

ITAndroids 2D Soccer Simulation Team Description 2012

Fábio Mello, Luiz Ramos, Marcos Maximo,
Rodrigo Ferreira, and Victor Moura

Technological Institute of Aeronautics,
São José dos Campos, São Paulo, Brazil
`fabio.mello50@yahoo.com.br`

`{luizmramos,maximo.marcos,roim.rodrigo,vicrmoura}@gmail.com`
`itandroids-soccer2d@googlegroups.com`
`http://www.itandroids.com/`

Abstract. ITAndroids 2D soccer simulation team was reestablished in 2011 by a new group of undergraduate students of Technological Institute of Aeronautics after a long time of inactivity. In the past, our team participated in several robotic competitions, including RoboCup in Soccer Simulation 3D, Rescue Simulation Agent and Rescue Virtual Robot Simulation in 2006. Our Soccer Simulation 2D team won the Latin American championship in 2005. This paper briefly introduces our recent research focus, including a Particle Swarm Optimization applied on evaluation processes, and a new approach, inspired by real soccer matches, that uses the *point to* model to make pass requests.

1 Introduction

ITAndroids was created in 2005 by Jackson Paul Matsuura, then a graduate student at Technological Institute of Aeronautics (ITA), as ITA's research group of robotics. In its first years of existence, the group consolidated itself in Brazil and Latin America, participating in several categories and competitions, including the RoboCup. Unfortunately, due to a personal decision of Matsuura to focus his efforts on building up the whole Brazilian robotics competitions scenario, the team faded with time and dismantled.

By mid-2011, a group of undergraduate students restarted the ITAndroids 2D soccer simulation team, intending to participate in Brazilian Robotics Competition (CBR) 2011. Advised by Matsuura, the first attempt of the new group was to build a team based on UvA Trilearn base team [10] using a Hierarchical Finite State Machine (HFSM) [1] architecture for modeling agents behavior. The team strategy during this period was based on hand-coded heuristics using UvA's built-in routines.

However, due to the short development time and the lack of experience of team members in the Soccer Simulation 2D environment, the team had a poor performance at CBR 2011.

Considering these results, the group decided to change its base team to agent2d [11]. Since we were intending to participate in RoboCup 2012 2D Soccer Simulation League with a competitive team, we have tried to focus on points of improvement within agent2d's code that would increase the team's performance without requiring too much development time. Also, we have based our development as much as possible on the structures already provided by the agent2d framework, in order to save time and make our code clear to other teams. Moreover, now we are trying to apply formal AI techniques instead of relying only on hand-coded heuristics.

This paper describes our development efforts using agent2d. Section 2 explains our attempts to improve agent2d sample action chain evaluator [6], while Section 3 introduces our proposed use of the *point to* action.

2 Improvements on the Action Chain Search Framework

Agent2d has a built-in a framework to search a sequence of action-state pairs (an action chain) on-line as described in [6]. Action chains are then evaluated by giving a score to the final state predicted after the execution of the action sequences. [6] mentions the difficulty to create an effective action chain evaluator. Also, in our tests, we have noticed that small changes in the sample action chain evaluator provided by agent2d code yields big differences in the team's performance.

The sample action chain evaluator has two phases of evaluation. Firstly, it searches extreme situations: desirable or undesirable, giving high or low scores, respectively. Secondly, it evaluates the state using the following features: X coordinate of the ball position, the ball distance to opponents' goal, if there's a free situation near offside line and if the agent still holds the ball.

Trying to improve the agent dribble skill, we introduced the sum of the inverses of the distances to opponents players as a feature to the second phase. Also, we tried to change the weights given to each feature. Since manually tweaking these weights has proven to be a difficult task, we looked for an optimization technique to apply.

Inspired by the approach taken in [7], we used Particle Swarm Optimization (PSO) [2] to optimize the parameters. As a measure of quality, we ran SoccerServer simulation against agent2d during one half (to cut the algorithm execution time by half) for each position and calculated the sum between goal balance and ball possession (as a number between 0 and 1). With this measure of quality, the particle fitness is dominated by goal balance and the ball possession acts as a tie-breaking method.

The PSO algorithm was executed in the coach since it is the only agent that always has a complete knowledge of the field at a given time. To measure ball possession, we used the approximation that a team has the ball if it has the closest player to the ball.

Tests using the modified action chain evaluator with the position of the best particle found in the search space have shown improvements in the team’s performance.

Due to the simulation time (3000 cycles or 5 minutes) required to evaluate a particle position, we were very limited in the number of particles and iterations we could afford to run using a single computer. We expect to develop a framework to parallelize our particles evaluations using a computers network for future optimizations.

3 Pass Request

Even though agent2d’s implementation of the Action Chain yields good offensive results, it has some flaws. For instance, it delegates too much of the decision making process to the agent carrying the ball, so decisions vary little, e.g. the dribbler almost always advances through a side wing. In real soccer matches, diverse goal opportunities arise when a player ask for a pass, usually pointing to wherever he thinks is a good place to receive the ball.

3.1 Using the *point to* Model

Analyzing the CBR 2011 final match [4], we observed the *point to* model applied by WARTHOGSIM to request the ball, in order to create promising situations. Further investigation revealed that many teams use this approach. However, the current usage of the command is still simple and limited. Our objective was to apply it in pass requests similarly to real soccer, simultaneously improving the Action Chain model. This research produced the following.

Offensive agents wait until they should *point to* an empty position. So, the dribbler decide whether each of the requested passes is a good decision or not, based on his own world model. Then, he notifies the receiver through another *point to* command and makes the pass. Finally, the receiver run towards an optimal interception point. This is exemplified in Fig. 1. In the case of no pass being considered good enough, the Action Chain is invoked normally.

3.2 Pointing to a Good Region

In order to determine when the candidate receivers should use the *point to* command, we built a Voronoi Diagram based on the position of the agents, with the methods provided by librcsc. If there is enough free space inside the agent’s cell, he points for a pass near the cell’s edge, as shown in Fig. 2.

3.3 Dribbler’s Decision and Feedback

When the dribbler identifies at least one player pointing, he decides whether he should pass or not. This is done applying a modified agent2D pass checker on points inside the receiver’s Voronoi Cell, and selecting a point where the receiver

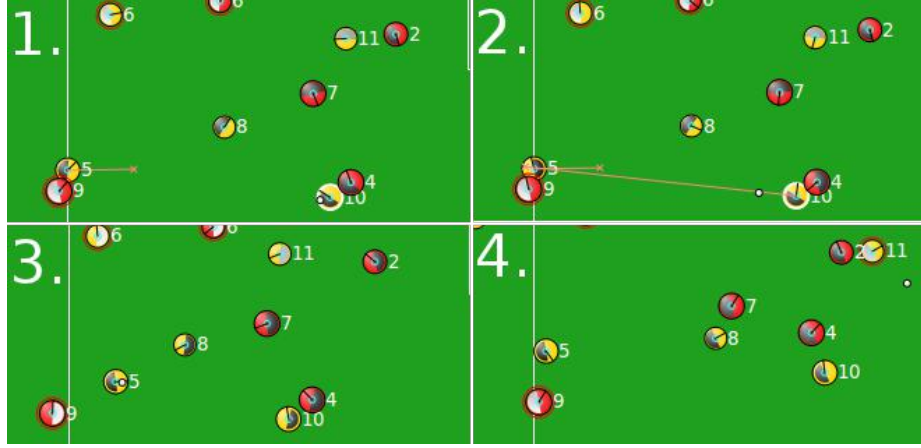


Fig. 1: Frame 1 shows player 5 requesting a pass. On Frame 2, player 10 passes, and uses his own *point to* command as a feedback. Then player 5 intercepts the ball, this results in frame 4 after a few cycles. Notice that going from Frame 1 to 4 would be improbable without our implementation, as opponent 7 would easily intercept a pass to player 8, and opponent 4 was ready to tackle.

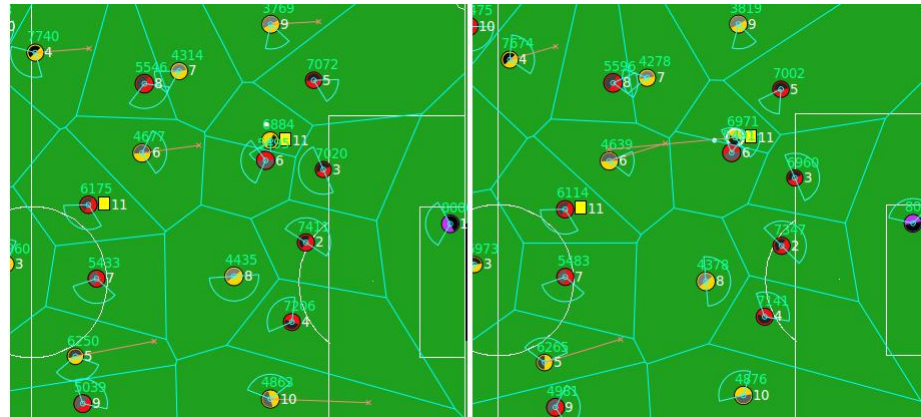


Fig. 2: Several players using the *point to* command inside their Voronoi Cells. On the right, the same region after 2 cycles, when player 11 decides to pass the ball to player 6.

can safely intercept the ball. This model assumes that the receiver will try to intercept the ball running to the best interception point possible, right after the pass. This assumption is acceptable due to our pass receiving technique.

3.4 Receiving the Pass

The receiver notices the dribbler's *point to* and starts running towards the interception point immediately. To make the interception, we applied a common prediction model present on several teams [8][9]. We believe that our specific implementation yields better results, as the receiver starts running as soon as the pass is done, which is an assumption of this model.

4 Conclusions and Future Work

This paper presented the recent efforts of ITAndroids' team in Soccer Simulation 2D. The team was reestablished in 2011 with a new group after a long period of inactivity. Unfortunately, the expertise of the old team was lost, except for the knowledge of former member Jackson Matsuura, who has given some advices to the new group. Our current team is based on agent2d and the main results that make our team different from the base team are a better action chain evaluator and a pass request mechanism, which proved to be enough to make a competitive team in today's standards.

Currently, our group is experimenting with the use of Genetic Algorithms to optimize ball interception parameters and Reinforcement Learning with SARSA algorithm to improve marking skill, but we couldn't get significant results yet. For immediate future work, we expect to develop a framework to exploit parallelism in optimization algorithms, such as Genetic Algorithm and Particle Swarm Optimizations. Our idea is to distribute the fitness evaluations that are based on SoccerServer simulations between computers connected in a network. This will reduce the noise influence present on the Server.

Acknowledgements. We would like to acknowledge Luís Paulo Reis for his PhD thesis [3], which has been our main introduction to Multi-Agent simulations (with the advantage of being written in our native language). Also, we would like to acknowledge UvA Trilearn [10], whose base team has given us a didactic introduction of building a 2D soccer simulation team, and Hidehisa Akiyama for agent2d [11], librcsc [12], and soccerwindow2 [13], which has permitted us to have a team able to compete in the current competition level.

References

1. Backland, M.: Programming Game AI by Example. Wordware Publishing, Inc., Plano, Texas (2005)

2. 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)
3. Reis, L.P.: Coordenação em Sistemas Multi-Agente: Aplicações na Gestão Universitária e Futebol Robótico, PhD Thesis, FEUP (2003)
4. CBR 2011 Final Match, online, available at: <http://www.youtube.com/watch?v=XDqrSL4mc-U> consulted on March 2011.
5. RoboCup 2011 Final Ranking, online, available at: <http://sourceforge.net/apps/mediawiki/sserver/index.php?title=RoboCup2011/Competition>, consulted on February 2011.
6. Akiyama, H., Shimora, H.: HELIOS2010 Team Description. (2010)
7. Norouzitallab, M., Javari, A., Noroozi, A., Salehizadeh, S.M.A., Meshgi, K.: Nemesis Team Description 2010. (2010).
8. Silva, B., Frias, D., Eloi, I., Grimaldo, J., Simões, M.A.C.: Bahia2D 2010: Team Description. (2010)
9. Tavafi, A., Nozari, N., Vatani, R., Yousefi, M.R., Rahmatinia, S., Pirdeyr, P.: MarliK 2011 Soccer 2D Simulation Team Description Paper. (2011)
10. UvA Trilearn 2003, 2005, online, available at: http://staff.science.uva.nl/~jellekok/RoboCup/2003/index_en.html, consulted on February 2011.
11. agent2d-3.1.0, 2011, online, available at: <http://pt.sourceforge.jp/projects/rctools/downloads/51943/agent2d-3.1.0.tar.gz/>, consulted on February 2011.
12. librcsc-4.1.0, 2011, online, available at: <http://sourceforge.jp/projects/rctools/downloads/51941/librcsc-4.1.0.tar.gz/>, consulted on February 2011.
13. soccerwindow2-5.1.0, 2011, online, available at: <http://en.sourceforge.jp/projects/rctools/downloads/51942/soccerwindow2-5.1.0.tar.gz/>, consulted on February 2011.