2012 4th International Conference on Computer Engineering and Technology (ICCET 2012)
IPCSIT vol.40 (2012) © (2012) IACSIT Press, Singapore

Imitative Attacking AI for Soccer Games

Wiruj Suwanpramote and Vishnu Kotrajaras

Department of Computer Engineering
Faculty of Engineering, Chulalongkorn University
Payathai Road, Pathumwan, Bangkok 10330, Thailand
E-mail: xthewiz@gmail.com, vishnu@cp.eng.chula.ac.th, ajarntoe@gmail.com

Abstract. Soccer is a very popular sport. It is attractive because of players' creativity, challenging situations and variety in gameplay. Soccer games successfully reproduce the feeling of the sport when human players play against each other. On the contrary, AI in soccer games usually does not possess all three qualities. An easy level AI lacks challenge and creativity because it does not perform known useful actions while a hard AI lacks variety and creativity because it executes best actions all the time. In this paper, we propose a method for providing creativity, challenge, and variety in gameplay by having AI imitate human players. Game situations are stored in our novel case database. Our case design takes into account the environment around a player as well as the player's positioning in grid format. Recording how players deal with each situation will give developers a huge playing database. Different players have different ways for tackling a problem and we can therefore obtain various plays with different degree of creativity and challenges. The paper shows that our case design can be used in actual situations. It can produce AI with the same style as its trainers.

Keywords: Soccer Games, Imitative Learning, Case Based Reasoning

1. Introduction

Recently, Technology in game industry has been developing rapidly, for instance, 3D Model or Computer Graphics, Texture Rendering, Sounds, or even Artificial Intelligence. AI technique was not used very much in games. Most developers used scripts to dictate their character behaviour. But AI scripts had problems. AI that was not very smart bored players because players found ways to exploit its weaknesses. On the contrary, making the scripted AI too clever would make players think they were cheated, and eventually quit playing the game. Soccer game AI was no exception. The states of a soccer field are very complex and are always changing, making it extremely difficult for developers to write scripts that react well in all situations. To address this problem, we developed an offensive AI for soccer games that learned to move and act by imitating players. Utilizing our model for Case Based Reasoning (CBR), our AI recorded states from various players and recorded actions from a player who had the ball. AI then behaved according to its recorded action. With this approach, game developers only need to design the game play and the characters that will be controlled by the players. With human players giving examples, our AI was able to pick up actions that demonstrated creativity and challenge in actual soccer games.

There were a number of researches from the Robocup Simulation League [1, 2] utilizing CBR. Floyd's CBR [3] recorded objects seen by players, but considering only seen objects might cause confusion. One case might be mistaken for others even though the ball was at a different location on the field. Ros [4] proposed features that should be stored in a case. However, the focus was on Four-Legged league, resulting in unnecessary attributes. Positions were stored as values, resulting in high precision stored cases. But this could also result in too many cases, thus making it hard for case prototyping.

Our work presented case prototyping similar to Floyd [5]. But we used the division of cells as case pre-processing step. To instruct the AI to mimic the trainer's playing style, we used "action frequency" in our post-processing step. This value represented how often the trainer performed each action in a state.

Robocup Simulation League does not provide interfaces for direct and total control of agents. Therefore, to apply our approach, we needed to find another test environment that permits direct and total control of agents. We addressed this issue by implementing a new game environment for the testbed. Unity3D was chosen for our development.

2. Soccer Game Environment

We used Unity3D as development tool for our testbed. Fig.1 shows Unity3D application interface.

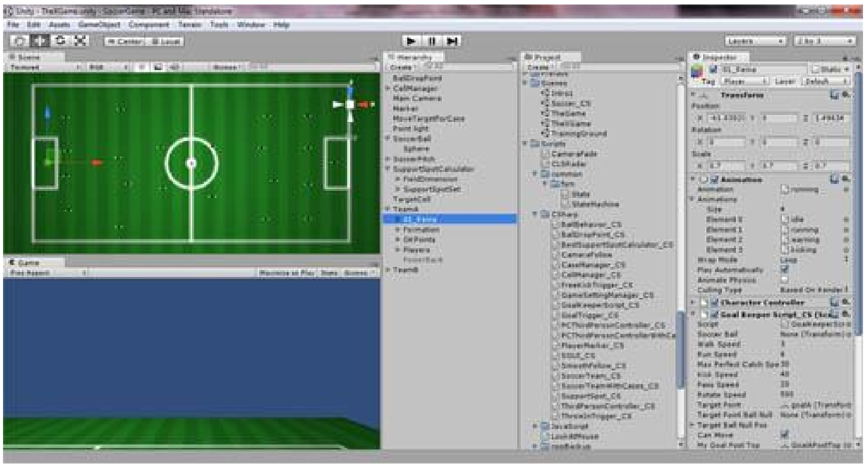

Fig. 1: Unity3D interfaces components.

Our testbed was implemented as a soccer game that consisted of 2 teams, each had 10 field players and 1 goal keeper. All field players possessed same abilities and attributes. We defined a "controlling player" to be a trainer-controlled character that currently possessed the ball. The controlling player was able to carry out various actions, i.e. move, short pass, long pass and shoot to its opponent's goal. All teammates of the controlling player acted as support, finding space on the field in order to receive passes made by the controlling player. Each game lasted 10 minutes. Our human-control team played from left to right. The opposite team was implemented using our default AI. We adapted Buckland's AI [6] for its attacking mode and used zoning (each player standing at his home position and only chasing the ball if the controlling player brought the ball into his zone) for defensive play.

3. Case Representation

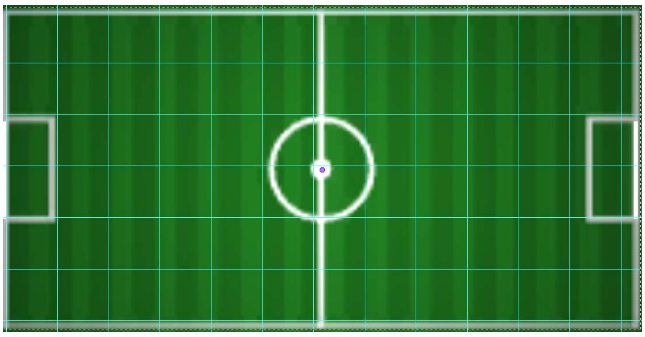

Fig. 2: Soccer field is split into cells.

A case, in our approach, was a snapshot that represented a particular situation on the soccer field. It basically described objects that a controlling player encountered. A situation usually consisted of other field players and the status of the soccer ball. Our trigger for recording each situation was an action performed by the controlling player.

Below is our case representation:

$$Ci = \{ \{tm\}, \{op\}, soccerBall, Act, ActAttr \} \quad (1)$$

Where

Ci = ith Case

tm = a teammate identity and cell identity of a cell the teammate is in

op = an opponent identity and cell identity of a cell the opponent is in

soccerBall = Soccer ball position

Act = Action performed by the controlling player

ActAttr = Attribute for the action

From the case features above, tm is a map of teammate identity and cell identity that character is standing on. Op maps opponent player's identity with cell identity each opponent is located. Our testbed recorded the action performed by a controlling player for each situation. Action attributes were meta data for each action, i.e. move action used old cell identity and new cell identity as attributes in order to provide direction to the AI, pass action used target cell identity and pass power in order to determine how the AI should kick the ball to a desired target.

4. Case Capturing

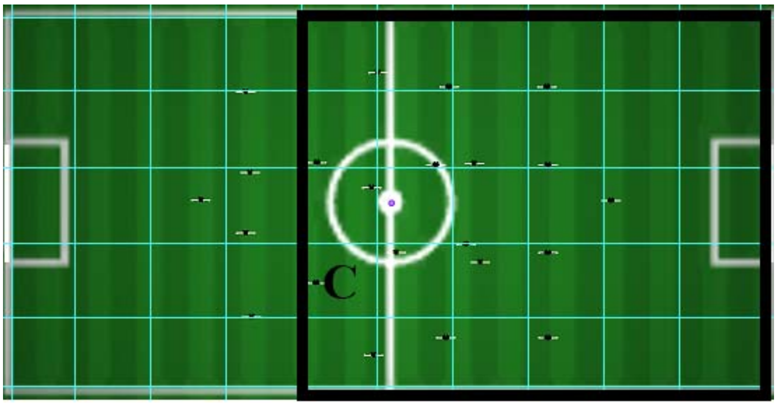

Fig. 3: If the controlling player performs an action at cell "C", each player identity in bold rectangle will be recorded, together with the cell the player is in.

Our soccer field was split into cells, each one identified by an identity called "Cell ID". When our human trainer controlled the controlling player to do something, it triggered our case storage process. The position of the controlling player was used to identify the cell it was currently in. Then case information, as shown in the previous section, was stored. When attacking, soccer players mainly concerned about players and space between themselves and their opponent's goal. Our case storage used this observation to identify players and their information that should be stored in the case. Fig. 3 shows that when a case is triggered at cell "C", the information of each of the players (except the controlling player) in the bold rectangle is stored as a pair of identification number and cell number.

5. Post Processing

After each soccer match ended, all cases in all cells were processed. We needed to perform post processing in order to combine repeated cases, thus preventing each case from getting stored repeatedly, wasting storage space.

Cases with the same players' positioning but different actions were regarded as different cases and would not be combined. We had a counter to accumulate how often an action associated with the same situation was performed. This counter partly represented the trainer's playing style. The number on the counter also indicated how successful each action was for each case, since players tended to repeat successful actions. All counters were given values during the post processing of cases.

6. Result

Our test was carried out by having our trainer play the games against our default AI for 20 matches. Our AI, having learnt from the trainer, was set against the default AI for 10 matches. Each match lasted 5 minutes. Fig. 4 shows the percentage of recorded usage for the most frequently used action for each cell by human players (blue bars) compared to the percentage of the same actions executed by our trained AI (red bars). The numbers on the x-axis represent cell IDs.

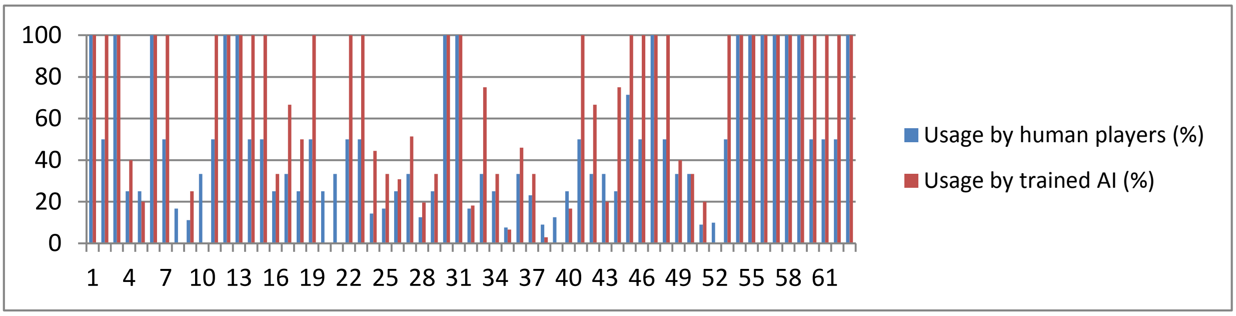

Fig 4: Ratio of usage comparison.

It can be seen that the ratio of each most frequently used action (in a cell) that AI performed was similar to that of the trainer. Within a cell, if the human trainer did not do some action very often, our AI behaved similarly. But if the trainer performed some action quite often, our AI tended to perform a lot of that action as well.

7. Conclusion and Future Works

Our paper shows that using imitation learning, with our case storage style and our case retrieval, an AI that played in a similar manner to its trainer could be constructed. With the proposed methodology, AI with various styles of play, multiple types of challenge and creativity could be constructed from gathering players' data. However, our work still had limitation due to the size of cells. We therefore aim to continue our work by utilizing cells with varying sizes weighed by locations on soccer field. We also plan to apply a new imitation framework for defensive play.

8. Acknowledgements

This research was supported by Game Lab, Chulalongkorn University.

9. References

[1] Robocup Soccer Simulation League Official Website. *http://www.robocup.org/robocup-soccer/simulation/*

[2] Robocup Soccer Simulation Community . *http://sourceforge.net/apps/mediawiki/sserver/index.php?title=Main_Page*

[3] Michael W. Floyd and Babak Esfandiari and Kevin Lam : *A Case-based Reasoning Approach to Imitating RoboCup Players.* In: *Proceedings of the Twenty-first International Florida Artificial Intelligence Research Society Conference, FLAIRS 2008*, Coconut Grove, Florida, USA, May 15-17, 2008, 251-256.

[4] Raquel Ros, Manuela Veloso, Ramon López De Màntaras, Carles Sierra, Josep Lluís Arcos : *Retrieving and Reusing Game Plays for Robot Soccer.* In: T.R. Roth-Berghofer et al. (Eds.): *ECCBR 2006*, LNAI vol.4106, Springer-Verlag Berlin Heidelberg 2006, pp.47–61.

[5] Michael W. Floyd, Alan Davoust, and Babak Esfandiari. *Considerations for Real-Time Spatially-Aware Case-Based Reasoning: A Case Study in Robotic Soccer Imitation.* In: *ECCBR '08 Proceedings of the 9th European conference on Advances in Case-Based Reasoning,* Springer-Verlag Berlin Heidelberg 2008.

[6] Mat Buckland. *Programming Game AI by Example, Chapter 4: Sport Simulation, 133-191*. Wordware Publishing, Inc.