Towards Real-time Behavioural Evolution in Video Games

Christopher J. Headleand, Gareth Henshall, Llyr Ap Cenydd and William J. Teahan

Abstract This paper presents preliminary work in realtime behavioural evolution of non-player characters in video games. We present an approach, utilising a modified version of the Template Based Evolution algorithm, to evolve NPCs during a first person shooter game. Through the research we demonstrate how this approach could be a viable method of introducing evolutionary components into industry quality games, to produce procedural, emergent behaviours.

1 Motivation

There are examples of using evolutionary algorithms in games, but these tend to be found in experimental, or independent games rather than high quality systems released by a studio or publisher. One of the possible reasons for this is that while evolutionary algorithms have been used successfully to prime leading technologies such as procedural animation systems, they can have turbulent effects in gameplay mechanics. There is no guarantee that they will evolve a solution that is engaging, enjoyable or consistent with the established design of the game. However, procedural generation in games is becoming a field which industry is beginning to pay more attention, partly because it can reduce development times on larger games, as every unique component needn't be designed by hand, but also because it enables larger, open-world games where manual design may simply be impossible. It also introduces a new gameplay mechanic, by allowing the same player to have a different experience every time they enter the game world. Due to this interest from industry, evolutionary game content warrants additional research.

Christopher J. Headleand, Llyr Ap Cenydd, Gareth Henshall, William J. Teahan
Bangor University, Wales
e-mail: {c.headleand, g.i.henshall, llyr.ap.cenydd, w.j.teahan}@bangor.ac.uk

2 Evolutionary Content in Games

There are very few examples of mainstream games which evolve content during play. Two examples in the literature are Galactic Arms Race (GAR), where new weapons systems are evolved using the cgNEAT algorithm [1], and Petalz [4] where flowers are procedurally evolved in a social online game. There are also games where content is evolved offline, including the game environment [6] and the game play rules [7] but these are typically academic projects.

One interesting game which developed out of research is Nero, in this game the player evolves robots through training in a sandbox environment before deploying them against another players team [5]. In this game the evolution of the robots is integral to the gameplay mechanic, making it quite unique. Another interesting item to identify is that similar to GAR, Nero uses a variation of the NEAT algorithm. However, there remains no examples of games where non player character behaviour has been evolved in real-time.

3 Template Based Evolution

Template Based Evolution is a method for the behavioural evolution of virtual agents, within a predesigned set of constraints. Within this method a an agents template is defined, which contains all the possible inputs and outputs of that agent. These templates contain evolved components, governing the attributes of the agent, the conditions that evaluate the inputs or the selection of an activated output given a set of preconditions. By using this approach, a designer is able to define an agent prototype, and ensure that all evolved versions of this template still comply to a basic set of conditions. This allows the designer to steer clear of problems that could be encountered if they had employed an open evolution method, such as invalid mutations and implausible behaviour.

Each TBE simulation is designed bottom-up, starting with the task environment and ending with the individual agents. The process is designed this way to force the designer to consider the echo logical nice of the agents as the principle driver of evolution. For a full description of TBE see [3, 2].

3.1 The Environment

The environment is the location where the agents fitness is tested through successive trials. A trial is a particular challenge that the agent must survive if they are to reproduce. These can be explicit obstacles such as a predator, or more implicit challenges, such as the need to migrate between two locations before a seasonal change.

TBE simulations make use of implicit fitness functions, which is the key purpose of the environmental trials. The key distinction between the two is that explicit fitness functions reward specific behavioural elements, shaping an overall behaviour from a set of predefined primitives. Implicit functions however operate on a more abstracted level, rewarding the completion of a task, but the agent is free to complete it in any fashion.

3.2 The Species Template

The species template is a generic prototype of the agent being evolved. It contains a description of all the inputs and outputs accessible to each agent within that species as a subsumption architecture. A common code block is defined for each agent, with the agents genome defining either the behaviour selected at each layer or conditions under which the sensors are activated.

3.3 The Agent

Each individual agent is defined by a genome, which is a collection of attributes. Each position in that genome has a specific role in structuring the behaviour of the agent, and these positions are the same across all agents. For example, position 0 in the genome may define how every agent in the species processes an input, and position 2 may define the agents speed. However, the values stored in each of these positions may be different in each agent, the product of their evolution.

3.4 Modification for Real-Time Game Content Generation

Typically, as with most evolutionary algorithms TBE uses a relatively large population and many generations. However, this would be inappropriate in a computer game where the player will combat against a limited number of agents, in a continuous fashion. To modify TBE for this application, instead of evolving a new generation of agents to replace the previous, a new ”brain” is evolved and deployed onto the existing agents in the scene. If no existing agents are available (for example if the player has destroyed them) when a new brain is to be deployed then a new agent is spawned at the edge of the environment to host it. This cycle of generation, as opposed to being based on a set time or task, is instead triggered by combative engagements withe the player. At the end of an engagement the current genomes in play are evaluated and used to spawn the replacement genomes. As each generation only contains five agents a relatively high mutation rate is used to maintain diversity.

4 Proof of Concept Game

To test the use of TBE in real-time gaming, a testbed, first person shooter (FPS) game was developed. In this implementation a large game environment, made of block obstacles, is randomly spawned with the player placed in the centre. Then five opponent NPCs are spawned at the edge of the map, in a position ensuring that they are initially occluded from the player (see figure 1). The objective of the

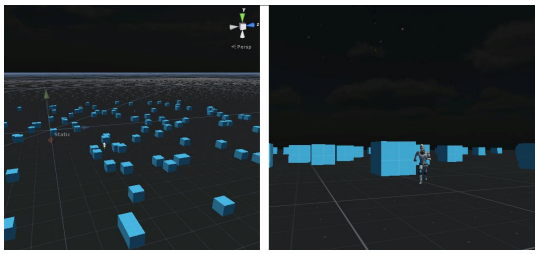

Fig. 1 A birds-eye, and player-perspective view demonstrating the randomly generated world. The player is spawned towards the centre of the map, the team of opponents are spawned towards one of the edges.

game is for the player to survive 5 minuets in the game world. The opponents have the ability to shoot the player, which is moderated for playability by an artificial stupidity function which diminishes each opponents accuracy at increased ranges. Both the player and the NPCs can shoot 5 rounds per second and survive 20 shots before being destroyed.

4.1 Template

The agent template is a subsumption architecture (see fig 2), activated by three sensors. The sensors allow the agent to know if it could see the player at different ranges, if it could hear one of its companions calling for help, and if it had been recently (0.5 of a second) been shot. Each layer of the subsumption activated a one of the seven following behaviours.

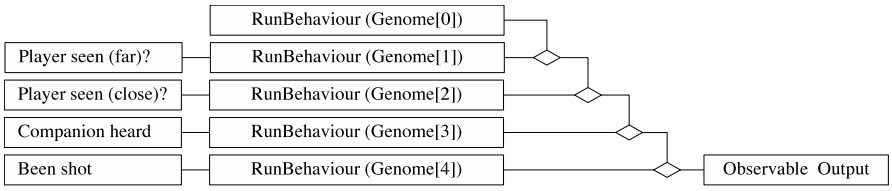

Fig. 2 A visual representation of the subsumption-based species template used within all the opponent agents.

Shoot	If the opponent agent is able to see the player, and this behaviour was activated it would shoot in the direction of the player.
Retreat	The agent would select a direction, 180 degrees from the player and move.
Run to NPC	The agent would move towards the direction of the last call for help. If there had been no call for help then the agent would remain stationary.
Call for help	The agent would call for help.
Run to player	If the agent was able to see the player it would run towards it. If the agent was unable to see the player it would move forward at its current heading.
Explore	The agent would randomly explore the environment.
Regroup	The agent would run towards the nearest companion agent.

4.2 Evolution Cycle and Fitness

As mentioned in section 3.4 the evolution cycle is defined not by a specific length of time, or the completion of a task, but is instead punctuated by the termination of an engagement with a player. An engagement is defined as a combat encounter between the player and one or more of the agents, notably that shots must have been fired, and hit at least one of the parties in the confrontation. The engagement is considered to have concluded when either the agents, or the player has retreated outside of the conflict zone or when one of the engaging parties have been destroyed. Engagements are based on the player perspective, thus, if the player is engaged with one agent, and another joins mid-combat it is considered a single engagement, not two separate engagements.

From the agents perspective, encounters are either considered **successful** (they have inflicted more damage than they have received), **survived** (they have received more damage, but inflicted some), **retreat** (they have received damage and inflicted none) or **destroyed** (the agent was killed in the engagement).

5 Post-Gameplay Conclusions

The game demonstrated that evolution can be applied to the real-time generation of behaviours. However, the most interesting insight we gained is that the NPCs trended towards learning roles within a team, rather than single agent strategies. For example, after one engagement a team of NPCs were evolved who remained motionless, aside from a single agent who explored the environment and called for

help when it saw the player, resembling scouting behaviour. Another engagement produced a team of agents who confronted the player at different ranges (snipers and assault troops) keeping the player distracted. Occasionally tactics were evolved which allowed the NPCs to overpower the player, roughly two out of three games.

One criticism was that if an NPC survived, its personality sometimes changed too drastically, breaking immersion in the game. One possible adaptation to solve this issue in future implementations, would be to conduct a similarity match between the new genomes being deployed and the current genomes in play. The algorithm could then attempt to deploying new genomes which are to current genomes in play to the same host agent, causing a more subtle change in personality.The next stage of this study will extend the template to allow for more complex behaviour and add additional behaviours. In addition, a full user study will be undertaken to assess the gameplay quality of evolved behaviours using TBE.

6 Acknowledgements

Gareth Henshall would like to thank HPC Wales for supporting his research.

References

1. Erin J Hastings, Ratan K Guha, and Kenneth O Stanley. Evolving content in the galactic arms race video game. In *Computational Intelligence and Games, 2009. CIG 2009. IEEE Symposium on*, pages 241–248. IEEE, 2009.
2. Chris Headleand, Llyr Ap Cynedd, and William J Teahan. Berry eaters: Learning colour concepts with template based evolution evaluation. In *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, volume 14, pages 473–480.
3. Christopher James Headleand and William J Teahan. Template based evolution. In *Proceeding of the fifteenth annual conference companion on Genetic and evolutionary computation conference companion*, pages 1383–1390. ACM, 2013.
4. Sebastian Risi, Joel Lehman, David B D'Ambrosio, Ryan Hall, and Kenneth O Stanley. Combining search-based procedural content generation and social gaming in the petalz video game. In *AIIDE*, 2012.
5. Kenneth O Stanley, Bobby D Bryant, and Risto Miikkulainen. Evolving neural network agents in the nero video game. *Proceedings of the IEEE*, pages 182–189, 2005.
6. Julian Togelius, Renzo De Nardi, and Simon M Lucas. Towards automatic personalised content creation for racing games. In *Computational Intelligence and Games, 2007. CIG 2007. IEEE Symposium on*, pages 252–259. IEEE, 2007.
7. Julian Togelius and Jürgen Schmidhuber. An experiment in automatic game design. In *Computational Intelligence and Games, 2008. CIG'08. IEEE Symposium On*, pages 111–118. IEEE, 2008.