Effects of Communication on the Evolution of Squad Behaviours

Darren Doherty and **Colm O'Riordan**
Computational Intelligence Research Group
National University of Ireland Galway
Galway, Ireland
Email: darren.doherty@nuigalway.ie

Abstract

As the non-playable characters (NPCs) of squad-based shooter computer games share a common goal, they should work together in teams and display cooperative behaviours that are tactically sound. Our research examines genetic programming (GP) as a technique to automatically develop effective team behaviours for shooter games. GP has been used to evolve teams capable of defeating a single powerful enemy agent in a number of environments without the use of any explicit team communication. The aim of this paper is to explore the effects of communication on the evolution of effective squad behaviours. Thus, NPCs are given the ability to communicate their perceived information during evolution. The results show that communication between team members enables an improvement in average team effectiveness.

Introduction

In recent years, there has been an emergence of squad-based shooter games. The artificial intelligence (AI) of the non-playable characters (NPCs) of these games should be team-orientated and tactical as the NPCs should work together to devise the most effective method to achieve their common goal. As tactics are highly dependent on the situation (i.e. team supplies, enemy movement, etc) (Thurau, Bauckhage, and Sagerer 2004) it is very difficult for game developers not only to code the tactical behaviours but also to decide when and where it would be effective to use certain tactics. As such, game developers find it difficult to create teams of NPCs that are able to correctly assess a situation, choose effective courses of action for each NPC and work together to achieve their common goal.

Rather than attempting to develop complex behavioural systems that may allow NPCs to display intelligent team behaviour, game developers have opted to continue using deterministic techniques to implement the AI of NPCs and use simple techniques to make it appear as if the NPCs are cooperating in an intelligent manner. For example, some developers prevent two NPCs from simultaneously shooting at the player, causing them to appear to be taking turns attacking the player. This is combined with audio cues from the NPCs such as shouting "cover me" when an NPC goes to reload its weapon to create the illusion of cooperative behaviour. However, using rudimentary or "cheating" mechanisms to simulate cooperative behaviour in shooter games is less than ideal. Moreover, the use of deterministic techniques results in repetitive and predictable behaviour.

We propose that genetic programming (GP) can be used to evolve effective team behaviours for NPCs in squad-based shooter games. In previous work, GP has been successfully used to evolve effective teams in shooter environments of varying difficulty (Doherty and O'Riordan 2006; 2007). In these experiments, teams are evolved against a single powerful enemy agent that can be likened to the human player of a single-player shooter game. The difficulty of the environment is varied by altering the field of view (FOV) and viewing distance of NPCs. In modern shooter games, both the NPCs and the human player(s) have limited visual ranges within which information can be perceived.

In our previous research, the evolved teams could not communicate with each other. It was found that the effectiveness of evolved teams decreases significantly as the environments become more difficult. In this paper, NPCs are given the ability to share perceived information as the game is played in order to explore the effects of communication on the effectiveness of evolved squad behaviours. We hypothesise that explicit communication between team members should allow the NPCs to perceive the environment as a team rather than individually, which should result in more effective emergent team behaviours.

Related Work

With the emergence of squad-based shooter games, developers have struggled to create systems that allow teams of NPCs to display effective squad behaviours. As such, developers have opted to use simple techniques to create the illusion of cooperation amongst the NPCs. Command hierarchies (Reynolds 2002) and cognitive architectures (Best and Lebiere 2003) have both been proposed as methods to implementing squad AI for shooter games. Decentralised approaches (Van Der Sterren 2002a) where the team behaviour emerges from interactions of team members and centralised approaches (Van Der Sterren 2002b) where a team leader makes the decisions have also been suggested. However, none of these approaches are perfect and all require a considerable amount of time and effort to design and implement.

Evolutionary computation (EC) techniques have not been used extensively in the exploration and research of AI for computer games. As the environments and range on NPC behaviours in a computer game are generally very complex, developers are hesitant to introduce EC techniques into their games as there is no guarantee desirable behaviours will be found. However, a number of games that have incorporated EC have proven to be very successful, e.g. *Black & White* (Lionhead Studios 2001) or *S.T.A.L.K.E.R.: Shadow of Cherynobyl* (GSC GameWorld 2006). The research community has begun to realise the potential of EC techniques as developmental tools for game-AI. Champandard (2004) used a GA to successfully evolve NPCs in an first-person shooter game to dodge enemy fire. In addition, GAs has been used to successfully design tactics for an RTS game (Ponsen 2004) and to successfully tune an NPC's weapon selection parameters for a shooter game (Cole, Louis, and Miles 2004).

A few attempts have been made at evolving teams for shooter games (Stanley, Bryant, and Miikkulainen 2005; Bakkes, Spronck, and Postma 2004). Both techniques have been used to successfully evolve team behaviours. However, neither technique is ideal for developers to use to create squad behaviours for NPCs. In both cases, the team's behaviour is evolved in an adaptive manner, while the game is being played, so developers cannot tell or test, in advance, what behaviours the NPCs will exhibit or how tactically proficient they will be. Moreover, the first system (Stanley, Bryant, and Miikkulainen 2005) requires a human player to specify which attributes are to be evolved and the second mechanism (Bakkes, Spronck, and Postma 2004) requires a number of game-specific enhancements to the GA paradigm.

GP has been successfully used to simulate team evolution in a number of different simulated domains. GP was first applied to team evolution by Haynes et al. (Haynes et al. 1995b). Luke and Spector (1996) used GP to successfully evolve predator strategies that enable a group of lions to successfully hunt gazelle. In Luke's work, heterogeneous teams are shown to perform better than homogeneous teams. GP has also been used to enable a team of ants to work together to solve a food collection problem (LaLena 1997). The ants must not only cooperate in order to reach the food but must also work together to carry it as it is too heavy for one ant to carry alone. Richards et al. (2005) used a genetic program to evolve groups of unmanned air vehicles to effectively search an uncertain and/or hostile environment. Their environments were relatively complex, consisting of: hostile enemies, irregular shaped search areas and no fly zones. In addition, GP has been used to successfully evolve sporting strategies for teams of volleyball players (Raik and Durnota 1994) and teams of soccer players (Luke et al. 1997).

It has been argued that communication is a necessary prerequisite to teamwork (Best and Lebiere 2003) and plays a key role in facilitating multiagent coordination in cooperative and uncertain domains (Chakraborty and Sen 2007). Moreover, in a study conducted by Barlow et al. (2004) on teamwork in multi-player shooter games, it was found that communication is one of the three main factors that contribute to a team's success, together with role assignment and team coordination.

Gaming Environment

The environment is a 2-dimensional space, enclosed by four walls and is built using the *Raven* game engine (Buckland 2005, chap. 7). Items are placed on the map at locations equidistant from both the team and enemy starting points. These items consist of health packs and a range of weapons that respawn after a set time if collected (see Figure 1).

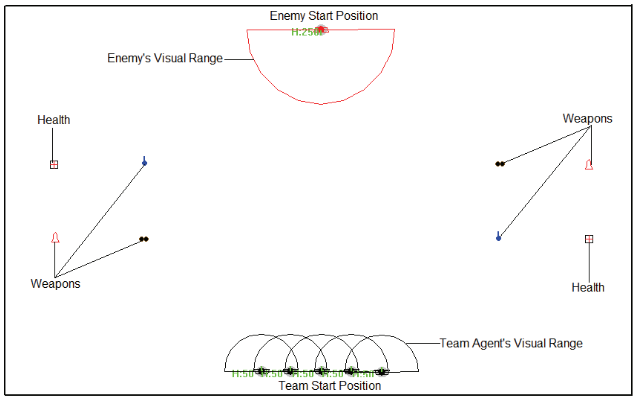

Figure 1: Environment map (FOV 180, viewing distance 50)

Both types of agent (i.e. team agents and enemy agent) use the same underlying goal-driven architecture (Orkin 2004) to define their behaviour. Composite goals are broken down into subgoals; hence a hierarchical structure of goals is created. Goals are satisfied consecutively so the current goal (and any subgoals of it) are satisfied before the next goal is evaluated. If an NPC's situation changes, a new, more desirable goal can be placed at the front of the goal-queue. Once this goal is satisfied, the NPC can continue pursuing its original goal. Although the underlying goal architecture is the same, team agents use a decision-making tree evolved using GP to decide which goal to pursue, whereas the enemy uses desirability algorithms associated with each goal. These desirability algorithms are hand-coded to give the enemy intelligent reasoning abilities. Random biases are used when creating these desirability algorithms, in order to vary the enemy's behaviour from game to game.

The team consists of five agents each of which begin the game with the weakest weapon in the environment. The enemy agent has five times the health of a team agent and begins the game with the strongest weapon with unlimited ammunition. Both types of agent have a memory allowing them to remember information they perceive. Any dynamic information, such as team or enemy positions, is forgotten after a specified time. If more than one team agent has been recently sensed by the enemy, the enemy will select its target based on distance. Weapons have different ideal ranges within which they are more effective and bullets for weapons have different properties, such as velocity, spread, etc. Agents also have limited auditory and viewing ranges within which they can perceive game information.

The Genetic Program

In our genetic program, the entire team of five NPCs is viewed as one chromosome, so team fitness, crossover and mutation operators are applied to the team as a whole. Each agent is derived from a different part of the chromosome, so evolved teams are heterogenous (see Figure 2).

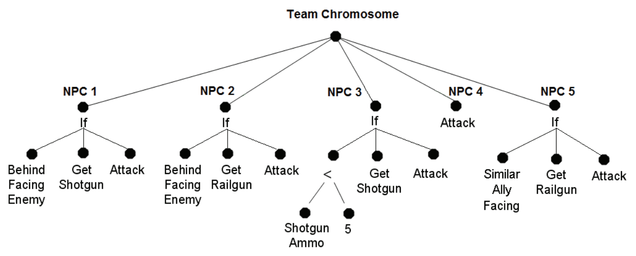

Figure 2: Sample GP chromosome

A strongly typed genetic program is used so as to constrain the type of nodes that can be children of other nodes. In strongly typed genetic programming (Montana 1995), the initialisation process and genetic operators must only allow syntactically correct trees to be produced. There are five node sets and a total of fifty nodes used in the evolution. These nodes represent: goals the NPC can pursue along with the *IF* statement, conditions under which goals are to be pursued, positions on the map, gaming parameters that are checked when making decisions and numerical values.

Fitness Calculation

The fitness function takes into account the games' duration and the remaining health of the enemy and team agents.

$$RawFitness = \frac{AvgGameTime}{Scaling \times MaxGameTime} +$$

$$\frac{EW \times (Games \times TSize \times MaxHealth - EH) + AH}{Games \times TSize \times MaxHealth}$$

where $AvgGameTime$ is the average duration of the games, $Scaling$ reduces the impact game time has on fitness (set to four), $MaxGameTime$ is the maximum game length (i.e. 5000), EH and AH are the amount of health remaining for the enemy and for all five team agents respectively, EW is a weight (set to five) that gives more importance to EH, $Games$ is the number of games played per evaluation (i.e. twenty), $TSize$ is the team size (i.e. five) and $MaxHealth$ is the maximum health of a team agent (i.e. fifty).

The team's fitness is then standardised such that values closer to zero are better and the length of the chromosome is taken into account to prevent bloat.

$$Fitness = (MaxRF - RawFitness) + \frac{Length}{LengthFactor}$$

where $MaxRF$ is the maximum value $RawFitness$ can hold, $Length$ is the length of the chromosome and $LengthFactor$ is a constant used to limit the influence $Length$ has on fitness (set to 5000).

Selection

There are two forms of selection used. The first is a form of elitism where m copies of the best n chromosomes from each generation are copied directly into the next generation. Three copies of the best and two copies of the next best individual are retained in this manner. The second method is roulette wheel selection. Any chromosomes selected in this manner are subjected to crossover and mutation (given probabilities of 0.8 and 0.1 respectively). To increase genetic diversity, there is also a 2% chance for new chromosomes to be created and added to the population each generation.

Crossover

The crossover operator is specifically designed for team evolution (Haynes et al. 1995a). A random $Tsize$ bit mask is selected that decides which of the team agents in the parent chromosomes are to be altered during crossover. A '1' in the mask indicates that the agent at that position is copied directly into the child chromosome and a '0' indicates the agent is to take part in crossover with the corresponding agent of the other parent. A random crossover point is then chosen within each agent to be crossed over. The node at the crossover point in each corresponding agent must be from the same node set in order for the crossover to be valid.

Mutation

Two forms of mutation are used. The first, randomly chooses two agent trees from the same team chromosome and swaps two randomly selected subtrees between the agents. Similar to the crossover operation, the root nodes of the subtrees must be from the same node set. The second form randomly selects a subtree from the chromosome and replaces it with a newly created tree.

Experimental Setup

In order to explore the effects of communication on the evolution of squad behaviours, the environments, genetic program and game parameters used for these experiments are identical to those used in previous work (Doherty and O'Riordan 2007), in which teams were evolved without the use of explicit communication. In Doherty and O'Riordan (2007), teams have been evolved in eight shooter environments of varying difficulty. As there was no explicit communication, teams evolved to cooperate implicitly. However, it was found that the effectiveness of evolved teams decreases significantly as the environments become more difficult.

The only difference between these experiments and those of previous research is that in these experiments NPCs are given the ability to share perceived information as the games are played. As game information is sensed by an NPC, it is broadcast by the NPC to each of its teammates in the form of messages. Each of the teammates then receive the message and store the information in memory. Types of information that can be exchanged between teammates includes the location of health and ammunition packs as well as the enemy's current position. A visualisation of an agent informing teammates of the location of shotgun ammunition is shown in Figure 3. We hypothesise that this sharing of game information between team members should allow the NPCs to perceive the environment as a team rather than individually and that this should result in more effective emergent team behaviours.

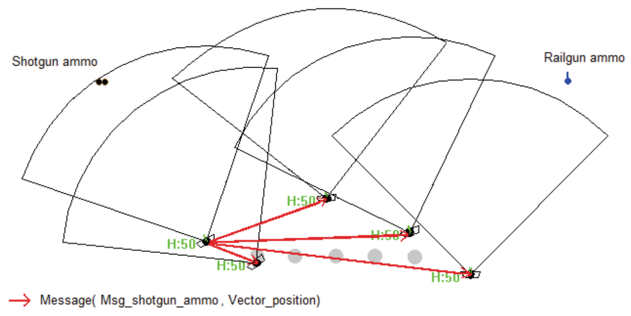

Figure 3: Sharing perceived information with teammates

In these experiments, the difficulty of the environment is varied by altering the agents' visual perception capabilities. Experiments are set up for two fields of view (90 and 180 degrees) and four viewing distances (50, 200, 350 and 500 pixels) so a total of eight experiments are conducted. The enemy viewing distance is scaled relative to the viewing distance of the team NPCs. As there are five team agents and only the one enemy agent, the collective viewing range of the team covers a much larger portion of the map than that of a single agent. Additionally, the human player in single-player shooter games, to which the enemy is likened, usually has a much longer viewing distance than that of the NPCs. For these reasons, it was decided to allow the enemy's viewing distance to be twice that of a team agent.

Twenty separate evolutionary runs are performed in each of the eight environments. In each of the runs, 100 team chromosomes are evolved over 100 generations. Each team evaluation in each generation comprises twenty games. The best performing team from each of the runs is recorded.

As the enemy's behaviour varies from game to game, due to the random biases used when initialising its desirability algorithms, each recorded team is tested more extensively using a larger number of games to obtain an accurate and robust measure of its effectiveness. The effectiveness tests involve evaluating each recorded team's performance over 1000 games and recording the number of games won by the team out of the 1000. Note that draws are not counted in the measure of team effectiveness as draws are very uncommon. Once these tests are performed for each of the recorded teams, the results from each environment are compared to previous results. Statistical significance tests are performed to determine if explicit communication provides a significant benefit to the evolution of effective team behaviours.

Results

Figure 4 and Figure 5 show the number of wins obtained by the most effective teams evolved with communication and without communication in the 90 and 180 degree FOV environments respectively. In both sets of environments, the results show that communication causes an improvement in the most effective teams evolved in all environments bar the least difficult. In the least difficult environments, the results for the best teams evolved with and without communication are almost even, differing by only 3 wins in the 90 degree FOV environment and 4 wins in the 180 degree FOV environment. We believe that communication does not benefit the teams in these environments as the individual NPCs can view the majority of the map by themselves and do not need their teammates to communicate the game information.

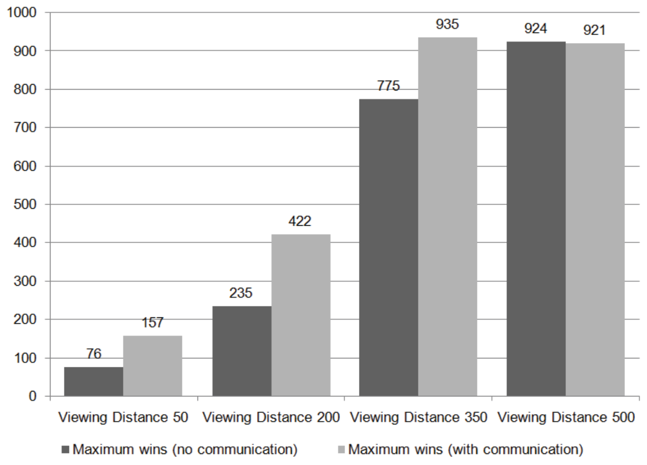

Figure 4: Comparison of maximum wins FOV 90

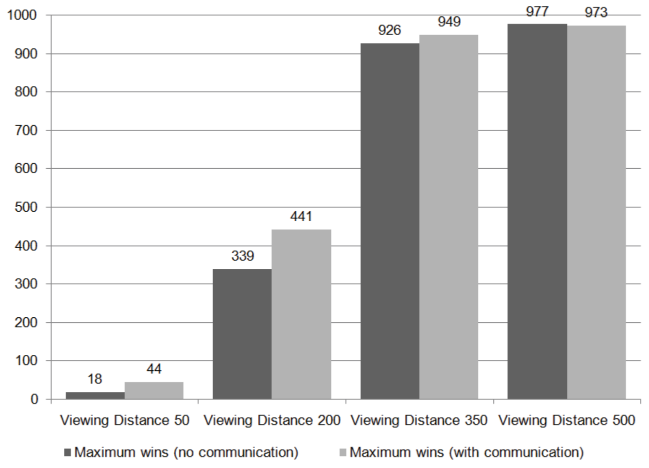

Figure 5: Comparison of maximum wins FOV 180

As the environments in Figure 4 and Figure 5 become increasingly more difficult, the percentage improvement in the effectiveness of the best teams evolved with communication over those without communication also increases. In general, communication seems to benefit the teams more as the viewing distances of team agents decreases. This is justifiable as team agents with more restricted perceptual ranges would find it more difficult to locate specific game objects on their own, and thus should benefit more from the sharing of game information.

Figure 6 and Figure 7 show the average number of wins obtained by the twenty teams evolved with communication and the twenty evolved without communication in the 90 and 180 degree FOV environments respectively. Similar to Figure 4 and Figure 5, excluding the least difficult environments, the results show an increase in the percentage improvement in average team effectiveness in those teams evolved with communication over those without communication as the environments become more difficult.

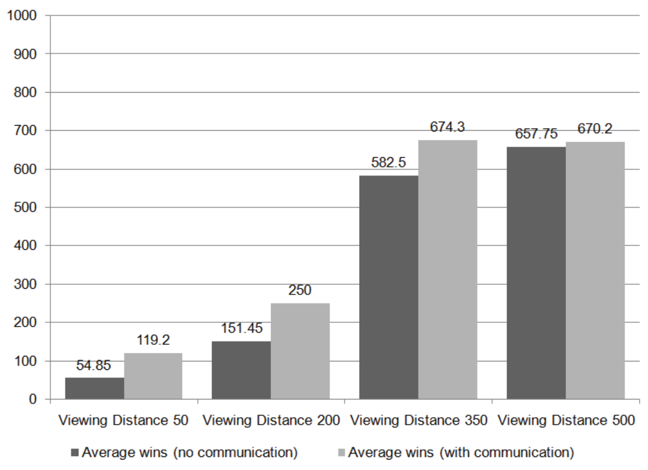

Figure 6: Comparison of average wins FOV 90

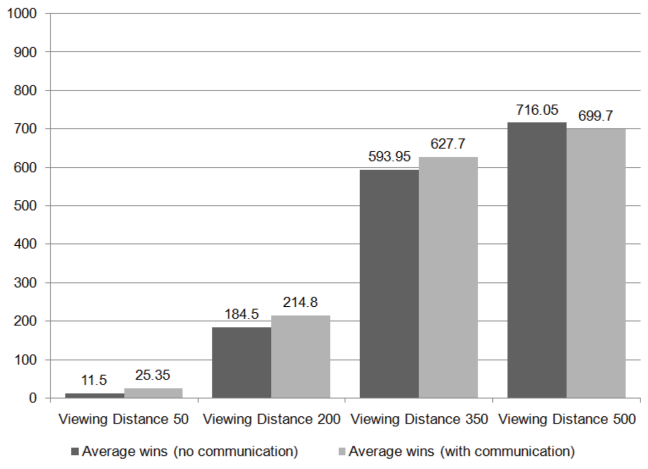

Figure 7: Comparison of average wins FOV 180

To test the significance of the results, paired T-tests have been performed between the teams evolved without communication and the teams evolved with communication in each of the environments. For a confidence interval of 95%, any comparison that records a p-value below 0.05 shows a statistically significant difference in the two samples. The results displayed in Figure 6 shows that communication affords an improvement in the average team performance in all environments where the FOV is 90 degrees. This improvement in performance is statistically significant for the 50, 200 and 350 pixel viewing distance environments (with p-values of 0.00, 0.00 and 0.03 respectively) but is not significant for the 500 pixel viewing distance environment (p-value 0.80). In Figure 7, the improvement in team effectiveness is only statistically significant in the 50 pixel viewing distance environment (p-value 0.00). In addition, the use of communication actually causes a decrease in team performance in the 500 pixel environment but this disimprovement is not statistically significant (p-value 0.82).

In the 200 pixel and 350 pixel viewing distance environments the results are statistically better when the FOV is 90 degrees (p-values of 0.00 and 0.03 respectively) but not when the FOV is 180 degrees (p-values of 0.33 and 0.63 respectively). This may be due to the fact that the enemy's FOV is also more restricted in the 90 degree FOV environments making it more difficult for the enemy to spot team agents attacking from the sides. Hence, communication between team members may provide the team with opportunity to attack more effectively. Additionally, the visual range of NPCs in shooter games is usually cone shaped, meaning their FOV is closer to 90 degrees than 180 degrees.

Conclusions and Future Work

This paper explores the effects of communication on the evolution of squad behaviours for teams of NPCs in shooter games. The results show that communication between team members enables an improvement in team effectiveness in all environments bar the least difficult one. In the least difficult environment, individual NPCs can view the vast majority of the map by themselves and communication is not needed to inform them of key game information. In addition, the decrease in team effectiveness when using communication is not statistically significant. In contrast, teams evolved in the more difficult environments, where NPC viewing ranges are most restricted, were shown to have a significant improvement in effectiveness when communication is used. The sharing of information by the team saves the NPCs having to explore the environment individually. Despite achieving a statistically significant improvement in effectiveness in the most difficult environments, the evolved teams still only managed to obtain win percentages averaging 11.9% and 2.5% in comparison to win percentages achieved in the least difficult environments which averaged 67% and 70% for FOVs of 90 and 180 degrees respectively.

The current experiments show that as the agents' individual visual fields become larger, the need for communication is reduced. This is due to the fact that the only information being communicated here is perceptual information. As an agent's own visual field becomes large, there is less of a need for teammates to inform them of the locations of game objects as they can more easily find the locations of game objects themselves. We hypothesise that information other than perceived game information that can be communicated amongst the agents, such as tactical commands, may be more important and may be unaffected by the broadening of visual fields.

In future, we wish to continue our research on the role of communication in facilitating teamwork for squad-based shooter games by exploring different visual ranges for the NPCs and evolving behaviours on different types of map. Additionally, we wish to add explicit communication nodes into the genetic program in an attempt to directly evolve effective communication between the team members. We hypothesise that teams will evolve to make use of the communication nodes in order to perform more cooperatively, particularly in the more restrictive environments.

References

Bakkes, S.; Spronck, P.; and Postma, E. O. 2004. Team: The team-oriented evolutionary adaptability mechanism. In Rauterberg, M., ed., *Proceedings of the Third International Conference on Entertainment Computing (ICEC 2004)*, volume 3166 of *Lecture Notes in Computer Science*, 273–282. Springer.

Barlow, M.; Luck, M.; Lewis, E.; Ford, M.; and Cox, R. 2004. Factors in team performance in a virtual squad environment. In *SimTecT 2004 Simulation Technology and Training Conference*.

Best, B. J., and Lebiere, C. 2003. Spatial plans, communication and teamwork in synthetic MOUT agents. In *Proceedings of the 12th Conference on Behavior Representation in Modelling and Simulation*.

Buckland, M. 2005. *Programming Game AI by Example*. Wordware Publishing, Inc.

Chakraborty, D., and Sen, S. 2007. Computing effective communication policies in multiagent systems. In *Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multi–Agent Systems*. New York, USA: ACM.

Champandard, A. J. 2004. *AI Game Development: Synthetic Creatures with Learning and Reactive Behaviours*. New Riders Publishing.

Cole, N.; Louis, S.; and Miles, C. 2004. Using a genetic algorithm to tune first–person shooter bots. In *Congress on Evolutionary Computation 2004*, volume 1, 139–145.

Doherty, D., and O'Riordan, C. 2006. Evolving tactical behaviours for teams of agents in single player action games. In Mehdi, Q.; Mtenzi, F.; Duggan, B.; and McAtamney, H., eds., *Proceedings of the 9th International Conference on Computer Games: AI, Animation, Mobile, Educational & Serious Games*, 121–126.

Doherty, D., and O'Riordan, C. 2007. Evolving team behaviours in environments of varying difficulty. In Delany, S. J., and Madden, M., eds., *Proceedings of the 18th Irish Artificial Intelligence and Cognitive Science Conference*, 61–70.

GSC GameWorld. 2006. *S.T.A.L.K.E.R.: Shadow of Chernobyl*. Published by THQ. http://www.stalker-videogame .com.

Haynes, T.; Sen, S.; Schoenefeld, D.; and Wainwright, R. 1995a. Evolving a team. In Siegel, E. V., and Koza, J. R., eds., *Working Notes for the AAAI Symposium on Genetic Programming*. Cambridge, MA: AAAI.

Haynes, T.; Wainwright, R.; Sen, S.; and Schoenefeld, D. 1995b. Strongly typed genetic programming in evolving cooperation strategies. In Eshelman, L., ed., *Genetic Algorithms: Proceedings of the Sixth International Conference (ICGA95)*, 271–278. Pittsburgh, PA, USA: Morgan Kaufmann.

LaLena, M. 1997. Teamwork in genetic programming. Master's thesis, Rochester Institute of Technology, School of Computer Science and Technology.

Lionhead Studios. 2001. *Black & White*. Published by Electronic Arts. http://www.lionhead.com/bw/.

Luke, S., and Spector, L. 1996. Evolving teamwork and coordination with genetic programming. In Koza, J. R.; Goldberg, D. E.; Fogel, D. B.; and Riolo, R. L., eds., *Genetic Programming 1996: Proceedings of the First Annual Conference*, 150–156. Stanford University, CA, USA: MIT Press.

Luke, S.; Hohn, C.; Farris, J.; Jackson, G.; and Hendler, J. 1997. Co–evolving soccer softbot team coordination with genetic programming. In *International Joint Conference on Artificial Intelligence–97 First International Workshop on RoboCup*.

Montana, D. J. 1995. Strongly typed genetic programming. *Evolutionary Computation* 3(2):199–230.

Orkin, J. 2004. *AI Game Programming Wisdom 2*. Charles River Media. chapter Applying Goal-Oriented Action Planning to Games.

Ponsen, M. 2004. Improving adaptive game–AI with evolutionary learning. Master's thesis, Delft University of Technology.

Raik, S., and Durnota, B. 1994. The evolution of sporting strategies. In Stonier, R. J., and Yu, X. H., eds., *Complex Systems: Mechanisms of Adaption*, 85–92. Amsterdam, Netherlands: IOS Press.

Reynolds, J. 2002. *AI Game Programming Wisdom*. Charles River Media. chapter Tactical Team AI using a Command Hierarchy, 260–271.

Richards, M. D.; Whitley, D.; Beveridge, J. R.; Mytkowicz, T.; Nguyen, D.; and Rome, D. 2005. Evolving cooperative strategies for UAV teams. In *Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation*, 1721–1728. New York, NY, USA: ACM Press.

Stanley, K. O.; Bryant, B. D.; and Miikkulainen, R. 2005. Evolving neural network agents in the nero video game. In *Proceedings of the IEEE 2005 Symposium on Computational Intelligence and Games (CIG05)*.

Thurau, C.; Bauckhage, C.; and Sagerer, G. 2004. Imitation learning at all levels of game–AI. In *Proceedings of the 5th International Conference on Computer Games, Artificial Intelligence, Design and Education*, 402–408.

Van Der Sterren, W. 2002a. *AI Game Programming Wisdom*. Charles River Media. chapter Squad Tactics: Team AI and Emergent Maneuvers, 233–246.

Van Der Sterren, W. 2002b. *AI Game Programming Wisdom*. Charles River Media. chapter Squad Tactics: Planned Maneuvers, 247–259.