RobôCIn Team Description Paper 2023

Cristiano Santos de Oliveira[1], Mateus Gonçalves Machado[1], Walber de Macedo Rodrigues[1], Thiago da Silva Araújo[1], Pedro Vitor Cunha[1], Rafael dos Reis de Labio[1], Felipe Nunes de Almeida Pereira[1], Mateus Ferreira Borges Soares[1], Gabriel Lopes de Souza[1], Maria Luísa dos Santos Silva[1], Edna Natividade da Silva Barros[1], Paulo Salgado Gomes de Mattos Neto[1], and Tsang Ing Ren[1]

Universidade Federal de Pernambuco, Centro de Informática, Recife PE, Brazil
`robocin@cin.ufpe.br`
`https://www.robocin.com.br/`

Abstract. RobôCIn Soccer Simulation 2D team, based at the Universidade Federal de Pernambuco, was founded in 2018. In our debut competition at the Latin American Robotics Competition (LARC) in João Pessoa, Paraíba, Brazil, we secured fourth place against other Latin American teams. The following year, we competed in the RoboCup for the first time and achieved a ninth-place finish. Also, we had won second place in the Brazil RoboCup Open 2019 (LARC). In 2020, we obtained third place in the Brazil RoboCup Open 2020 (LARC). In 2021, we placed seventh at the RoboCup, and were champions at LARC for the first time. In 2022, we achieved the tenth position at RoboCup and became two-time champions of LARC.

Keywords: Data Science · Data Analysis · Graphic User Interfaces · Web Development · Statistics · Classification · Formation · Adaptative Formations · Behaviour

1 Introduction

RobôCIn is a robotics research team from the Centro de Informática (CIn), Universidade Federal de Pernambuco (UFPE), created in 2015 to participate in competitions and research subjects related to robotics. We are currently working in three categories: Soccer Simulation 2D (SS2D), Very Small Size (VSS) since 2015, Small Size since 2018.

RobôCIn utilized the well-structured base of agent2d 3.1.1, as presented in [2], as the foundation for our code development in the first year. To improve performance, we subsequently integrated gliders2d-v1.6 [7] and MarlikBlock [8] to enhance the agent's movement behavior. Also, we are investigating the recently released Cyrus2D[9] to possible points of integration in the future. Our most recent agent release includes two significant new features: a implementation of the Hungarian Algorithm for agent positioning selection and corner kick dynamic opponent marking. Additionally, we updated the librcsc used in our agent, enhanced our goalkeeper to address issues resulting from the server 17.0.1 update[1] and developed a new infrastructure tools to support our ongoing development efforts.

RobôCIn has designed and implemented a containerized testing infrastructure and development environment. This testing environment allows the integration of the latest binaries we develop and runs simultaneous execution of multiple games to verify performance. Furthermore, it incorporates an analyzer infrastructure which can be enhanced through the implementation of personalized python scripts for improved log analysis. One of the main difficulties encountered was the complexity of setting up the RoboCup 2D environment. To ensure consistency across multiple platforms, we developed a Docker-based container for 2D simulation that provides an isolated environment independent of the host system.

2 Forward Players Positioning

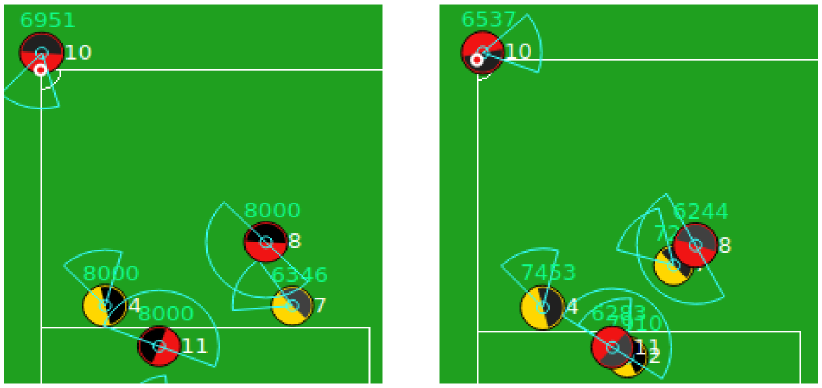
The Voronoi's discretization method is based on a set of points, with the objective of divide an Euclidean Plane in to a set of sites, in our case we are using the opponents' players' positions. As proposed by Gliders [7], the Voronoi algorithm provides a set of potential positions for the attacking players.

To filter Voronoi's output points, a circumference is created with the ball as center, resulting in a refined set of potential locations for the forwards players. Then the problem becomes how to assign our attacking players into nearest point inside the defined circumference. And, Inspired on our Small Size League team open source implementation of the Hungarian algorithm[3] we incorporated the SIM2D implementation of it to solve Voronoi's positions assignment problem. So, the Hungarian algorithm [5], solves the assignment problem in polynomial time, making it an efficient solution to determining the best assignment between the set of potential positions and the players themselves, taking into account their distances from the Hungarian output points.

3 Dynamic Defense against Opponent Corner Kick-ins

RobôCIn's agent positioning for opponent corner kick-ins is based on static information - the formation file configuration -, the usage of this file configuration can be observed on Fig. 1, determining the positions of the players during kick-in scenarios. The players were programmed to search for their designated positions, remaining stationary until the "play_on" game. However, this approach is limited in that it produces the same defensive positions regardless of the opponents' positioning inputs, thereby restricting the agent's ability to adapt and behave differently in response to varying position states.

To effectively tackle this issue, the chosen strategy involves adjusting the defensive players positions who are within a defined radius from the corner mark to the nearest unmarked opponent's position relative to their own, as it can be seen in Fig 1. The subsequent step involves utilizing the Hungarian Algorithm [5], a highly regarded method for resolving assignment problems. To evaluate the algorithm's performance, we conducted statistical analyses using the SoccerAnalyzer[6].



(a) Static marking (b) Dynamic marking

Fig. 1: Players positioning before (a) and after (b) positional adapting.

4 Defensive Ball Interceptor

In recent RoboCup matches, it was observed that opponents were able to easily break through our defense lines using long passes and had a higher total number of passes than our team, RobôCIn. To address this issue and hinder the opponent's passing game, a new defensive ball interceptor was implemented.

Inspired by Tehran's 2022 Team Description Paper [4], the defensive ball interceptor was designed to improve our team's defensive skills and counter the ball pitches into our goal or even into the area. The interceptor was integrated into RobôCIn's agent, utilizing a set of vectors between the goal line, ball, and agents, along with the ball's direction, to determine whether an action is suitable for the current field situation.

The ball interceptor operates by predicting the best interception point of the ball-to-goal or opponent-to-opponent trajectory and then directing the nearest teammate agent to catch the ball on its moving point. By doing this, it ensures that the defense is well-prepared to intercept the ball before it reaches our goal or even our defensive area.

Overall, the implementation of the defensive ball interceptor provides a scientific and effective approach to improving RobôCIn's defense strategy. Through advanced modeling and predictive techniques, our team can now better anticipate the opponent's moves and more effectively defend against their attacks.

5 Goalkeeper

Last year, there was a Major update to RoboCup Soccer Simulation Server[1] which introduced new parameters and constraints regarding the goalkeeper movement and actions. This update had a big impact both on our goalkeeper performance and our overall winning rate. To address this situation, we decided to re implement our goalkeeper agent behavior based on Helios's[2].

We first noticed the problem when, in defending situations, our goalkeeper suddenly stood still near the post on wide lane attacks. This behavior led to open goal scenarios where adversary attackers could easily score shooting from a distance, mostly from outside the box.

Based on this problem, we implemented two new goalkeepers. One behaves differently whether the ball is coming to goal or not. If not, the goalie tries to maximize its catchable area by constantly moving side to side as adversary passes occur in front of him. If the ball is shot towards the goalie, it tries to catch the ball whether or not it has a high success rate on the catch. The second goalkeeper only uses a simpler version of the Defensive Ball Interceptor algorithm to track where is the optimum point for it to intercept the ball. Also, the omni-directional movement was implemented in goalie behaviors, a feature that was not being used in our team until now.

6 RoboCIn Testing Module

Improving the testing infrastructure has always been a crucial aspect for efficiently gathering statistical results and an extensive set of logs against various teams. Additionally, it was essential to develop a testing module that is straightforward to use for adding new team binaries and maintaining compatibility with the latest competition-based operational system.

The RobôCIn Testing Module is a comprehensive testing module that offers containerized and parallel game execution to minimize waiting time, and statistical test result as can be seen in Table 1. Additionally, it is equipped to run pre-configured training scenarios created by the offline coach, and requires only a one-time setup for binary execution.

The user has the responsibility of specifying the number of games and the opposing teams for the RobôCIn development version in a configuration file. The root computer then launches the containers and manages the game execution. At the conclusion of the games, the root computer collects and analyzes the game logs. Each container, based on the number of available nodes, runs the specified number of games and preserves the respective logs.

	RBCNDev	RBCNMaster
Victories	36.02	33.80
Deafeat	33.80	36.02
Draw	30.18	30.18
Score	566	539
G.B	27	-27
S.P.G	1.14	1.08

Table 1: Shows a standard sample RoboCIn-tester output. G.B stands for Goal Balance and S.P.G for score per game.

7 Experiments and Results

To validate the modifications developed, RoboCIn Testing Module is used to evaluate statically. It is designed a series of games and run around five hundred times to generate a sufficient amount of data. This process allows us to analyze the performance of the changes we've made and ensure they meet our standards.

7.1 Dynamic Corner Marking

In order to check the improvement at the dynamic corner marking feature proposed, the offline coach was utilized, making a corner play mode after three hundred cycles. The training sessions were played against the improved defense. The analysis can be seen on Table 2.

	RBCNStatic	BCNDynamic
P.G.S	6.8	3.5
P.C.I	26.0	71.0
P.B.T.C	72.1	74.0

Table 2: Statistical Analysis for Dynamic Corner Marking. P.G.S stands for Percentage of Goals Suffered, P.C.I stands for Percentage of Corner Kick Intercepts, and P.B.T.C Percentage of Ball Takings in Corner Area.

An essential aspect to consider is the mean number of goals conceded, which has been significantly reduced. This indicates that the system has been successful in improving its defensive capabilities. Additionally, the use of dynamic marking has been shown to improve the intercepts at corner kicks, suggesting that the defenders are now better positioned on the field.

7.2 Forwards Players Positioning

To evaluate the effectiveness of our updated assignment for Voronoi's output using Hungarian Algorithm, we conducted a series of 500 games against our developer branch. By comparing the results of these games, as can be seen in Table 3, we were able to quantity the improvement provided by the new algorithm.

	RBCNHung	RBCNDev
Victories	34.96	31.43
Deafeat	31.43	34.96
Draw	33.61	33.61
Score	562	545
G.B	17	-17
S.P.G	0.94	0.92

Table 3: Statistical analysis results of the Hungarian algorithm use on forwards players positioning. G.B stands for Goal Balance and S.P.G for Score per Game

7.3 New Goalkeeper

In order to assess the impact of our new Goalkeeper class, which focuses on goalie positioning and blocking ball trajectory targets, we conducted a series of 500 and 1000 games. A five hundred games for the first goalkeeper (RBCNGK1) and thousand games for the second goalkeeper (RBCNGK2) to validate the implementations. Both of them played against the same adversary, our latest stable agent.

Despite the difference in number of games played, we can see a higher tendency of defeat rate on RBCNGK2 rather than RBCNGK1.

	RBCNGK1	RBCNKG2
Victories	38.51	33.58
Deafeat	29.44	39.74
Draw	32.06	26.68
Score	566	1269
G.B	82	-156
S.P.G	1.0846	1.1674

Table 4: RBCNGK1: Goalkeeper based on adversary incoming shot. RBCNGK2: Goalkeeper based on Defensive Ball Interceptor

References

1. Robocup soccer simulation server. `https://github.com/rcsoccersim/rcssserver`, accessed: 2019-01-01
2. Akiyama, H., Nakashima, T.: Helios base: An open source package for the robocup soccer 2d simulation. In: Behnke, S., Veloso, M., Visser, A., Xiong, R. (eds.) RoboCup 2013: Robot World Cup XVII. pp. 528–535. Springer Berlin Heidelberg, Berlin, Heidelberg (2014)
3. Cruz, J.: Soccer common (2022), `https://github.com/robocin/soccer-common`
4. Khanjari, S.: Tehran2D team description paper (2022), `https://rcsoccersim.github.io/robocup2022/TDP/TDP_Tehran2D.pdf`
5. Kuhn, H.W.: The hungarian method for the assignment problem (1995), `https://web.eecs.umich.edu/~pettie/matching/Kuhn-hungarian-assignment.pdf`
6. Pereira, F.: Socceranalyzer (2022), `https://github.com/robocin/SoccerAnalyzer`
7. Prokopenko, M., Wang, P.: Gliders2d: Source code base for robocup 2d soccer simulation league (2018)
8. Tavafi, A., Nozari, N., Vatani, R., Yousefi, M.R., Rahmatinia, S., Pirdeyr, P.: MarliK 2011 Soccer 2D Simulation team description paper (2011), `http://archive.robocup.info/Soccer/Simulation/2D/TDPs/RoboCup/2011/MarliK_SS2D_RC2011_TDP.pdf`
9. Zare, N., Amini, O., Sayareh, A., Sarvmaili, M., Firouzkouhi, A., Rad, S.R., Matwin, S., Soares, A.: Cyrus2d base: Source code base for robocup 2d soccer simulation league. arXiv preprint arXiv:2211.08585 (2022)