Assignment 6 - Project Proposal

Inge Becht

April 19, 2013

1 Introduction

In Video-Games it is a challenge to create Artificial opponents that are able to make smart decisions based on a game situation, without the human players having the feeling that the AI is cheating. This is especially a problem in case of first person shooter game situations like ... ... Creating a fair Artificial Intelligence can be achieved by giving it only partial knowledge about the gameworld, for example by keeping parts of the game map unknown or by not knowing the position of the opponents at all times. In my thesis I want to focus on the latter. I want to find out if an AI can succesuflly reason about the whereabouts of the opponent when only knowing its location if the opponent can directly be observed. To make such reasoning possible, a motion model must be made that is able to predict the opponent's position given some prior information. With this model a reasoner can than be made that can make decisions about possibly ambushing the opponent or defending. For motion modelling I want to draw upon a technique that has already been researched by [] called Maximum Entropy Inverse Reinforcement Learning, but making it applicable for the game of Capture the Flag. The reasoning part will probably consist of a finite state machine that depending on certain situations will act in a certain way.

Can the Maximum Entropy Inverse Reinforcement Learning Algorithm be adapted for opponent motion modelling in a game of Capture the Flag? Can Maximum Entropy Inverse Reinforcement Learning be applied for opponent modeling to create a better AI in ca game of Capture the Flag? Can I improve

1.1 Introduction to Capture the Flag

Capture The Flag (CTF) is a well known combat objective found in video games like the Call of Duty series and Counter strike, in which two enemy teams try to catch the other one's flag from its spawn area and bringing

it back to base. Both teams can shoot at each other and in case the flag bearer gets killed, the flag will stay at its posiiton until its team retrieves it or another enemy opponent catches it and becomes the new flag bearer. The game is won after one of the teams has catched the flag for a pre-determined number of times.

2 Related Works

There have been different studies on the topic of opponent motion modelling. In the works of [2] two different models for opponent position prediction are tested in the team oriented game Counter strike. The main goal is to find out the absolute error by these implementations, as well as to test how human like the predictions made are. The tested motion models were Hidden Semi Markov Models and Particle filters.. The conclusion of the research is that Hidden Semi Markov Models work best, and are similar in accuracy towards how humans would predict opponent behavior. In case mistakes are made by the model, the mistakes are very human-like, making the AI more believable and less perfect. The research mostly thus concentrated on researching how human-like the models are, but does not research how to succesfullyintegrate such a prediction system in a multi-agent AI system.

In [5] a same approach is made using a particle model, but this time not the focus on if the predictions made are human-like but if integrating this in an AI system can enhance performance. Essentially this is the same that I want to do, except in this case it is used on a different game type, a Real Time Strategy game. The game Star Craft was used and the EISBot was enhanced with their developed reasoning capabilities. The outcome was hat the model outperformed the other models by 10 %, but that making more game states available to the bot does not always improve the performance. Although this research does sound quite similar to my own research question, the domain is different enough to expect different result, but still their particle model implementation could help me with developing my own way of prediction opponent position.

Instead of trying to predict the opponent position, the authors of [3] focused on creating a reasoner that anticipates behavior of an opponent in Quake. They did this by enhancing the Soar Quakebot with anticipation strategies. These anticipation strategies are used only in case the bot has a high chance of successfully predicting what the opponent is about to do, and in this case reasons what the bot might do if he was in its place. Using this chunking is

applied that construct some rules that completely predict the actions of the opponent and how to respond towards this situation. Although they state their additions were fruitful they do not really show experiments that confirm this. Also there are still elements in their reasoning that could be approved upon, by using recursive reasoning and by making it more general for other games. Although the reasoner is developed for a single reasoning agent, the ideas given could help for a multi-agent based system. While the previous articles give inspiration towards how to make models for the prediction of opponent position, this article goes more in depth on how to create the reasoning part of the AI.

Article [1] deals with the same problem of tracking people, but does this in real life situations, with a robot that assists humans in day to day tasks. To keep the robot from interfering it needs to predict where humans are heading and what their intensions are. This papers explains all aspects of these kinds of predictions. For example, using sensor data to sense humans and how to keep track of a single person. Both these aspects are less important for my own research, but there are sections that explain how Hidden Markov Models can be used so that motions patterns can be learned, something that is indeed interesting for me. as well as the use of an Expectation Maximisation algorithm. In section 2 their own way of modelling expectations of human motion is explained, and some basic ideas can be retrieved for own research. In section 3 Hidden Markov Models are used to learn motion patterns. This can come in really handy in case of building a reasoner, which in essence should be able to recognise some patterns and then use these patterns to respond to a given situation. Section 4 is less important, as it is about person detection and identification, something I will presume is given information in the game AI world, or at least does not need any sensor data like in case of a robot. The conclusion of their own research is that motion patterns can indeed be learned using the Hidden Markov Models and that it is able to reason with this knowledge.

In the paper [4] both the combination of making predictive models for opponent positions is discusses as well as a way of intercepting the opponents in game. The predictive models are made using a particle filter both with IRL and Brownian motion to test which works best. Not surprisingly, IRL seems to give the most accurate result and helps to intercept the opponent the best. For interception 3 different heuristics are tested. Although it looks a lot like what I want to do, it is different in multiple aspects. In this case a different game with different goals is used. There is only 1 bot against multiple different opponents, and it is only tried to intercept.

3 Method and Approach

-Want to use Inverse reinfocement learning for this purpose The game environment will be used is called the AISandbox, a tool developed my makes of the AIGameDev blog and Guerilla Games.

It would not be efficient to work from scratch, so code was used from the AISandbox Challende to implement this in. The code is written in Java. The way the code works is by -Use of code Terminator

4 Evaluation

There are multiple ways to evaluate the model. As we have seen there have been instances of research that checks if a model makes humanlike predictions or if its prediction has the smallest possible errors The way the implementation of the research problem will be evaluated depends on how far the implementation will be finished.

References

[1] Maren Bennewitz, Wolfram Burgard, Grzegorz Cielniak, and Sebastian Thrun. Learning motion patterns of people for compliant robot motion. *Internationl Journal of Robotics Research*, 24:31–48, 2005.

[2] Stephen Hladky and Vadim Bulitko. An evaluation of models for predicting opponent positions in first-person shooter video games, 2008.

[3] John E. Laird. It knows what you're going to do: adding anticipation to a quakebot. In *Proceedings of the fifth international conference on Autonomous agents*, AGENTS '01, pages 385–392, New York, NY, USA, 2001. ACM.

[4] B. Tastan, Yuan Chang, and G. Sukthankar. Learning to intercept opponents in first person shooter games. In *Computational Intelligence and Games (CIG), 2012 IEEE Conference on*, pages 100–107, 2012.

[5] Ben G. Weber, Michael Mateas, and Arnav Jhala. A particle model for state estimation in real-time strategy games. In *Proceedings of AIIDE*, page 103–108, Stanford, Palo Alto, California, 2011. AAAI Press, AAAI Press.