Multi-Agent Artificial Intelligence in Pursuit Strategies: Breaking through the Stalemate

D. Michael Franklin, Kevin Markley
Southern Polytechnic State University, Marietta, GA
mfranklin@spsu.edu, kevinlmarkley@gmail.com

Abstract

The typical artificial intelligence in gaming is single-agent. It is tasked with attacking the playing character and focuses so tightly on this objective that it acts as if it is the only enemy in the game. Typically it does not differentiate being in a setting where it is the only enemy attacking the player and a similar setting where there are multiple agents attacking the player [8]. These multiple agents are acting as single agents and losing their potentially multiplicative effect. This leads to nonsensical and simplistic "tricks" that defeat the game's artificial intelligence (AI) as well as defeating the individual AI agents without having to overcome their strategy. We wish to show that in gaming AI coordinated enemies can significantly improve the gaming experience while maintaining the game designer's original strategic intent. This coordinated multi-agent AI will be shown to have a significant impact on length of play even in simplistic games. While there are several existing methods for multi-agent AI, we present a novel approach that shares information from each individual AI agent with the other agents on their team. This differentiates it from flocking (where the other agents are often treated as additional obstacles to be avoided) and teaming (where the agents focus on the same objective but without the coordinated formational attacks). It is our hypothesis that such information sharing at the individual AI agent level creates a coordinated AI for the overall game that increases the difficulty and challenge of gameplay and requires a better strategy from the player to overcome.

In gaming, where a major concern is aesthetics (i.e., how a game makes a user feel), this loss of strategic influence can be devastating to the overall enjoyment of the game. As an example, in the early first person shooter Doom from ID Software[10] the player explored room after cavernous room of a dungeon, each filled with a variety of monsters. The variety of the monsters within these rooms was specifically designed to create an ever-increasing challenge to the player as they entered and had to run around the room while avoiding the enemy, gathering goods, and dispatching the monsters. However, the AI was designed so that each monster independently pursued the player once they entered the room. This resulted in an unintentional "cheat" whereby a player could enter a room, wait one second for each monster to recognize them, and then retreat from the room. This resulted in each of the monsters funneling through a choke point (the door) and made elimination of the threat trivially easy. This result short-circuited the otherwise well-designed gameplay intended by the designers. It is noteworthy that this behavior was lessened in subsequent releases of the game [10].

We wish to present an alternative form AI that avoids this limited type of interaction, namely the AI agents acting independently of each other rather than working together as a team. To do so, we add the multi-agent functionality to the AI for a simple pursuit game. Initially the AI directs each agent independently to pursue the target player. These agents then suffer from collision and overlapping such that the player can evade the clustered agents without difficulty. Next we introduce our multi-agent AI that coordinates the efforts of the enemy agents so that they stay in formation and work together to corner the player. In so doing we wish to show that this greatly improves the quality of gameplay and the realism simulated by the AI. Further, this upholds the original intention of the AI as designed by the developers and avoids unrealistic "cheats" to circumvent the intended gameplay. While this research is centered in gaming, we also believe that it has further reaching applications in security, simulations, and robotics.

Introduction

Classical artificial intelligence adds a depth to gameplay that mimics playing against another human. This desirable trait offers gamers a richer, more meaningful experience. While the ultimate AI might be perfectly analogous to a human opponent, the reality often falls short. There is a delicate balance that must be achieved between an invulnerable omniscient intelligent agent and a rudimentary walking target. This balance is challenging to create and even more difficult to maintain. Game creators are tasked with using the state of the world as the game understands it (which describes every object within it in perfect all-knowing detail) to create a character to mimic human behavior (which does not have such complete knowledge). As a result, the AI is often so simplistic that it becomes nothing more than walking background set decoration in an effort to keep it vulnerable. Also, because this type of character creation is difficult, programmers often take shortcuts. There are many examples available where the game presents the player with several simple AI characters in an effort to mask their vulnerability. This initially seems to offer the presentation of greater difficulty but his charade quickly falters. It is presumed that this team of enemies will work as a cohesive unit and take advantage of their greater numbers, that they will reason and act in a strategic way, and that their advantage is unsurmountable. The usual reality, however, is that this is not true. In most cases this team of AI characters is really just multiple instances of a single character AI. Due to the fact that this AI is not exhibiting teamwork there is often a "trick" (an unintended action or set of actions that circumvent the challenge that was intended by the game designer) that undermines the true nature of this weaker AI.

Learning in multi-agent systems is challenging. There is a lot of detail to be considered in the arena of teamwork in games.

> "Learning to act in a multi-agent environment is a difficult problem since the normal definition of an optimal policy no longer applies. [Michael Bowling and Manuela Veloso (2001)]."

There is much to be learned from the research in multi-agent systems. This research informs our efforts in this arena. We wish to design an engaging AI that will take advantage of outnumbering the player to reason and act as a team. By acting as a team we wish to show that even a simple AI can create a stronger challenge than the simple, single-agent method. In so doing, we will create an AI that feels more natural and avoids these tricks that allow it to be circumvented. By applying lessons learned from multi-agent system research we can have the agents themselves reason in an informed manner with the knowledge that they are not alone. While multi-agent research does not currently go so far as to begin to form strategic reasoning, we wish to show that it can. Strategic reasoning arises when the team of intelligent agents uses the available information in a way that exhibits a collective intelligence. Additionally, the agents' actions are given from the decision of the collective intelligence rather than from the agent individually. This is the contribution of this work, to reason collectively and demonstrate, even in such a simplistic environment, that this reasoning is advantageous.

We can either assume that these agents can communicate their intentions to each other or presume that there is only certain knowledge that is available to each. In either scenario we can show that the resulting performance is better than the multiple single-agent schemes.

While this research is centered in a game arena, these findings have important ramifications in other areas as well. First, we do hope to improve the AI in games by making these teams of reasonable intelligent agents perform in such a manner that the aggregate challenge they present matches the goals of the game designers. Second, moving beyond the casual game arena and into the serious games and simulations arena, we can see that such AI would better model critical systems and interactive environments. Finally, we can see that multi-agent systems can benefit from this application of collective intelligence.

In the next section we discuss the Related Works. This will show the research into which this work can be placed. The Methodology will outline the game itself and the tests and trials that were run. The Analysis will quantify the Results and related findings and lead to the Conclusions and present the Future Work.

Related Works

While there has been quite a lot of research into various artificial intelligence methods, the primary focus of much of this research is concentrated on single-agent systems, or in larger systems with multiple independent agents. Our focus is on multi-agent systems where the team is coordinating its actions guided by some larger strategy.

Simon Parson and Michael Woolridge (2002) provides a general background in both Game Theory and Decision Theory, specifically as it applies to multi-agent systems. Michael Bowling and Manuela Veloso (2002) introduces Game Theory into multi-agent learning. These works give several approaches to solving multi-agent learning systems and their mathematical foundations. These reference works provide the underpinning of the work that will be introduced herein in multi-agent systems and large scale game solutions.

There is much foundational work in both game theory and learning in multi-agent systems. Rather than review each of the multitudinous examples (like Littman (1994), Hu et al. (1998), and Greenwald et al. (2003)) in this proposal, there is a larger work that summarizes each of these and compares them. Bowling et al. (2004) also firmly establishes this background while entrenching itself in the multi-agent learning scenario, and in particular in how the related work from game theory (e.g., the Nash equilibrium) fits into the more limiting field of multi-agent learning.

While these works vary from seminal to specific, they stand to show the difference in approach between multiple agent scenarios (i.e., the agents are in the same space but act independently) and multi-agent systems (i.e., the agents are coordinating their actions and sharing information).

Methodology

Our goal is to show that multiple agents operating as a team (i.e., sharing AI information and goals) will outperform multiple agents implementing single-agent

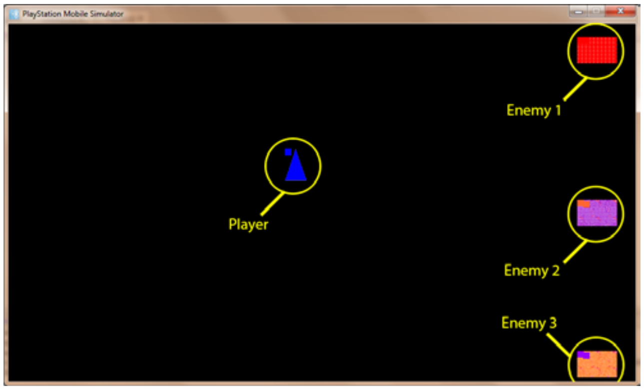

Figure 1

artificial intelligence. To do this we devised a simple game that pits one player against three opposing agents. The paradigm that we are using is a pursuit game with no obstacles. The player is placed on the left side of the screen in the middle and the three opponents are placed on the right side spread out evenly (Figure 1). Once the game begins, the opponents pursue the player until caught. The game is configured to vary the speed of both the player and the opponents and to vary the type of AI that the opponents are using.

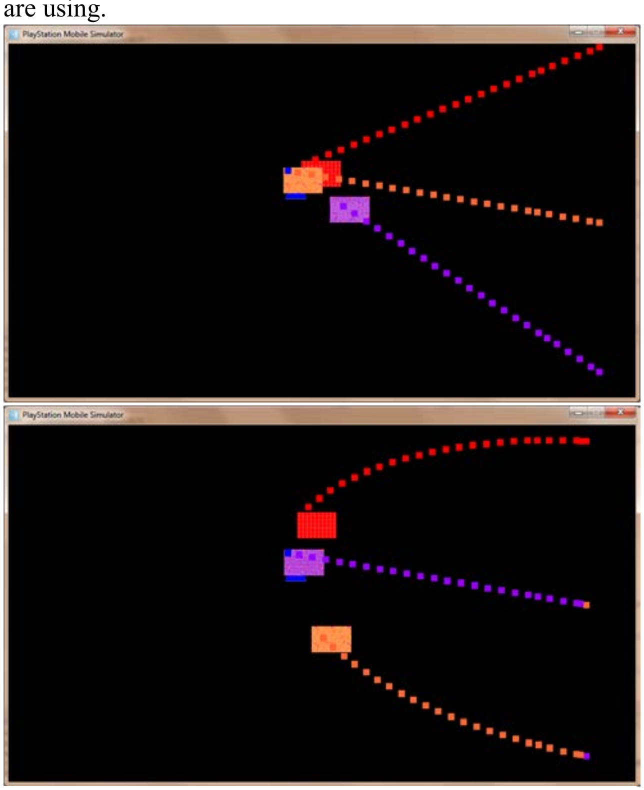

Figure 2

For this research we used two different methods of AI. The first method used was direct-pursuit. In this method the opponent directly pursues the player. Even though there are multiple agents in the game, they are all operating independently. The second method used is called aim-and-miss. This method seeks to plot the direct line to the player and then coordinate that information with the same information from the other opponents. This direct line pursuit angle is then adjusted based on the current formation. A formation is the arrangement of the opposing agents with respect to the player. While several are

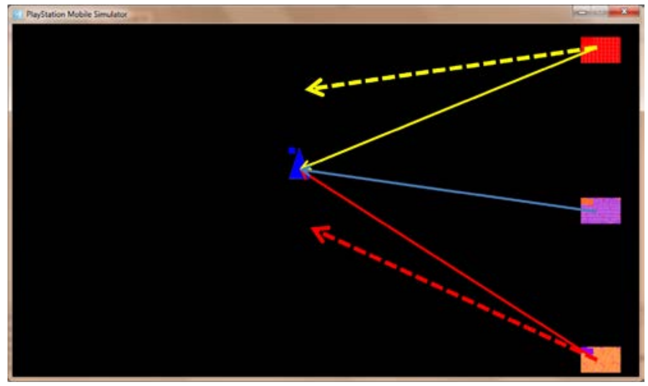

Figure 3

possible, and many were considered, this current research just explored one formation to isolate the artificial intelligence component.

The formation chosen was the C-formation where the center opponent performs direct-pursuit on the player and the other two opponents form a reverse C to close on the player like a claw (Figure 2). As the opponents pursue the player the top and bottom opponents exchange their aiming information. This causes each to aim above and below the player, respectively. By aiming high and low, respectively, they keep the player hemmed in as they close in. The team of opponents thus demonstrates a primitive strategy by surrounding the player and blocking escape angles.

To calculate the proper pursuit lanes for the aim-and-miss AI the direct-pursuit angles were calculated first. This calculation was done using the Euclidean distance as calculated via vector notation. The resultant vector is the direct-pursuit angle. This vector is then weighted against its opposite opponent's vector to derive a pursuit vector that takes the opponent just high or low of the player. To be specific, suppose the top opponent in the C-formation calculates its direct-pursuit angle to be -30° relative bearing to the player. The bottom opponent then calculates its direct-pursuit angle to be +40° relative bearing to the player. If these opponents turned to these headings they would be implementing the direct-pursuit algorithm. In the aim-and-miss algorithm each opponent adjusts its angle of pursuit by examining first its position in the formation, then its own pursuit angle, and finally its adjusted heading angle. In the example above the top opponent adjusts its heading by considering an equal-weighting formulation of its own direct-pursuit angle with the direct-pursuit angle for the bottom opponent. In this example the top opponent takes its own heading of -30° and adds the heading of the bottom opponent (40°) to arrive at a heading of 10°. This new heading angle insures that the top opponent will aim above the player and keep it from running into the player (and thus be easily avoided). The bottom opponent is also adjusting its heading to -10° to aim below the player. These two will create the pincer effect to sandwich the player between the two outer opponents. As the top and bottom opponent approach from the edges they will stay outside the directly-approaching center opponent to surround and capture the player (Figure 3).

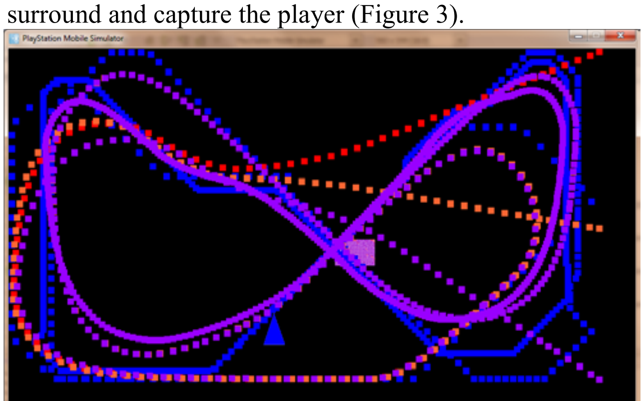

Figure 4

The tests were run by having individuals (n = 12) play the game with both AI's. They were given a short introduction to the game and oriented to the controls. After this, they played three games under each method. The type of AI they faced for the first three trials was determined randomly to remove experience bias. The participants controlled the player by moving with the directional arrows and worked to avoid the opposition. During each game there were traces placed on the screen to show the paths of the player and each of the opponents to show the pursuit paths. Each participant was then asked about their opinion of the two different AI schemas. The game was written for the PlayStation Vita and used the PlayStation Mobile (PSM) Development suite. The PSM uses C# programming.

Results

The study showed that the two AI schemas performed quite differently. The direct-pursuit AI averages 21.34 seconds per trial (std dev of 9.52), with some games ending only after the participant grew tired of running around the screen. In these latter instances it was clear they could have eluded the opponents as long as they wanted. The aim-and-miss AI showed a clear advantage with an average of 5.63 seconds per trial (std dev of 1.68). The results of the various trials and their respective runtimes are found in Figure 6. While the runtimes may have varied, the difference did not – the aim-and-miss AI was the clear winner.

The reason for the performance difference is quite clear as well. By looking at the initial pursuit paths of both schemas (Figure 2) the difference is in the approach. The direct-pursuit AI moves straight towards the player and the three opponents cluster together. Because their AI is the

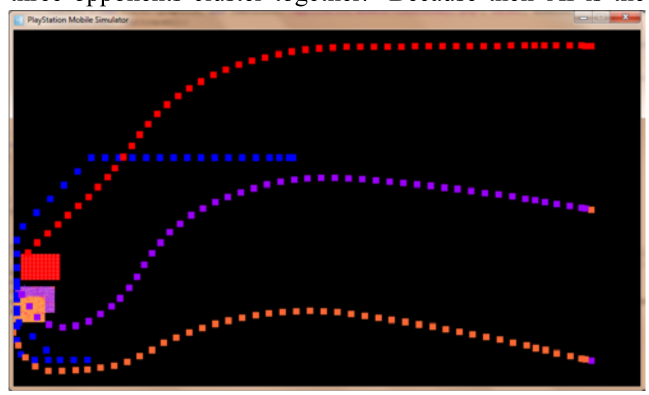

Figure 5

same they then perform as essentially one opponent, giving up their numerical advantage. This quickly evens the odds for the player as it is easier to avoid one opponent than it is to avoid three. Figure 4 shows the pursuit paths after a long game play with this simpler AI. The aim-and-miss AI initial pursuit angles show that the top and bottom opponents approach the player from their respective sides and only curve towards the player as they get close. This shows the claw formation quite clearly and exemplifies why this more strategic multi-agent approach performs better. Figure 5 shows this clearly as the game ends quickly because the player is trapped by the C-Formation

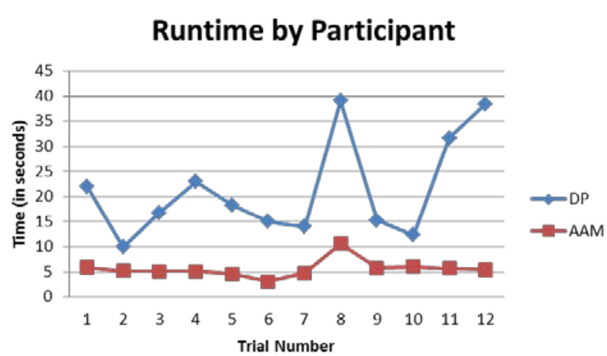

Figure 6

and cannot escape.

Conclusions

We set out to show that a single-agent AI, even when running on multiple agents simultaneously, cannot outperform the multi-agent AI that coordinates its behaviors. The trials showed clearly that the same simple game was considerably more difficult for the participants when the AI was working together to trap them. While the sample size may have been small, we have no reason to believe that the conclusions would have changes with a larger sample set of individuals. The consistency evident in our initial trial was also present in our expanded trial (which expanded our number of participants 3-fold). Listening to the participants talk about their experiences was quite enlightening:

"The first game was easy, but after several rounds it became clear that I could trap the opponents by letting them almost catch me, then evade until I got tired. The second game, where they were working together to outsmart me [sic], was much more difficult. It felt like I was being stalked, like they were hunting me. I never even came close to getting away." – Participant 4

The data was also convincing – it makes a significant difference to have the opposing agents coordinate their efforts and maintain formations. This technique was confirmed in the traces and the gameplay – it traps the player in a corner quickly and covers their exits. The trials confirmed the hypothesis. It was our goal to develop a simple test of this AI and our game shows the promise realized. It is essential to realize that our methodology was to develop a simpler way to increase the quality of the AI in a game without an undue burden on the hardware running the simulation. As a result, we have isolated the communication among the agents as our contribution to this novel approach to coordinating multi-agent behavior.

In the future we wish to build on this research by examining additional formations to see their effect on the pursuit game. Additionally, we would like to examine other formulations of the weighted algorithms that perform the aim-and-miss scheme. This game could also be modified to include obstacles, larger spaces, or 3D. There is more to be realized from the promise that this research has already shown and we want to build on this work to do so.

References

[1] Engelmore, R., and Morgan, A. eds. 1986. *Blackboard Systems.* Reading, Mass.: Addison-Wesley.

[2] Bowling, M. and Veloso, M. (2001). *Rational and convergent learning in stochastic games*. In International Joint Conference on Artificial Intelligence, volume 17, pages 1021-1026. LAWRENCE ERLBAUM ASSOCIATES LTD.

[3] Bowling, M., Veloso, M., et al. (2004). *Existence of multiagent equilibria with limited agents*. J. Artif. Intell. Res. (JAIR), 22:353-384.

[4] Chernova, S. and Veloso, M. (2009). *Interactive policy learning through confidence-based autonomy*. Journal of Artificial Intelligence Research, 34(1):1.

[5] Han, K., Veloso, M., et al. (2000). *Automated robot behavior recognition*. In ROBOTICS RESEARCH-INTERNATIONAL SYMPOSIUM, volume 9, pages 249-256.

[6] Jorg Bewersdor and Translated by David Kramer (2005). *Luck, Logic, and White Lies: The Mathematics of Games*. A. K. Peters, Ltd., 888 Worcester Street, Suite 230 Wellesley, MA 02482.

[7] Littman, M. (1994). *Markov games as a framework for multi-agent reinforcement learning*. In Proceedings of the Eleventh International Conference on Machine Learning, volume 157, page 163.

[8] Mataric, M. (1996). *Learning in multi-robot systems*. Adaption and Learning in Multi-Agent Systems, pages 152-163.

[9] Mataric, M. (1997). *Reinforcement learning in the multi-robot domain*. Autonomous Robots, 4(1):73-83.

[10] Doom, id Software (1993). *http://www.idsoftware.com/games/doom*