ITAndroids 2D Soccer Simulation Team Description Paper 2025

Luiz F. B. Ramos, Marcos R.O.A Maximo, Diogo C. L. de Paula, José A. F. Tizon, Gabriel S. Tedesco, Paulo A. C. da Silva, and João G. M. Gonzaga

Aeronautics Institute of Technology
São José dos Campos, São Paulo, Brazil
{ramosbrito2004, dcl.depaula}@gmail.com, mmaximo@ita.br
itandroids-soccer2d@googlegroups.com
http://www.itandroids.com.br

Abstract. This paper presents the latest developments in ITAndroids 2D, a RoboCup Soccer Simulation 2D (RCSS2D) team. Building upon previous iterations, we introduced key improvements in defensive marking, ball interception, and decision-making logic. A new marking strategy was implemented to enhance defensive positioning near the goal, minimizing vulnerabilities against opposing attacks. Additionally, the interception system was refined by incorporating an optimized trajectory estimation model, allowing players to reach the ball more efficiently. Another significant fix addressed an issue where players would enter an infinite spinning loop when in possession of the ball. The performance of the updated team was evaluated against last year's version over a set of 1,000 matches, achieving a win rate of 34.3%, a tie rate of 42.5%, and a loss rate of 23.2%.

1 Introduction

ITAndroids is a competitive robotics team from Aeronautics Institute of Technology reestablished in 2011. The group participates in the following leagues: RoboCup Soccer Simulation 2D (RCSS2D), RoboCup Soccer Simulation 3D, RoboCup Humanoid Kid-Size, IEEE Humanoid Robot Racing, IEEE Very Small Size, and RoboCup Small Size League.

Our RCSS2D team, ITAndroids 2D, has continuously participated in the Latin American Robotics Competition (LARC) and Brazilian Robotics Competition (CBR – acronym for Competição Brasileira de Robótica) since 2011. ITAndroids 2D competed in RoboCup from 2012 to 2024, except in 2014 and 2020. Table 1 shows the placements in these competitions.

2 Previous works

Our code base uses agent2d [1] as the base team. Since 2011, we have focused on improving the mechanisms already present in the agent2d framework. We improved the action chain evaluator with Particle Swarm Optimization (PSO) [2].

Table 1: Placement of ITAndroids 2D in past RoboCup and LARC competitions.

Year	RoboCup	LARC
2024	11th	4th
2023	10th	3rd
2022	11th	3rd
2021	11th	5th
2020	—	4th
2019	13th	1st
2018	9th	2nd
2017	15th	3rd
2016	13th	2nd
2015	13th	1st
2014	—	1st
2013	13th	1st
2012	10th	1st

Furthermore, we developed heuristics [3] to increase attack and defense performance: type of formation (attack or defense) selection based on probability of scoring a goal, field evaluator selection based on opponent team, and optimal defender marking in opponent attack situations. Many early improvement ideas were inspired by HELIOS [4] and Nemesis [5].

The team proposed in 2018 a novel technique to determine the in-game ball possession [6]. We used a Finite-State Machine called Possession Automaton that takes into account the current and the last game situations to infer the ball possession. Since we do not estimate ball possession based on a single game cycle, we obtained a classification accuracy 18% higher than the default estimator of the base team.

We experimented with Deep Reinforcement Learning (DRL) techniques to improve goalkeeper defense in penalty situations [7]. After five training experiments using the Proximal Policy Optimization (PPO) algorithm, we achieved a penalty defense rate of 40% against agent2d, twenty percent higher than the base team rate.

In 2024, we developed an individual defensive behavior using DRL [8], which taught players how to get the ball from the opponent by kicking or tackling it. An extended implementation of the Deep-Q Network (DQN) algorithm was selected for training. This approach was rewarded with an action policy that considerably outperformed agent2d.

Currently, our efforts are focused on developing a better marking behavior for the defense of our goal when opponents are close, improving the intercept time estimation, as well as fixes for some observed mistakes in player decision-making.

3 Marking during goal defense

We have noticed that several of the goals scored against our team can be traced back to poor overall marking of the opponents near the goal area. In these

situations, our players are often seen crowding one specific opponent or are badly positioned to intercept a shot to the goal, as seen in 1.

Fig. 1: An example of poor marking executed by our team (in yellow). Player 2 is unprepared to block opponent's 11 advance, and player 3 is not in position to block if opponent 9 receives the ball and attempts to shoot.

To address this, we separated the marking behavior into two: the default one used when players are far away from the goal, and a new one used whenever the opponents are close to our goal (our player must also be closer to the goal than the opponent) and we are in a defense formation. Taking inspiration from similar attempts at this problem, such as the "Goalie Buddy" from the IraNad 2D soccer simulation team [9], the players in the latter circumstance will attempt to mark the opponents and position themselves according to the bisecting angle between the goalie and the end of the goal closest to them. Additionally, they will tighten the marking distance more than usual, since tackling the ball away is likely to break down the opponents' attack.

This implementation was awarded with a 1.52 win to loss ratio (evaluated over 5 hundred matches) against the master version of ITAndroids, which is a substantial amount.

An interesting side effect of that implementation is that since players inside and outside the goal zone work in different marking systems, two players can mark the same opponents, effectively boxing in the attackers. It is still unclear to us whether that contributes to the overall gain in performance achieved by this approach.

4 Intercept table improvement

Whenever a free ball is detected by one of our players, be it from a pass from its own team or from the opponent's, it creates a table with all of the known teammates in its vicinity and the estimated number of cycles with which they could reach the ball, which constitutes what we call its intercept table. Using that table, it determines whether itself should chase after the ball or not.

In previous iterations of our code, however, such estimation was calculated using the ball's inertia point, which means the point at which its velocity reaches zero. This causes our team to not only assign the wrong players to chase after the ball, but also make the assigned player start running in a suboptimal direction to catch it. It also negatively affects the passes between our players, since in numerous occasions, the player receiving the pass will run towards the inertia point and lose the ball to an opponent intercepting its path.

This was changed using an approach that, at least in the current stage, disregards the noise in the information received by the players, treating it simply in terms of a physical problem where the player's speed is fixed and the ball's trajectory is decelerating. In that setup, we wish to find the direction the player should run to reach it in the minimal number of cycles. That is, solving the equation:

$$\boldsymbol{r}^0_{ball} + \boldsymbol{v}^0_{ball} t + \frac{1}{2}\boldsymbol{a}_{ball} t^2 = \boldsymbol{r}^0_{player} + v_{player}\hat{\mathbf{v}}_{player} t \tag{1}$$

where:

$$\hat{\mathbf{v}}_{player} = (\cos\theta, \sin\theta) \tag{2}$$

and we want θ to minimize the time t. After this, it becomes a simple matter of minimizing the function:

$$t = f(\theta) \tag{3}$$

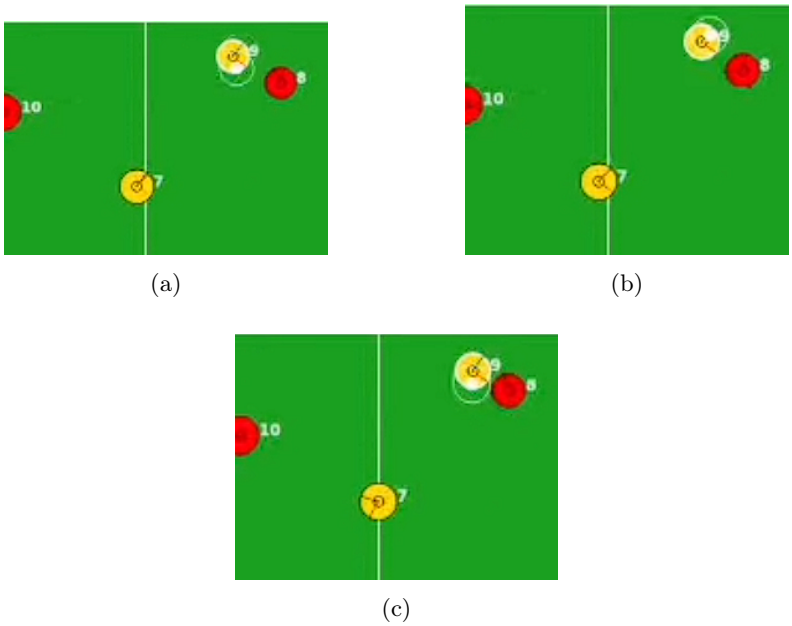
Overall, this update rendered us an improvement of 1.23 win to loss ratio (evaluated over 5 hundred matches) against the ITAndroids master branch.

5 Spinning problem

A behavior we often observed in matches during our previous RoboCup and LARC participation was that a player in possession of the ball would start spinning for an indefinite amount of time (as shown in figure 2), which would eventually lead to it being tackled or trapped by opponents.

This turned out to be a fault in its pass behavior, where spinning the ball to readjust its position could lead to an infinite loop, and it is only broken whenever an opponent enters tackling range.

In specific regions of the field, where the passing algorithm suspects we are in an attacking situation, and the opponents are not in tackling range yet, it can determine that a field scan should be executed before the pass. However, the mechanism that determines whether this 'scan field' command has been executed already or not depends on a float comparison, which had not been proofed of floating point errors. Therefore, it could fail to recognize that such a scan has already been done, and continue spinning for another cycle. The solution to this was a simple error range correction.



(a) (b) (c)

Fig. 2: Player 9 performing a spin.

6 Conclusions and Future Work

In this work, we introduced several improvements to ITAndroids 2D, focusing on defensive marking, intercept behavior, and resolving the spinning issue in ball possession. The marking strategy was refined by introducing a dedicated approach for goal defense, enhancing player positioning to reduce vulnerabilities near our goal. Additionally, we improved the intercept table calculation by optimizing trajectory estimations, allowing players to reach the ball more efficiently. Lastly, we addressed the spinning problem, which previously caused players to get stuck in an indefinite loop when attempting to pass. The compounded difference of these changes can be observed in table 2.

Table 2: Performance of ITAndroids2025 against ITAndroids2024 evaluated over a set of 1000 matches.

Outcome	Percentage
Wins	47.1 %
Ties	34.1 %
Losses	18.8 %

In the future, we wish to use reinforcement learning to better assign player positions to the available heterotypes, as well as to restart the update of our librcsc version.

7 Acknowledgments

We would like to acknowledge the RoboCup 2D Soccer Simulation community for sharing their developments. In particular, we would like to acknowledge Hidehisa Akiyama for agent2d, librcsc, soccerwindow2, and fedit2 [1]. Finally, we acknowledge our sponsors: Altium, CENIC, Field Pro, Intel, ITAEx, MathWorks, Micropress, Neofield, Polimold, Rapid, SIATT, SolidWorks, STMicroelectronics, Virtual Pyxis, and Visiona Space Technology.

References

1. Hidehisa Akiyama and Tomoharu Nakashima. "HELIOS Base: An Open Source Package for the RoboCup Soccer 2D Simulation". In: *The 17th annual RoboCup International Symposium*. July 2013.
2. Fábio Mello et al. "ITAndroids 2D Soccer Simulation Team Description 2012". (2012).
3. Felipe Coimbra et al. "ITAndroids 2D Soccer Simulation Team Description Paper 2017". (2017).
4. Hidehisa Akiyama and Hiroki Shimora. "HELIOS2010 Team Description". (2010).
5. Mehrab Norouzzilatab et al. "Nemesis Team Description 2010". (2010).
6. Felipe Coimbra and Lucas Lema. "ITAndroids 2D Soccer Simulation Team Description 2018". (2018).
7. Diego Fidalgo et al. "ITAndroids 2D Soccer Simulation Team Description Paper 2020". (2020).
8. Davi Vasconcelos et al. "ITAndroids 2D Soccer Simulation Team Description Paper 2024". (2024).
9. Reza Aghayari et al. "IraNad Soccer 2D Simulation Team Description Paper 2022". (2022).