

ITAndroids 2D Soccer Simulation

Team Description Paper 2017

Felipe Coimbra, Miguel Neto, Camila Rapereda, Guilherme Marinot, Cláudio Silva,
and Matheus Marcontes

Aeronautics Institute of Technology,
São José dos Campos, São Paulo, Brazil
{camila.rapereda, claudiosilva.clfs, felipecoimbra97, guilhermemarinot,
matheusdouglass125, miguelneto.1995}@gmail.com
itandroids-soccer2d@googlegroups.com
<http://www.itandroids.com/>

Abstract. ITAndroids 2D Soccer Simulation team was reestablished in mid-2011 by undergraduate students at Technological Institute of Aeronautics. The team is currently one of the strongest teams in Brazil, having won 2nd place in the 2017-year edition of Brazilian Robotics Competition besides the 1st place in the previous four editions. Moreover, the team has also qualified for the last **four** editions of RoboCup. This paper describes our developments in the last years, including improvements to agent2d's action chain framework, hand-coded heuristics to improve critical areas of the team, a log analyzer tool, improvements to the goalie's behavior. Moreover, we discuss our plans for future development.

1 Introduction

ITAndroids is a robotics competition team from Aeronautics Institute of Technology reestablished in 2011. The group participates in the following leagues: RoboCup 2D Soccer Simulation (Soccer 2D), RoboCup 3D Soccer Simulation, RoboCup Humanoid Kid-Size, IEEE Humanoid Robot Racing and IEEE Very Small Size, besides currently having plans of start competing in RoboCup Small Size league.

Our Soccer 2D's team ITAndroids 2D has continuously participated in Latin American Robotics Competition (LARC) and Brazilian Robotics Competition (CBR) since 2011. Moreover, ITAndroids 2D competed in RoboCup in 2013, 2015 and 2016. The team also qualified for RoboCup 2014, but unfortunately it was not able to attend to the competition.

ITAndroids 2D is considered a very strong Soccer 2D team in Brazil and Latin America. ITAndroids 2D won 1st place in LARC/CBR from 2012 to 2015 and 2nd place in 2016. In RoboCup, we placed 10th in 2012 and 13th from 2013 to 2016.

Since the beginning, the progress of our soccer 2D teams was largely supported by the RoboCup community. Our current code is based on agent2d [10] and a large amount of our work was focused on improving mechanisms already present in agent2d framework. Also, many of our ideas were inspired by other teams' work, specially HELIOS [1] and Nemesis [2]. Intending to give back to the community, we created a wiki in Portuguese, especially to help new teams in Brazil.

This paper describes our development efforts in the last years and points out some improvements we want to implement in a near future. Sec. 2 describes improvements to the action chain framework [1]. In Sec. 3, we discuss heuristics implemented to improve the team's defense. In Sec. 4, we show the new stamina model we developed. Sec. 5 explains how we enhanced the goalie behavior. Sec. 6 details our latest attempt on understanding and improving the lower level aspects of the team's code. Sec. 7 presents

a log analyzer tool we developed. Sec. 8 presents a modification to the basic defense formation and our perspectives regarding this field. Sec. 9 gives more details about the wiki we made to help new teams and our recent additions to it. Finally, Sec. 10 concludes and shares our ideas for future work.

2 Improvements to Action Chain Search Framework

Agent2d has a built-in framework to search for a sequence of actions (action chain) online as described in [1]. In a previous TDP [4], we described our work in improving this framework, which consisted mainly in adding new features to the sample action chain evaluator and using optimization techniques, in specific Particle Swarm Optimization (PSO) [7], to optimize the weights given to each feature.

Continuing on this track, we added a new feature calculated as following: take straight lines from the player to all teammates; then, for each line compute the distance from it to the nearest opponent; finally, sum all the distances computed. After optimizing with this feature on, we observed that the player holding the ball was positioning itself better for pass. On the other hand, the feature was also making the team less offensive, spoiling its attack.

2.1 Choosing between an offensive or a defensive strategy

Later, we developed a neural network to predict the probability of scoring a goal given a game state (considering the positions of all players and the ball). This mechanism was then used to choose between deactivate (defensive strategy) or activate (offensive) the described feature when the probability of scoring a goal was low or high, respectively. After manually tweaking the thresholds, the team with the neural network showed a better performance.

2.2 Dividing the attack in subtasks

Moving further in that direction, we assumed that the agents adaptability and the optimization procedure would benefit from dividing the attack in subtasks. Our idea was to let the agent to think about a shorter term subtask (e.g. take the ball away from our defensive area or move the ball to the opponent side of the field) instead of always thinking about scoring a goal.

A particular way of doing this division in subtasks is to think in terms of which region of the field the ball is. Thus, we divided the field in 8 regions and optimized a different action chain evaluator for each one. With this approach, it was possible to achieve good results training against specific teams, however our previous strategy that uses a neural network to do a simple strategy switch achieves an overall better robustness against differences on the defense policy of each team. Nevertheless, we still believe that a more complex way of choosing different strategies depending on game situation can lead to better team performance and we expect to do more experimentation in this direction.

2.3 Changing the Field Evaluator Depending on the Opponent

In the LARC 2015, our team was not being successful in offensive plays against the RobotBulls team, although playing very good against other opponents. Watching the games between competition days, we realized that their defense was particularly good against our offense due to positioning, saving almost every play attempt. We then created a new set of Field Evaluator parameters based on RobotBulls' defensive positioning, so our team would use the new parameters against them and the standard parameters against the others. The changes were successful and we beat them in the next games, including the finals.

2.4 Controlling the Overall Team Behavior Through the Field Evaluator

During Robocup 2016, we noticed that our offensive players were moving too much to the sides of the field, even though it was clear that playing in the center would be a better move. We added a new feature to the Field Evaluator that would force the player to move to the center if the spaces between the defenders in that area were larger than in the sides. It was given a better score if the player decided to move to the side of the closest opponent that had a larger space between players.

In the next games at the competition, our team was more offensive and developed better opportunities by the center of the field when going to the sides did not seem like a good move. Later on, we tested it against the base team and compared with the results of our previous team and the statistics were quite similar.

We developed this feature for the specific case where our opponent had mainly four defenders, so in the general case it did not performed as good as in the one we planned it for. We intend to develop a more general version of this idea in future works and analyze its efficiency.

3 Defense Heuristics

In order to improve the team's defense we preferred to use heuristics based on statistics and domain knowledge instead of formal AI.

Basically the defender has to decide whether or not to tighten his marking. The advantage of doing that is to decrease the opponent's space and create more chances to tackle. The problem arises when the forward gets to dribble around the defender.

To make this decision properly we analyzed a lot of parameters including the position of the ball in the field, the danger of the play, the position of the offside trap, the distance between the defender and the ball, etc.

Every time a new set of conditions were added to the decision-making process, the team was tested against its old version. The tests consisted of 50 matches. After that it was possible to decide if the modifications were worth it by plotting the mean goal difference against the number of matches. This plot usually converges when the team is actually improving.

After a lot of hard working the team was able to maintain an average goal balance of 0.5 in 100 matches against its version without the modifications, which is a pretty good result.

4 New Stamina Model

The default agent2d stamina model is too simple. We analyzed this model and improved it including new conditions, specially for the defenders (that used to save needlessly too much stamina). The new model allows defenders to dash in dangerous situations (preventing through balls for example) without compromising their stamina capacity.

5 Improvements to the Goalie's Behavior

During the CBR 2013 competition, our team was consistently losing to team UAI Soccer, due to last minute changes they made to their team during the competition. Once in CBR one may submit new versions of the code every day, we decided to improve our defense by changing the goalkeeper behavior, in an attempt to turn the game and win the competition.

To improve our goalkeeper, we watched several games from team MarliK during RoboCup 2012. Being an agent2d based team, MarliK released their source code for that competition [15] so we also went through their code to see what was going on. We decided to implement a similar system to our goalkeeper. The basic principle is that the goalkeeper now moves more freely in the defensive field when it is safe to do so. As a consequence, several offensive algorithms developed by other teams can fail, because they were devised considering that the goalkeeper always stays near to the goal. We noticed that the MarliK goalie has a very good offensive positioning, but does a poor job participating in back plays by receiving the ball outside the goal area and then passing it.

To fix this, we made the goalkeeper execute the Agent2D action chain when he safely receives the ball outside the goal area. Also, we modified the goalie behavior inside the goal area by mixing the agent2d code and the MarliK goalie code, while also implementing some heuristics that we devised

6 Improvements to the World Model

Agent2d is founded upon the base library librcsc [11], which contains low level implementations that models the team's viewing of the game. In 2015, after some extensive analysis on the library, we have greatly increased our understanding on how the team behaves and used this reasoning to add new parameters and variables to better model the world. For example, the agent2d's intercept model is fairly simple as it does not account for the possibility of tackle. We have been dealing with this issue for some time. Unfortunately, we have not extensively tested the results of such work, but understanding the key factors that make up agent2d has taken us one step further to perfecting the team's world model.

7 Log Analyzer Tool

Initially, we developed a log analyzer tool to solely support our formation detection system, as described in our 2013 TDP [5]. Later, developing a GUI for the tool to help us debug, we realized that the tool itself had potential for other uses. For example, drawing all ball positions from a given match, we could visually see the paths usually followed by a team or where it tends to stay with the ball (see Fig. 1). We believe that with this kind of visual tool, a human can analyze logs much faster and precisely than having to watch the whole match.

8 Formation Modification

The formation of the team was modified in order to improve the defensive behavior. After analyzing several matches against different opponents, it was possible to detect certain weaknesses in the defense formation that resulted in opponent's scoring. Therefore, using the program *Fedit*[13] the team was able to improve the defense positioning. The resulting modifications introduced a formation that changed between four or five defenders, strengthening our defense.

9 Wiki Project

In 2011, when ITAndroids was reestablished, it was very difficult to find information about Soccer 2D and how to start a new team, especially in the brazilian scenario.

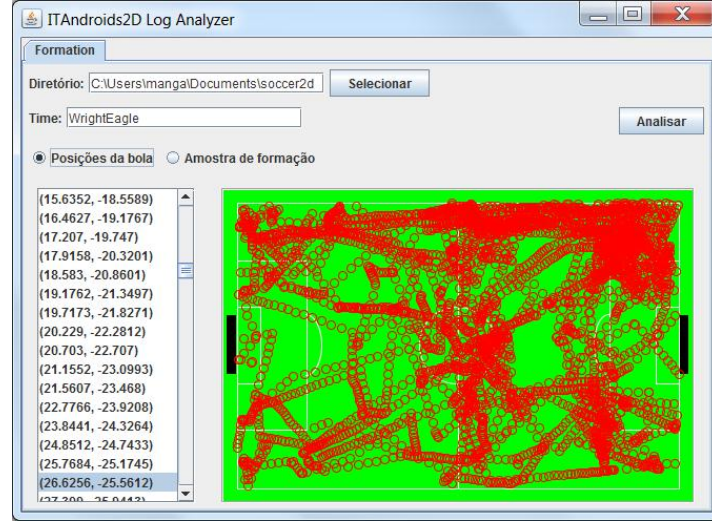


Fig. 1. ITAndroids 2D log analyzer tool.

Besides that, one of the biggest obstacles to the long term development of the team is the transfer of knowledge between senior and junior members. Often, new members would waste dozen of hours performing basic introductory tasks like tweaking the team formation and configuring team files, this made the incorporation of new members very time consuming.

The internet is lacking in tutorials related to Soccer 2D, what makes it more difficult to introduce the competition to new students or to introduce it in universities new to it, so we decided to make our own wiki to fill that gap. Currently, our wiki is only in Portuguese (our native language) and covers only very basic topics from the installation and configuration of the necessary tools, to the main aspects of agent2D. We plan to translate it to English in the near future.

In national and latin american competitions, we have received a very positive feedback concerning the wiki. Many new teams have reported that they have used our tutorials and that it has helped their development.

Furthermore, having a well written detailed documentation of code is extremely important to not lose knowledge inside the category throughout time, a worrying issue that has been happening recently. Making drastic modifications in the team also require a deep understanding of agent 2d's classes and how they are used. All of this can only be possible with documentation usage, due to the code's extension and the complexity of such projects.

9.1 Rewriting and expansion of "Loop Principal" content

We have moved the work done in the Wiki up to now in a separated part. That's because we are looking for a new way of writing its content in a more organized manner: with the use of summaries, hierarchical organization of wiki sheets, go-back link options and many other features. There is no consense of what is best to orient our code descriptions(separate by classes, separate by folders, separate by purpose...), thus it's very likely that the new Wiki will have multiples orientations.

The team's study of the code started from the *Main* and focused only in *Player* virtual classes(Couch and Trainer left aside). We have analysed the very beginning up to the Decision-Making layer to understand how is each level of intelligence grouped:

from low-level server connection until high-level *ActionChainGraph* operation. With this study we were able to better identify when and how each *Soccer Agent* is created, how do matches start, stop and end, how server messages are received, sent and interpreted, and how does the decided action to be done by the agent affect the tables' data.

9.2 Roles and the ActionImpl Method

From the previous study, some classes were divided by having crucial role: *PlayerAgent* and *SamplePlayer*, specially the last one, that has the virtual override of the *ActionImpl* method, responsible for the decision-making. The *ActionImpl* activates and updates the ActionChain, which does all of the decision-making, creates a Role for the player and then calls the execution of the action decided by the ActionChain, which varies according to the Role given.

The possible Roles for a player to have are: center-back, center-forward, defensive-half, goalie, keepaway-keeper, keepaway-taker, offensive-half, side-back, side-forward and side-half. We focused mainly in the study of the Offensive Half Role.

9.3 Deeper understandings of the Action Chain Graph algorithm

Motivated by the lack of offensive behavior in strongly closed defensive formations, a deeper study of the decision-making process was made. After the *execute* method is called in ActionImpl there are two possible routes: either the agent may interact with the ball or not. Methods are called depending of the situation which basically look at the ActionChainGraph, already updated to the world's situation, and take the sequence of actions to be made.

This comprehend the options *Shoot*, *Pass*, *Dribble*, *Hold* and *Move*. It was observed the lack of Shoots as the first action in ActionChain result. This study was very important to understand the ActionChain in the matter of its data structure structuring and how the *BestFirstSearch* is implemented within the *Decision Tree*.

10 Conclusions and Future Work

This paper presented the most recent efforts of team ITAndroids 2D.

Our current code is based on agent2d. We have been experimenting with many formal AI techniques: heuristic-guided search, Particle Swarm Optimization [7], Neural Networks, K-Means Clustering [8] etc. Furthermore, there is a lot of improvement in our team achieved by specific hand-coded heuristics. At the same time, overall strategy planning has been slowly growing, with recent gatherings in formation modification knowledge and new perspectives of modelling the team behavior by creating new parameters in the Field Evaluator.

The Wiki Project is now being visualized as the first step towards a heavier documentation of our code, although its limitations are significant and well known. Alongside this important task, the actual perspectives of our team are deepening and optimizing the modelling of our current field evaluators, studying the possibility of creating an adaptative formation and, yet not mentioned in the previous work, the search for new custom cooperative behavior, such as triangulation passes and better long distance infiltration.

Acknowledgements

We would like to acknowledge the RoboCup community for sharing their developments and ideas. Specially, we would like to acknowledge Hidehisa Akiyama for agent2d [10],

librcsc [11], soccerwindow2 [12] and fedit2 [13] softwares. Also, we would like to acknowledge Luís Paulo Reis for his PhD thesis [9] and UvA Trilearn [14] for their base code; both played a great role in introducing our team to the league. Current team members are also grateful to previous ones who have made great contributions to ITAndroids 2D. Finally, we thank the company RADIX for sponsoring our team.

References

1. Akiyama, H., Shimora, H.: HELIOS2010 Team Description (2010)
2. Norouzitallab, M., Javari, A., Noroozi, A., Salehizadeh, S.M.A., Meshgi, K.: Nemesis Team Description 2010 (2010)
3. Bai, A., Zhang, H., Lu, G., Jiang, M., Chen, X.: WrightEagle 2D Soccer Simulation Team Description 2012 (2012)
4. Mello, F., Ramos, L., Maximo, M., Ferreira, R., Moura, V.: ITAndroids 2D Team Description 2012 (2012)
5. Mello, F., Ramos, L., Maximo, M., Ferreira, R., Moura, V.: ITAndroids 2D Team Description 2013 (2013)
6. Lau, N., Reis, L.P., Mota, L., Almeida, F.: FC Portugal 2D Simulation: Team Description Paper (2011)
7. Eberhart, R.C.; Kennedy, J.: A New Optimizer Using Particle Swarm Theory. In: Sixth International Symposium on Micro Machine and Human Science (MHS'95). Nagoya, Japan (1995)
8. Alsabti, K., Ranka, S., Singh, V.: An Efficient K-Means Clustering Algorithm. Electrical Engineering and Computer Science, Paper 43 (1997)
9. Reis, L.P.: Coordenação em Sistemas Multi-Agente: Aplicações na Gestão Universitária e Futebol Robótico, PhD Thesis, FEUP (2003)
10. agent2d-3.1.1, 2012, online, available at: <http://pt.sourceforge.jp/projects/rctools/downloads/51943/agent2d-3.1.0.tar.gz/>, consulted on February 2012.
11. librcsc-4.1.0, 2011, online, available at: <http://pt.sourceforge.jp/projects/rctools/downloads/51941/librcsc-4.1.0.tar.gz/>, consulted on February 2012.
12. soccerwindow2-5.1.0, 2011, online, available at: <http://pt.sourceforge.jp/projects/rctools/downloads/51942/soccerwindow2-5.1.0.tar.gz/>, consulted on February 2012.
13. fedit2-0.0.0, 2010, online, available at: <http://pt.sourceforge.jp/projects/rctools/downloads/48791/fedit2-0.0.0.tar.gz/>, consulted on February 2012.
14. UvA Trilearn 2003, 2005, online, available at: http://staff.science.uva.nl/~jellekok/RoboCup/2003/index_en.html, consulted on February 2012.
15. MarliK source code, 2012, online, available at: <http://www.socsim.robocup.org/files/2D/binary/RoboCup2012>, consulted on February 2014.