocean.org>

Abstract. In this paper, we describe the research focus, ideas implemented and future research directions in the WrightOcean team. Our code is based on the UT Austin Villa base code. We’ve optimized the basic skills such as walking and kicking using CMA-ES algorithm, and added a lot of high level strategies.

Keywords: RoboCup · CMA-ES · WrightOcean.

1 Introduction

As a new team of RoboCup Soccer Simulation 3D League, WrightOcean team was established in July 2017 and it belongs to Lab of Robotics, School of Information and Electrical Engineering, Ludong University in China. It still is a young and passionate team, and all of the members are undergraduates that are keen on robotics.

In order to focus on the field we are interested in, we use the base code release of the UT Austin Villa RoboCup 3D simulation team[1]. We have improved and added many of our own ideas based on the open source code. To test our work, we took part in the RoboCup China open in Shaoxing, China in April 2018 and received the third rank finally. We won the championship in the RoboCup China open in 2019 and the third place in the Robot World Cup in Sydney, Australia in the same year. And we won the champion in the RoboCup China open in 2020 and the runner-up in the 2020 RoboCup Germany Offenburg Tournament.

2 OVERVIEW

In order to quickly focus on areas of interest, such as robot motion control, intelligent decision-making and multi-agent cooperative strategy, we use the UT Austin Villa Base Code Release[2]. The code release provides a fully functioning agent. On this basis, we designed different kinds of kicks and goalie dives, and we

^{*} Supported by Laboratory of Robotics in Ludong University.

^{**} Fei Liu is the corresponding author.

used CMA-ES (Covariance Matrix Adaptation Evolution Strategy) algorithm[3] to optimize these basic skills, such as walking, kicking, and getting up (Section 3). In addition, we implemented many of our own ideas in the high-level strategy, which enabled the agent to make appropriate decisions according to different environments and make the whole team more consistent (Section 4). And the cooperative control of multi-agent will be our focus in the future.

3 Agent Skills

3.1 Walk

Walking like a human is very important for robots. Fast and stable walking is the key factor to improve the competitiveness of 3D football team, especially in the simulation 3D game. The UT Austin Villa code release has an omnidirectional walk engine based on a double inverted pendulum model[4], and includes slow and stable walk engine parameters. According to Patrick MacAlpine’s overlapping layered learning method[5], we optimize the parameters using CMA-ES algorithm.

3.2 Kick

Kicking as a basic skill is as important as walking. A powerful kick can help us score. And we can also use kicking to pass ball. It is the basis of the team cooperation system.

UT-Austin-Villa base code supports a stable but short kick skill. But it can’t threat the opponents’ goal. So we create a new kick skill. Then, we put the parameters of the skill into the CMA-ES algorithm[3] to optimize. These are as follows.

Create Kick Skill UT-Austin-Villa base code has a kicking skill execution mechanism based on key frame. One kicking skill is generated by adding a keyframe composed of joint angles. To do it more efficiently, we use the AIUT3D Motion Editor[6]. It is a open source software, developed by the AIUT Team. The action we designed is shown in Figure 1.

Optimize We can use the action editor to give players a variety of kicking skills, but the initial kicking skills are weak and we need to have strong kicking skills. So at this point, we need to optimize and strengthen the player’s kicking skills to achieve the desired results. We use the CMA-ES[3] algorithm, which is mainly used to solve the continuous optimization problem. We sample multiple Gaussian distributions to get a new solution, and update the parameters of the Gaussian distribution with a better solution. The evolutionary strategy algorithm is mainly used as a method to solve the parameter optimization problem. It mimics the principle of biological evolution and assumes that the results (traits)

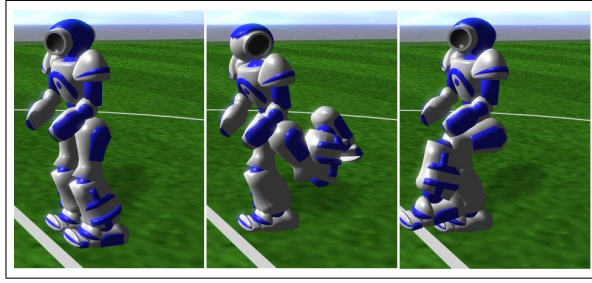


Fig. 1. kick skill

no matter what changes occur in genes always follow the zero mean value and gaussian distribution of a certain variance. Note that evolutionary strategies are different from genetic algorithms, but both are important variants of evolutionary algorithms (EAS). The main features of the algorithm are as follows: using multivariate normal distribution to generate new search points, sorting based selection process, step size control to make rapid convergence more convenient, covariance matrix adaptive algorithm to increase the likelihood of successful step size, can improve performance according to the order of magnitude of the problem. At the same time, according to the method of hierarchical optimization learning [5], the optimization process is divided into actions and behaviors for separate optimization, so as to simplify this difficult process. Task decomposition is a popular method of learning complex control tasks when single block learning is difficult to complete. Hierarchical learning is a machine learning paradigm of hierarchical task decomposition, which aims to learn complex behaviors by learning a series of sub-behaviors step by step. A key characteristic of layered learning is that the upper layer is directly dependent on the lower layer that has been learned. The lower levels are frozen before learning the higher levels. Meanwhile, stratified learning introduces the possibility of learning some behaviors at the same time by learning higher level and lower level at the same time, thus reducing this limitation in the search space. However, one potential disadvantage of increasing the size of the search space is the increase in dimensions, which may make learning more difficult. We use an extension of the hierarchical learning paradigm called overlapping hierarchical learning, which allows specific parameterized behaviors to be independently learned and then learned at the "seams" where they affect overlap, splicing them together and aiming to provide a middle ground. For example, in the process of optimizing the big kick, the movement of the big kick should be optimized at a fixed position first. After the optimization of the movement becomes mature, the movement towards the goal and kicking the ball should be optimized on this basis. Other kicking skills are basically the same.

3.3 Kick Type Selection

We have three types of kick: long range kick, fast kick and dribble. When the kicker gets the ball, how to choose a reasonable kick type is important. We build up an evaluation system for kick type selection. The robot will evaluate each type of kick according to the situation on the field, and the result is whether to choose this type of kick. If all three kick types are available, we prefer long range kick, then fast kick, and finally dribble.

In our project, the ball holder will always choose the type of kick, including the dribbling and kicking action. If it chooses to dribble, it will use a reactive dribbling strategy to push the ball toward the opposing goal. If the kick action is chosen, the fixed action is performed by kicking the ball towards the selected target point, which is selected between the opposing goal and the passing point. When the ball belongs to the opponent, we still produce a ball handler that will tackle the ball. There are many ways to choose the type of kick. The most basic method is to choose the type of kick based on the available data of the game environment. Another method is to use the formula to calculate the score of several types of kick and choose the type of kick according to the score. When the opposing team has the ball, it is easy for us to go for the ball and get ejected by the server because of the collision. A tackle strategy involves a certain amount of deviation (depending on the speed of the ball) from the holder's tackle point towards the ball's velocity. This allows the robot to tackle with a lead and reduce penalty appearances.

3.4 Get Up

The robot will inevitably fall down due to collision and other reasons in the game, . So a fast and stable get up skill can help team to grab the ball. UT-Austin-Villa base code supports a stable but slow get up skill. We use the CMA-ES algorithm[3] to optimize the get up skill so that the robot can get up more fastly. Meanwhile, in order to make the robot walk smoothly after getting up, we trained trained both the getting up skill and walking together.

3.5 Dive

In order to defend the opponents' shot, we design some dive actions for the goalie. Just like kicking skill, we used AIUT3D Motion Editor[6] to accomplish it. We designed three dives to defend shots from any angles. They are shown in Figure 2.

4 Strategy

4.1 Ball Holder Decision

RoboCup 3D is a multi-agent environment full of cooperation and confrontation. The team's overall strategy is a typical distributed multi-agent problem. It makes

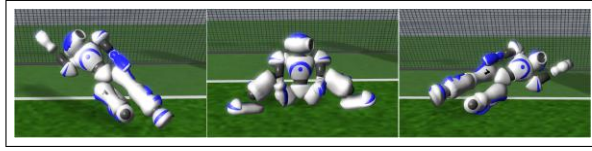


Fig. 2. dive skill

full use of existing resources to achieve a common collective goal, that is to win the game. Its behavior decision is a kind of mapping from the perception state to the execution action. It establishes an environment model based on perception, and selects the most appropriate action on this basis. This decision depends on the strategy of the whole team, which in turn determines the coordination between the behaviors of these individuals. Only when every player performs his part in the same strategy system can the whole team fully embody a kind of coordination. In the game, the team's goals, passes and intercepts are all completed by the ball holder. Therefore, the decision-making of the ball holder is particularly important in the game. In our project, the ball holder will always choose the type of kick, including dribbling and kicking.

If it chooses to dribble, it will use a reactive dribbling strategy to push the ball toward the opposing goal. If the kick action is chosen, the fixed action is performed by kicking the ball towards the selected target point, which is selected between the opposing goal and the passing point. When the ball belongs to the opponent, we still produce a ball handler who tackles the ball.

4.2 Role Assign

As is known to all, soccer is a team sport, the tacit cooperation between players is the key to the team's winning. A team has 11 players, and how to control them to make the whole team show consistency is an interesting and challenging work. There are many roles in football games, such as striker and defender. Everyone has his own mission. In the 3D competition, each robot is isomorphic, so we can dynamically change their roles to improve the team's offensive and defensive efficiency.

First of all, we determine the current team formation according to the position of the ball. Of course, our formation is not static. If an opponent enters an area that is threatening to us, we consider changing the current formation and adopting a man-to-man defensive strategy.

We consider how to assign agents to these positions after the formation is determined. We first select a player holding the ball based on factors such as position, orientation, whether it falls, and the opponents' position. We consider the match between the remaining players and positions as a bipartite graph matching problem. Our goal is to make the convergence time of the formation as short as possible. Therefore, we use the distance between the player and the position as the basis weight, and adjust according to the player's orientation,

whether it falls, and whether there are obstacles between the player and the target. Finally, the Hungarian algorithm is used to solve the problem[7].

4.3 Pass

In order to improve the ball control rate and shooting success rate of the team, players should create better shooting opportunities by controlling the ball and moving. In the previous passing mode, we set up some passing target points according to the distance of kicking ball, and calculate the score of each point according to the situation on the field. Then the ball holder will kick the ball to the point with the highest score, and our role assignment system will assign a robot to this point to receive the ball.

This year, we use the idea of deep learning to improve our passing mode. We use deep learning to train a neural network based on value evaluation. Through a lot of training, our players will choose the point with the highest score after network training as the passing point when carrying out the passing so that our passing will be more accurate.

4.4 Communication

On the football field, the information that the player sees is limited and may have delay. For example, Due to the limited visual range of the robot, The robot may not be able to see the enemy robot behind it on the football field, or It may perceive that the position of the enemy robot is delayed or inaccurate. If the coordinate position of the enemy robot behind it is determined by my team. If the friends see it, the team-mates will send the position of the enemy through communication. The robot can get the coordinate information of the enemy by analyzing the message from the team-mates. We added the enemy's coordinate position information according to UT's paper.

In addition, we also need to ensure that every enemy player has an equal opportunity to be seen by our players and send their coordinate information accurately to prevent the information of some enemy robots from being accurately obtained in a certain period, which will affect the decision-making. The delay of enemy position information transmitted by communication is small and accurate.

4.5 Formation

The game needs multi-agent cooperation to achieve the common goal, and the important embodiment of cooperation is the formation of players on the field. Formation is required by multi-agent cooperative behavior, which makes multi-agent cooperate in an orderly and intelligent way. In order to meet the needs of multi-agent cooperation in RoboCup real-time dynamic environment, we design the corresponding formation strategy according to different modes and situations on the court. Generally, the formation is determined by the position of the ball.

However, it is sometimes unreasonable to rely only on the position of the ball to decide the formation. For example, when an opponent is waiting for the ball in our half court, once he receives the ball, our players have no time to defend and lose points. Therefore, we optimized the original man to man strategy to deal with the situation we encountered on the court, which requires us to predict the form of the court in advance, so as to make a dynamic adjustment of the formation in time. The game needs multi-agent cooperation to achieve the common goal, and the important embodiment of cooperation is the formation of players on the field.

4.6 Kick Decision Classifier

In this year, we used machine learning method to optimize the kick selection problem. There are three types of kicking in our project: long distance kicking, fast kicking and dribbling. We collect the rewards of using different kicking methods in various environments, and use supervised learning method to train a kicking decision classifier. When the ball holder wants to play, we will choose a way to play according to the score of the kick decision classifier.

5 Future Work

At present, we still have some gaps in basic skills (like kicking and walking) compared with the top teams. So our priority is to continue to improve these skills. With reliable skills, our focus will be on multi-agent cooperative control. We will explore different ways to implement a variety of team strategies. At the same time, we are studying the use of reinforcement learning to train agents, and we are doing some work in this area.

Acknowledgment

This work was supported by Science and Technology Program of Yantai, China (Grant No. 2019XDHZ085), Major Basic Research Project of Natural Science Foundation of Shandong Province, China (Grant No. ZR2018ZC0438), National Natural Science Foundation of China (Grant No. 61673200), Program of Natural Science Foundation of Shandong Province, China (Grant No. ZR2017BF032), and Laboratory of Robotics in Ludong University.

References

1. Patrick MacAlpine and Peter Stone, "UT Austin Villa RoboCup 3D Simulation Base Code Release," in *RoboCup 2016: Robot Soccer World Cup XX, Lecture Notes in Artificial Intelligence*, Eds. Sven Behnke, Daniel D. Lee, Sanem Sariel, and Raymond Sheh, Berlin: Springer Verlag, 2017, pp. 135-143.
2. UT Austin Villa Base Code Release. [Online]. Available: <https://github.com/LARG/utaustinvilla3d>

3. N. Hansen, S. Müller, P. Koumoutsakos, "Reducing the time complexity of the de-randomized evolution strategy with covariance matrix adaptation (cma-es)," *Evolutionary Computation*, vol. 11, no. 1, pp. 1-18, 2003.
4. Patrick MacAlpine, Samuel Barrett, Daniel Urieli, Victor Vu, and Peter Stone, "Design and Optimization of an Omnidirectional Humanoid Walk:A Winning Approach at the RoboCup 2011 3D Simulation Competition," in *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI)*, July 2012.
5. Patrick MacAlpine and Peter Stone, "Overlapping Layered Learning," *Artificial Intelligence*, vol. 254, pp. 21-43, January 2018.
6. A motion editor for designing arbitrary key frame based motions for Nao to be used in RoboCup 3D soccer simulation league. [Online]. Available: <https://github.com/AIUT3D/aiut3d-motion-editor>
7. Munkres, James, "Algorithms for the Assignment and Transportation Problems," *Journal of the Society for Industrial and Applied Mathematics*, vol. 5, no. 1, pp. 32-38, 1957.