ITAndroids 2D Soccer Simulation Team Description Paper 2021

Diego T. M. Fidalgo, Felipe V. Coimbra, Gustavo F. Gottschild, Jony dos S. S. Filho, Vinicius F. Almeida, and Marcos R. O. A. Maximo

Aeronautics Institute of Technology,
São José dos Campos, São Paulo, Brazil
mmaximo@ita.br,{felipecoimbra97,jonysalgadofilho,gustaferraresi189,diego.fidalgo2,vinifreitas.d.a}@gmail.com
itandroids-soccer2d@googlegroups.com
http://www.itandroids.com.br/

Abstract ITAndroids 2D Soccer Simulation team is composed by undergraduate students of Aeronautics Institute of Technology. The team is currently one of the strongest teams in Brazil, having won 1st place 4 times consecutively from 2012 to 2015, Vice Champion in 2018 and was the Champion in 2019 Latin American Competition. Moreover, the team has qualified for the last seven editions of RoboCup, having participated of five. This paper describes some of our advances in 2019 and our plans for 2020.

1 Introduction

ITAndroids is a competitive robotics 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, IEEE Very Small Size and RoboCup Small Size league.

Our Soccer 2D's team, ITAndroids2D, has continuously participated in Latin American Robotics Competition (LARC) and Brazilian Robotics Competition (CBR) since 2011. Moreover, ITAndroids2D competed in RoboCup in 2012, 2013, 2015, 2016, 2017, 2018 and 2019. The team also qualified for RoboCup 2014, but unfortunately it was not able to attend to the competition. Our results in these competitions are represented in the Table 1.

Lack of continuation and documentation of the project and the spreading of the team towards other fields made ITAndroids2D slow down its improvements. This can be seen from the leagues results from 2015 to 2017. However, ITAndroids 2D recovered in 2017, after a complete restructuring of the project [3]. As a result, we won 9th place at RoboCup2018, our best absolute place in the competition.

In this paper, we describe some advancements during 2019 and early 2020: a reoptimization of Field Evaluator Weights with PSO (Section 2) and an optimization of goalkeeper for penalty using Deep Reinforcement Learning (DRL) (Section 3). In Section 4 we describe our future work and the conclusion of this report.

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

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

2 Reoptimization of Field Evaluator Weights with PSO

Action Chain is ITAndroids2D's default decision making algorithm for players that have possession of the ball [2]. It combines a dynamically created tree of exploring states with a greedy best first search to find the root-to-leaf paths that end in the best states possible. The desireness score of a state is calculated by a component of the algorithm called Field Evaluator.

Action Chain has fundamental importance for the performance of the team and is extremely sensible to the Field Evaluator modelling. State evaluation in ITAndroids2D consists of a combination of heuristics regarding various situational aspects, for example, ball's position and its distance to the opponent's goal.

These heuristics are highly dependent of parameters, the Field Evaluator Weights, that balance their importance in the calculated state score. These weights have been optimized in the past using Particle Swarm Optimization (PSO) [3]. It was decided to reoptimize these weights using this same algorithm once the league has considerably evolved since the last work.

PSO combines exploration and exploitation with its evolutionary model of search together with stochastic update rules. It is also naturally highly computationally parallelizable, making good use of cluster resources without adding any extra complexity to the algorithm.

2.1 Results

PSO can be very sensitive to variance while evaluating particles. Because of this, the optimization was made against a single team, the RobotBulls release of 2017. The algorithm used a population of 30 particles and each evaluation consisted of a batch of 10 games with the metric chosen composed by net goal balance and ball possession.

The exact cost function used is shown in equation 1. t_1 is the mean goal balance and t_2 is the mean ball possession of the 10 matches. Note that this equation makes goal balance twice more relevant to the total score than ball possession.

The evolution of the optimization is shown in the Figure 1. The algorithm clearly tends to converge as can be seen from the monotonic increase of the cognitive best scores towards the global best and the decrease of its variance.

$$f(t_1, t_2) = \frac{\tanh(0.33 \cdot t_1) + 1}{3} + \frac{t_2}{3} \tag{1}$$

Figure 1: Score evolution of particles during optimization. The 85 iterations were not enough to obtain full convergence but sufficient to acquire handful results.

These new results are almost as good as our best ITAndroids2D version against RobotBulls, which currently holds the statistics of 42.3% of victories, 21.2% of draws and 36.5% of losses for 500 games.

However, our results show that PSO may not be the best evolutionary strategy option for this type of black box optimization. Figure 1 shows that the global best score barely changes from the first iteration tentative. Since the particles are randomly initiated this is equivalent to saying that random search is potentially as efficient as PSO, which is absurd.

The reason of this observation is that the metric stablished in equation 1 depends on random variables t_1 and t_2 that have very high variance for 10-sized samples. This means score evaluation can be easily misevaluated as being much higher than it really is. Once PSO uses infinitely long term memory of previous particles evaluations, these mistakes propagates and severely interfere in the algorithm's outcome.

Table 2: Performance of ITAndroids2018 against RobotBulls2017

	Before optimization	After optimization
Wins	33.3%	39.0%
Draws	22.3%	20.5%
Losses	44.4%	40.5%

3 Optimization of the goalkeeper for penalty

Among the many benefits Deep Reinforcement Learning (DRL) may bring to ITAndroids2D, the optimization of the goalkeeper for penalties deserves note because the penalty event, in addition to its decisive character in a draw, uses a single player from each team, who are under movement rules much simpler and more constricted than those found in other situations of a match [10]. All these factors make the penalty one of the easiest soccer events to optimize, something very interesting for a team not yet used to DRL.

There are many state-of-the-art ways to implement such optimization, like Q-learning with Deep Neural Networks, Proximal Policy Optimization (PPO) and Soft Actor Critic (SAC). In this case, PPO was chosen to perform the task because of computational complexity and implementation issues [10].

One of the most important components of the training, the attacker must be a high quality one to ensure that the keeper agent really learns during the DRL. If it is too weak and simple, the agent will learn little on training, while one far superior to the keeper will also compromise the learning process since the agent will face failure almost all time, resulting in a policy with high and stable entropy.

To improve the agent, five experiments took place: the experimental conditions started rather simple, and each experiment had its features improved and adapted as the work progressed, in order to provide a better learning configuration. In each experimental test, observation space, action space and rewarding model were adjusted seeking the best learning environment for the agent. In addition to the default high-level and low-level feature sets provided by the Half Field Offense subtask of rcssserver, a third one, named Custom Feature Set was conceived for interest for using members from both sets appeared in the last performed experiments.

The first experiment used the high-level feature set, a continuous action space composed only by the action MOVE_TO and a reward model with rewards for

catching the ball and getting closer to it, and a penalty for suffering a goal. After 20M of training steps and several repetitions, policies with rising entropy were obtained, requiring a change in the next experiment.

The second one was performed under the same observation space and reward model as the first, and, aiming to fix the increasing policy entropy problem, its action space was substituted by a discrete one, composed by a discrete set of directions. As shown by the results the experiment produced, even though this time the entropy dropped, it also quickly stagnated, something demanding more severe changes.

In the third experiment, the custom feature set started to be used, in order to make rewards less sparse. The action space was also modified to one discrete as in the former experiment, but swapping the MOVE_TO action by the actions DASH and TURN, so the agent would always keep the ball at sight. Moreover, the reward model was updated by the addition of reward for staying at the bisector angle of the triangle formed by the ball and the two goal posts and penalties for not looking at the ball and for leaving the penalty area. Rewards for proximity to the ball were all removed so the agent cannot be biased. The aftermath of such configuration was, after 5M steps, a goalkeeper that, although capable of positioning itself well, was not able to intercept well done kicks. That is, in spite of the significant improvement of results, further changes were needed.

The fourth experiment used the same observation space, action space and reward methodology as the third one, and also added the intercept behavior. When an invasion of the penalty box or a kick were detected, DRL was put aside and the interception action was called, because, when the attacker ceases to have ball possession, just chasing the ball is the best initiative to be taken. In this experimental arrangement the agent finally started to have a significant success rate. As shown in Figure 2, 5M training steps afforded around 24 per cent of catches.

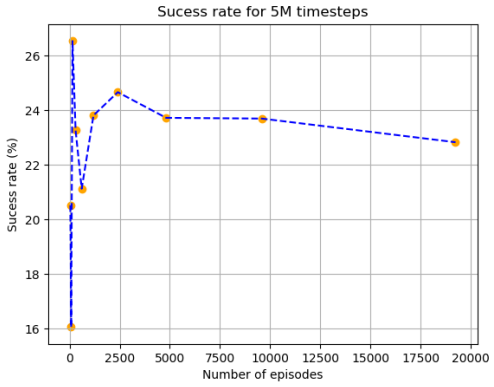

Figure 2: Success rate of the keeper in function of the number of episodes performed after 5M training steps, in the fourth experiment.

The last experimental disposition used the same settings as the fourth one, except by the removal of minor rewards. That is, the little rewards provided for proximity to the bisector angle were utterly removed, in order to keep the agent focused on its main goal: catching balls and not letting the adversary score. For the fifth experimental arrange generated the best results, as shown in Table 1, it was used two times: the first and the second one runned for 15M and 30M learning steps, respectively.

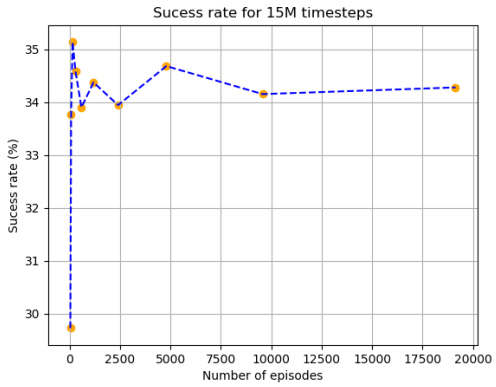

Figure 3: Success rate of the keeper in function of the number of episodes performed after 15M training steps, in the fifth experiment.

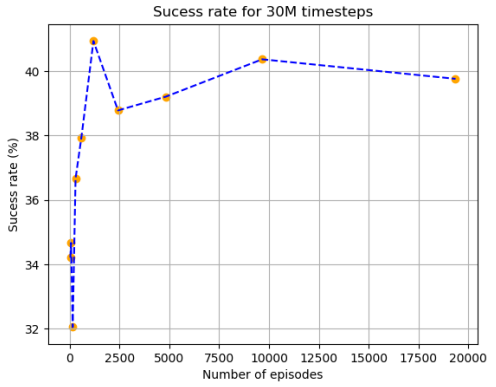

Figure 4: Success rate of the keeper in function of the number of episodes performed after 30M training steps, in the fifth experiment.

As shown in Figure 5, after 30M learning steps performed in the last experiment, the goalkeeper was able to achieve a success fraction around 20 per cent higher than that of Soccer 2D Base Agent, an excellent result since the Base Agent used for comparisons is significantly good. Besides the progress already made, it is possible to turn such behavioral improvements even more remarkable and interesting by the use of different types of attacker, a change which would make the optimizations far more reliable in a real match scenario.

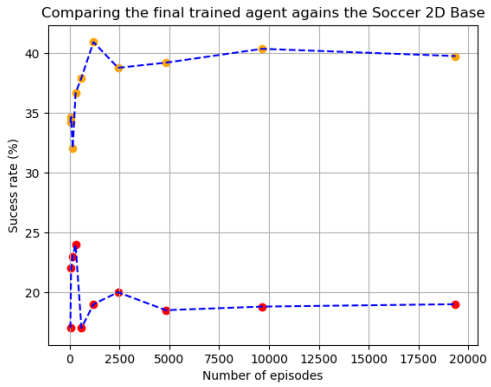

Figure 5: Comparing the agent trained for 30M learning steps in experiment 5 against the Soccer 2D Base Agent. The upper graphic belongs to the final trained agent from experiment 5.

This work demonstrates the power reward engineering has to speed up training and improve outcomes. Moreover, experiment 5 shows DRL may not just find good behaviors from zero, but also optimize existing ones.

4 Conclusions and Future Work

This paper presented the most recent efforts of team ITAndroids 2D. Among the several improvements made by the team since the last RoboCup, the most remarkable ones are the optimization, using PSO, of the field evaluator weights and the optimization, using a reasonable number of attacker types, of the goalkeeper with DRL.

The former development used, for the adjustment of the Action Chain parameters to better fit the heuristics implemented, an optimization technique which combines exploration with exploitation and comprises high computational parallelization and stochastic update rules. But the results achieved show there may be evolutionary strategy options better suited for the goals aimed than PSO, something explained by error propagation issues.

In the latter improvement, DRL was implemented with PPO and five experiments were performed, seeking to find the best learning environment. After that, the success rate of the goalkeeper after the last experiment was excellent, being around 20 per cent higher than that of the significantly good Soccer 2D Base Agent. Furthermore, since most of the attackers of RoboCup2020 Soccer Simulation 2D League are controlled using state-of-the-art methods, this optimization can be even further developed, using different kinds of attackers. Such improvement brings the keeper agent optimization to a much higher level of robustness.

5 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 [1], librcsc [4], soccerwindow2 [7] and fedit2 [6] software. Current team members are also grateful to previous ones who have made great contributions to ITAndroids 2D. Finally, we thank our sponsors Altium, Cenic, Intel, ITAEx, Mathworks, Metinjo, Micropress, Polimold, Rapid, Solidworks, ST Microelectronics, Wildlife Studios, and Virtual Pyxis. We also acknowledge Mathworks (MATLAB), Atlassian (Bitbucket) and JetBrains (CLion) for providing access to high quality software.

References

1. agent2d-3.1.1, 2012, online, avaliable at: http://pt.sourceforge.jp/projects/rctools/downloads/51943/agen t2d-3.1.0.tar.gz/, consulted on December 2018.
2. Mello, F., Ramos, L., Maximo, M., Ferreira, R., Moura, V.: ITAndroids 2D Team Description 2012 (2012)
3. Lema L., Coimbra F.: ITAndroids 2D Team Description 2018 (2018)
4. librcsc-4.1.0, 2011, online, avaliable at: http://pt.sourceforge.jp/projects/rctools /downloads/51941/librcsc-4.1.0.tar.gz/, consulted on December 2018.
5. Muzio A. F. V. (2018). Curriculum-based deep reinforcement learning applied to humanoid robots (Master's Thesis). Retrieved from Aeronautics Institute of Technology Digital Library http://www.bdita.bibl.ita.br/.
6. fedit2-0.0.1, 2017, online, avaliable at: https://osdn.net/projects/rctools/downloads /68531/fedit2-0.0.1.tar.gz/, consulted on December 2018.
7. soccerwindow2-5.1.0, 2011, online, avaliable at: http://pt.sourceforge.jp/projects/rctools/ downloads/519 42/soccerwindow2-5.1.0.tar.gz/, consulted on December 2018.
8. Melo L. C. (2018). A deep reinforcement learning method for humanoid kick motion (Bachelor's Thesis). Retrieved from Aeronautics Institute of Technology Digital Library http://www.bdita.bibl.ita.br/.
9. Aguiar L. G. G. A. (2018). Deep reinforcement learning based kick control for a simulated humanoid robot (Bachelor's Thesis). Retrieved from Aeronautics Institute of Technology Digital Library http://www.bdita.bibl.ita.br/.
10. Figueira A. M. S. R. (2019). Deep Reinforcement Learning applied to a Soccer 2D Keeper Agent (Bachelor's Thesis). Retrieved from Aeronautics Institute of Technology Digital Library http://www.bdita.bibl.ita.br/.