CSC2034

Critical Paper Review

Cameron Main | 200425522

Word Count: 1515
March 2022

NEWCASTLE UNIVERSITY

The 2006 paper, 'Three States and a Plan: The A.I. of F.E.A.R.' by Jeff Orkin of Monolith Productions and the M.I.T. Media Lab, demonstrates how implementing a planning system into a video game can improve the process of developing character behaviours. With the game F.E.A.R.: First Encounter Assault Recon (2005), serving as a case study, Orkin illustrates how planning can be successfully implemented to enable real-time artificial intelligence (A.I.) reasoning. Such revolutionary methods empower non-player characters (NPCs) with the ability to dynamically reason; thus, proving its superiority over the standard A.I. practises of the time.

F.E.A.R. is a singleplayer, first-person shooter created by Monolith Productions originally released in 2005 on PC, with subsequent releases on consoles a year later. The tone of the game aimed to place the player in a sinister but fast-paced shooter environment. Orkin claimed, "We [Monolith Productions] wanted F.E.A.R. to be an over-the-top action movie experience, with combat as intense as multiplayer against a team of experienced humans" [1].

Despite setting the bar high, Orkin and the remaining Monolith members were able to exceed the expectations of both game critics and consumers alike. Many praise the seemingly life-like and challenging intelligence of the A.I. [2]. In this review, I will explore what methods Orkin employed to achieve such an esteemed solution aimed at improving A.I. practises.

The 'Three States and a Plan: The A.I. of F.E.A.R.' paper was formed as supplementary material to the conference talk of the same name by Orkin at Game Developers Conference (GDC) 2006 [3], [4]. Orkin begins by mentioning what he believes to be the two most common A.I. techniques applied to games, that being A* path-finding and Finite State Machines (FSMs). "Nearly every game that exhibits any A.I. at all uses some form of an FSM to control character behavior, and A* to plan paths." [1]

Beyond this, however, there is a failure to elaborate or provide any explanation as to what exactly these techniques entail. Orkin assumes the reader's A.I. comprehension to be of a high standard and familiar with common techniques applied within games. Although the initial audience at GDC were most likely game developers, it is improper for Orkin to assume their understanding. Therefore, explanations should be provided with further references to each of the corresponding topics.

For clarity, FSMs are computational models that consist of a set of states that map inputs and a current state to a next state [5]. Within a game's environment, NPCs can utilise an FSM to determine what their current behavioural action is, and what they "intend" the next action to be based on an input stimulus. Of course, NPCs are not conscious so their actions are always predetermined to the set of possible states. Orkin's aim of enabling NPCs for real-time reasoning begins here with the FSM.

When Orkin introduces the FSM that Monolith used in the game, a diagram (seen below) is used to illustrate the three states an NPC can be in to dictate its behaviour.

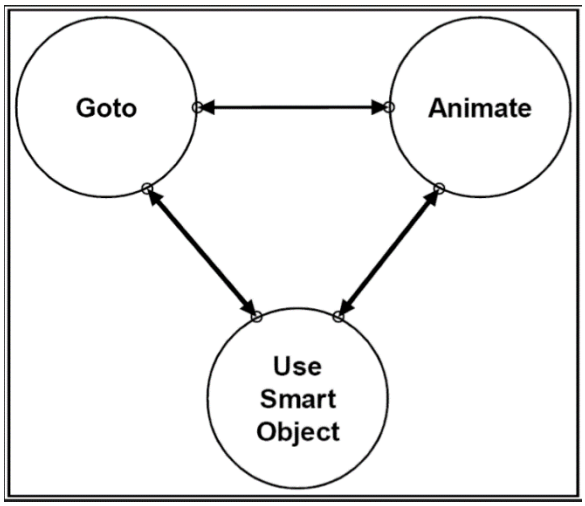

Figure 1. F.E.A.R.'s Finite State Machine [1]

Goto is a behaviour state used to guide the NPCs throughout the game world from point to point using predetermined nodes. The *Animate* state, as the name suggests, executes the animation to be played for relevant the situation of the NPC to obtain a suitable level of immersion for the player.

Unfortunately, a problem arises when Orkin addresses the *UseSmartObject* state. Although described well, a means an NPC can interact with game world objects such as doors and tables. Orkin informs the reader to simply disregard the category, and in fact "For the purposes of this paper, we can just consider *UseSmartObject* to be the same as *Animate*" [1].

As a result, not only is the figure Orkin provides invalidated but the paper title itself. In this case, perhaps 'Two States and a Plan' is better suited. The absence of consistency in this regard is poor for the author and leaves the reader confused due to the conflicting title, text and diagram. A large-scale software project, such as creating a video game, is an arduous and long process that allows for such redundancies to fester. Orkin should have attempted to explain how and why this occurred as a means to alleviate any possible confusion.

Regardless of this blemish, Orkin uses the FSM foundation to reveal the crux of the paper: Goal-Orientated Action Planning (GOAP). A previous paper by Orkin details how planning was not common within video games of the time, but instead reserved for mechanical, safety-critical systems [6]. Believing planning would produce a superior A.I. system, the Monolith team built their implementation into the game engine with GOAP.

GOAP is built upon the Stanford Research Institute Problem Solver (STRIPS); a formal planning system used to represent actions and goals [7]. Despite STRIPS being developed around 34 years prior, Orkin envisioned a link between the system and a potential to modify it for implementation within a games engine [1]. This was achieved within F.E.A.R. as information from the game environment, as well as the actions of the NPCs, are modelled similarly to that of STRIPS. It operates by orchestrating a search that establishes a list of possible actions an NPC can do, while the goals are defined at runtime. As a result, each NPC has a set of dedicated goals, ordered by priority, which could entail the likes of *Patrol*,

Attack Player or *Retreat*. Goals can be achieved through actions such as *Dodge*, *Fire Weapon* or *Reload*. If for any reason a goal fails, the succeeding goal is chosen.

One major benefit Orkin achieves with GOAP is the ability to decoupling actions from goals. Making NPC behaviours modular allows for the quick creation of new and unique character types in the game with minimal effort. Not only this but preconditions can be used as a deterministic factor into an NPCs behaviour. For example, an A.I. agent is only able to access specific behaviour states based on a specific criterion. In game, this could translate to an enemy NPC only being able to search for a player if the ceiling light is turned on in the game world. Consequently, this does indeed facilitate improved character behaviour as Orkin claims in his opening statements.

On the other hand, the most significant weakness of implementing GOAP is the memory required for it to perform. GOAP is expensive in comparison to a traditional FSM, so memory management must be pivotal in a successful implementation. The development team aimed to reduce memory usage by using hash tables to store actions depending on the effects of the action. Furthermore, all data regarding the game world and relevant to the A.I. was gathered into a blackboard - a common shared knowledge base between the A.I. agents [1].

As a result, the game was shipped with a minimum PC specification requiring at least 512 MB but recommended 1 GB of memory to run well [8]. Within the context of 2005 hardware, 1 GB of memory was demanding for the PCs at the time, with many similar titles within the genre requiring just half of these values. The typical PC in 2005 had just half a gigabyte of memory, meaning they fell just inside the minimum required specification [9].

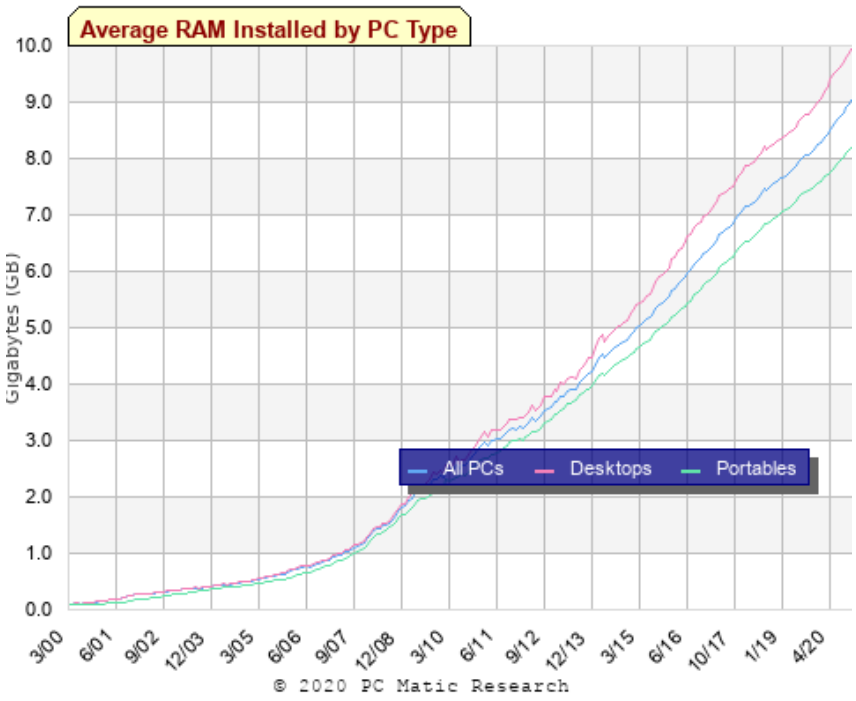

Figure 2. Average PC Memory by Year [9]

A lightbulb moment occurs when Orkin finally introduces the squad manager. When multiple NPCs are in close proximity to one another, the squad manager groups them into a squad and assigns goals to work cooperatively with one another. However, an NPC can overrule the manager's decision if it believes the threat level to be too high to complete the actions.

For example, if the squad manager sends an action to the agent to *Flank* with a goal of *Kill Player,* but world information, such as the player's position, indicates it would endanger the NPC then the goal is cancelled and exchanged for another. This occurs as the overall goal of NPCs is not to simply 'kill player' as with many other shooter games, but to instead reduce the level of 'threat' at any given time. Although the NPC may attack the player as a means to reduce the 'threat', in some circumstances it may opt to flee or hide to preserve its own life. This pseudo-fight-or-flight response distinguishes the F.E.A.R. from other games as it facilitates unique and unpredictable behaviours that engage the player.

Furthermore, each A.I. is completely unaware of the other's existence but can commit tactical flanking manoeuvres towards the player via squad manager commands. Thus, the *illusion* of communicative squad behaviour is there from the player's perspective, expanding immersion and developing character behaviour just as Orkin intended.

Orkin's work developing GOAP subsequently influenced game developers and their implementation of NPC A.I., with it making an appearance in numerous AAA titles such as Fallout 3, Just Cause 2 and Deus Ex: Human Revolution [10]. Overall, GOAP broke new ground by building upon a pre-existing system from another field, modifying it and paving the way for a new style of game development. Although GOAP comes at a performance cost, it reinvigorated the industry and F.E.A.R. was adored by players often enthralled by the advanced A.I. system Orkin implemented.

References

[1] J. Orkin, "Three States and a Plan: The A.I. of F.E.A.R.," in *Game Developers Conference*, 2006.

[2] "F.E.A.R. PC Reviews," Metacritic, 2005. [Online]. Available: https://www.metacritic.com/game/pc/fear. [Accessed March 2022].

[3] J. Orkin, "Three States and a Plan: The A.I. of F.E.A.R.," GDC Vault Audio Recording, 2006. [Online]. Available: https://gdcvault.com/play/1013282/Three-States-and-a-Plan. [Accessed March 2022].

[4] J. Orkin, "Three States and a Plan: The A.I. of F.E.A.R.," GDC Vault Slides, 2006. [Online]. Available: https://gdcvault.com/play/1013459/Three-States-and-a-Plan. [Accessed March 2022].

[5] P. E. Black, "Finite State Machine," Nist.gov, ed. 4 October 2021. [Online]. Available: https://www.nist.gov/dads/HTML/finiteStateMachine.html. [Accessed March 2022].

[6] J. Orkin, "Symbolic Representation of Game World State: Toward Real-Time Planning in Games," *AAAI Challenges in Game AI Workshop,* 2004.

[7] R. E. Fikes and N. J. Nilsson, "STRIPS: A new approach to the application of theorem proving to problem solving.," *Artificial Intelligence,* vol. II, pp. 189-208, 1971.

[8] System Reqs, "F.E.A.R. System Requirements," [Online]. Available: https://gamesystemrequirements.com/game/fear-first-encounter-assault-recon. [Accessed March 2022].

[9] PCMatic, "PC Memory Trends," TechTalk, [Online]. Available: https://techtalk.pcmatic.com/research-charts-memory/. [Accessed March 2022].

[10] J. Orkin, "Goal-Orientated Action Planning (GOAP)," [Online]. Available: https://alumni.media.mit.edu/~jorkin/goap.html. [Accessed March 2022].