# An Investigation into the Development of Companion AIs for First Person Shooters

GDEV60001 GAMES DEVELOPMENT PROJECT

Sean Tan
SUPERVISOR: NAME
SECOND SUPERVISOR: NAME

# Contents

# Abstract

Player frustration with artificial intelligence(AI) driven non-player characters(NPCs) intended to support them is a common plight of gamers, the source of which can range from the AI NPCs being largely superficial or being obstructive to the point it ruins the fun and immersion of the player. To address this and find potential solutions to it, an investigation into the development of companion AIs within the context of first person shooters was launched. This paper documents the research conducted into first person shooters, which includes topics like basic controls, gunplay and team communication. Another area of research this paper delves into is AI believability, covering topics like its importance in games and player immersion. The main portion of the research for this paper was focused on AI decision-making systems, where it covers how they work and evaluates its advantages and disadvantages. An artefact was also developed alongside it for the purpose of gathering primary data on one of the researched method of AI control, this was created using Unreal Engine 5 as a platform and the decision-making system chosen to have primary data gathered on it was behaviour trees. The paper goes over the data gathered from willing participants' experience playing with the artefact as it assists in evaluating the effectiveness of using behaviour trees as a system for controlling AI companions.

# Introduction

When First Person Shooters (FPS) were first created, Artificial Intelligence was typically implemented as a method of controlling Non-playable Characters (NPC) (Warpefelt, 2016). As FPS games evolved and gained popularity, more pre-existing mechanics along with new ones were being injected into the genre, like inventory systems and aiming down sights.

Warpefelt stated in 2016 that NPC is typically used as a generic umbrella term for any entity in the game not controlled by the player, the term has multiple categories under it and NPC as a term is broad enough to encompass opponent AIs, like the Cacodemon from DOOM (DOOM, 1993), to the shopkeeper AIs, like the Merchant from Resident Evil 4 (Resident Evil 4, 2005). Although the difference between the two examples can be easily extrapolated, since the Cacodemon attempts to halt the progress of the player, the question of what supports the supporting AI, such as shopkeeper AI and companion AI, from each other arises.

The fundamental difference between these definitions is the intent behind its implementation, and how it impacts the player's experience (Bouquet, Mäkelä and Schmidt, 2021). The companion AI that the project is concerned with directly falls under Warpefelt's "Friendly" categorisation and is stated to be unique from its other peers in the category due to them being controllable by the player (Warpefelt, 2016). Pretty in 2022 improved upon this definition, where they argued that alongside the player influence over its behaviour, they were autonomous and would be unique in comparison with other NPCs due to how it continually stuck with the player and assisted in aspects of the game like combat (Pretty, Fayek and Zambetta, 2022). For the purposes of the project, the updated definition by Pretty in 2022 will be adopted as the interpretation of what a companion AI is and will be used as the guide for research and development (Pretty, Fayek and Zambetta, 2022).

Despite the numerous memorable companions in FPS games, like Elizabeth from Bioshock: Infinite being the gold standard and Dogmeat from the Fallout series, companion AIs in FPS games often end up lacklustre and frustrate players to the point of reducing their immersion and faith in the AI's ability to assist them (Tremblay and Verbrugge, 2013). One of the main reasons for this contrast is due to the environment, they are ultimately an exception due to the narrative purpose they provide and their willingness to stick with the player throughout the story (Emmerich, Ring and Masuch, 2018). The project is mainly concerned with the decision-making aspect of AIs and how to improve the overall player experience especially when narrative isn't present. This will be completed through the evaluation of numerous different decision-making algorithms to address common player issues with AI within the context of an FPS environment. An artefact will be developed alongside the dissertation which will contain one of the decision-making methods, where willing participants will be able to interact with the AI companion and provide primary data to assess the implemented method's effectiveness.

## Aims and Objectives

This project aims to undergo an investigation into how to make effective companion AIs for FPS games to improve the overall player experience with AI companions. During which, several methods for AI decision-making will be evaluated by their relevancy to the project along with their notable use cases in the history of AI in games and their potential effectiveness.

Alongside the dissertation, an artefact will be developed that has one of the evaluated methods implemented and will be used to gather primary data from the input of willing participants who will fill out a questionnaire after playing with the AI companion. Additionally, the artefact will be created using an unfamiliar engine. An additional aim of the project will be to learn how to use Unreal C++ to use as the base of the artefact.

By the end of the project, multiple methods for AI decision-making will have been researched, covering their history of implementation, how they work along with the evaluation of their advantages and disadvantages. Additionally, an artefact will have been created where one of the decision-making algorithms covered and assessed in the literature review will be chosen and implemented within an ideal FPS environment. This is for the purpose of gathering primary data, where willing participants will get a chance to play with the AI agent and experience some of the behaviours and judge whether the AI agent was an effective companion, and this feedback will be provided through a questionnaire. The results of which will be analysed to gauge the quality of the implementation, along with the collective response to the selected decision-making algorithm and any improvements to make it more successful as a companion.

# Literature Review

## Controls of First Person Shooters

The First Person Shooter genre is centred around the concept of having players shoot at targets and view the game from the perspective of their in-game character (Grimshaw, 2007). However, the definition alone does not encapsulate all the sub-genres, games like Apex Legends are almost incomparable with Duck Hunt considering how it is commonly overlooked as an FPS despite fulfilling the defining prerequisites of the genre (Voorhees et al., 2012). Due to the large variety of sub-genres, the core mechanics of gunplay and movement must be discussed.

Gunplay refers to the behaviours, aspects of the guns used by the player in game (Mccabe, 2023). There are many different varieties of gunplay due to the broadness and diversity that FPS games bring, games like DOOM and Counter Strike: Global Offensive (CS:GO) typically restrict players to shooting from the hip (Baglin and Morgan, 2017), while more recent games, like Rainbow Six: Siege and Apex Legends , additionally allow players to shoot while Aiming Down Sights (ADS) for improved accuracy (Mccabe, 2023). Aside from visuals, the behaviour of the recoil and bullets are also considered under gunplay. Guns found in Apex Legends have some inaccuracy while firing from the hip, where they will shoot randomly within a certain range and while ADSing, this inaccuracy is negated heavily and replaced with a general pattern that it roughly follows, which is referred to as the spray pattern of a gun (Mccabe, 2023). CS:GO's gunplay is unique as the behaviour of the gun relies on the player's movement instead where, if the player shoots while moving, the spray pattern will be completely random and borderline impossible to control, however when standing still, the spread of bullets will strictly follow a set pattern (Baglin and Morgan, 2017). These formats have their place relevant to the FPS environment they have been implemented into, FPS games with ADSing in them typically advocate for faster paced gameplay like Call of Duty(Maclean, 2021), while CS:GO's gunplay forces players to play more methodically due to the slower paced gameplay, which emphasises the tactical, team centric aspects of the game (Young, 2021).

Movement controls in FPS games are often shared and the diversification of the mechanics come in the form of additional options. Popular games like Call of Duty have added additional movement options, such as sprinting (Kessner and Cortes, 2023), to be more engaging to the player and provide more ways for the player to interact in fights and keeps the pace of the game very quick(Maclean, 2021). Another example can be found in Team Fortress 2, where they added double jumping as an innate ability for the scout, one of the selectable classes in the game. The sub-genre of movement shooters adds even more to the overall pace through the added freedom and even more options to movement, games like Ultrakill give the player a lot of manoeuvrability both on the ground and in the air, along with dashing to quickly cover short distances (Zamedianskii, 2023). On the opposite side, tactical shooters like Valorant and CS:GO restrict the player's movement to the basics (Zhu and Zhang, 2023), this slows down the overall pace of the game and makes everything more meaningful, making players more methodical and rely on the teamwork to achieve victory(Young, 2021).

For team-based multiplayer games, communication is important for a team to work homogenously and coordinate a victory over their opponents (Korotin et al., 2021). There are multiple ways for players to communicate and one popular method is through the use of Pings (Zheng, Stein and Farzan, 2022). Pings refer to a system of in-game notifications, alerts and markers distributed by

players to call attention to specific things in-game, these markers are only seen by other teammates and can quickly share information through the press of a button (Leviatt, Keegan and Clark, 2016). These pings can also be diverse and are typically tailored to a predetermined set that works within the context of the game. Apex Legends is a recent example of a popular FPS game that also has a ping system, the system appears as a wheel directly on the screen of the player where they can select which of the predetermined pings they wish to place on the environment, they range from locational ones that signify where the team should move towards to more observational ones such as an enemy seen ping (Coviello, 2023). The aforementioned pings are categorised by colour along with having their own symbols to make it easy for players to quickly recognise what information is being relayed to them (Zheng, Stein and Farzan, 2022). Especially for team-based games, ping systems are advantageous to allow for players to establish and pass information between with effective and efficiency (Wuertz, Bateman and Tang, 2017).

## Believability

Believability, in regards to AI, refers to how well the AI fits into the environment of the game and how well it can enhance the immersion of the player, typically referred to as the illusion of intelligence (Dill, 2013). To do so, AI must be presented in such a manner that the player's expectations and preconceived notions of the agent are fulfilled (Emmerich, Ring and Masuch, 2018), for example, if the setting was historical militaristic, a talking cat NPC would break the players immersion.

The AI must also be able to react in accordance with the world around them, which ultimately boils down to the quality of the behaviours and the complexity of the architecture that dictates them (Warpefelt, 2016). Exuding human-like behaviours is a common goal shared amongst most AI (Pretty, Fayek and Zambetta, 2022), actions such as proper obstacle avoidance or shooting back while retreating from an opponent assist greatly and maintain the player's belief in the AI. However, there are some behaviours exuded by players that if implemented for an AI, would break the players' immersion. Players can frequently change where they are heading due to how they change their mind on what they think is important at that moment, but if this was also the case for AI, it would appear as if the AI was flawed and the player would no longer trust its ability to complete objectives (Dill, 2013).

Communication is another aspect important for believable AI, it refers to ways for the AI to convey cognitive understanding of whats happening in the world around it (Bouquet, Mäkelä and Schmidt, 2021). This can happen through multiple ways, in-game chat messages that say exactly what the agent is thinking or an emoticon above the agent's head to give a rough but quick estimation of what they're feeling will increase the overall believability of an AI as it makes them seem more interactive (Bouquet, Mäkelä and Schmidt, 2021). If the AI agent were to quickly change the direction they are headed because they have updated the most important objective to focus on, it would seem like a defective algorithm, however if a message or emoticon were to appear for the player, it would signify them of the change in objective or even explain why, the agent would no longer seem defective or erratic (Dill, 2013).

Curiously, these aspects of game AI clash with what is typically expected of an AI, as instead of efficiently maximising an output like its chances of success, it is more careful to fit in the environment and prevents its ability to play the game well (Dill, 2013). Game AI is intentionally designed in this method as a way to appeal to people, if it was unbeatable and was perfect in everything it did, then the game would no longer be fun for the player (Wetzel and Anderson, 2017). An unbeatable perfect AI teammate would not be enjoyable if it did everything for the player, which is why they are designed to only partially do things for the player (Warpefelt, 2016) and the limit is dictated by the developers.

## Methods Of Decision Making and AI Control

### Finite State Machines

Finite State Machines (FSM) are a conceptual model visualisation of a reactive system (van Gurp, 1999) that can show the effects certain inputs have on it. The system in question is composed of a finite number of states, transitions between the states, conditions for those transitions and it cannot be in more than a single state at any given moment. In regard to game AI, FSM is a method of controlling AI NPCSs where the states represent potential behaviours of the AI (Dill, 2013) it is a popular method for decision-making in AI due to its simple implementation and extensive use within the games industry (Johansson, 2012). Despite the ease of use there are some issues with using FSM, because of its inherent inability to be in more than one state at a time, if an AI had to choose between shooting and retreating whilst being shot at by an opponent, it might not output the best behaviour in a given scenario or could end up rapidly switch between the two most relevant states without being able to do anything, also known as State Thrashing. Additionally, due to the design of FSM, AI agents have no concept of memory as they will naturally forget the previous state or the overall objective (Roberts, 2022). There is also the issue of scalability, as during development of the AI, it is only natural for it to grow in size, however it slowly gets more and more difficult to manage new states and transitions between them (Johansson, 2012).

### Hierarchal Finite State Machines

Hierarchal Finite State Machines (hFSMs) is an improved form of FSM that prevents the need to consistently tend to the singular FSM (Roberts, 2022). This is done by breaking down the large singular FSM into smaller state machines grouped by capabilities and functions, the smaller FSMs have transitions between the groups allowing for states to be reused under different contexts (Roberts, 2022). This prevents some of the structural issues inherently found when constructing an FSM (Dawe, 2013) by making it easier to manage them via groups and alleviating the potential of state thrashing (Millington, 2006).

### Fuzzy Logic

Fuzzy logic is another format for decision-making in AI agents that unlike FSM, can be in multiple states at once, and is designed to cope with the gray areas of decision making by using math (Millington, 2006). Fuzzy logic accepts input data and feeds it into a fuzzifier function, which classifies them under pre-determined labelled categories which are known as fuzzy sets and are labelled using linguistic terms such as hungry or hurt (Roberts, 2022). Defuzzifiers return the

fuzzified data back into concise values with varied levels of weighting towards the fuzzy sets to be used again (Mendel, 1999). This allows an AI agent to perform a retreat manoeuvre while shooting back at the enemy, where it is 70% committed to retreating and 30% committed to shooting at the enemy. Despite the advantages proposed over FSM, fuzzy logic has its own set of issues to deal with, the process of fuzzifying and defuzzifying the data could potentially output an unwarranted behaviour, meaning that it becomes tricky to achieve a desirable balance (Roberts, 2022).

## Scripted AI

Scripted AI is a form of AI control that is heavily debated (Lewis, 2013), as in exchange for the autonomy provided by decision-making systems, it is traded for a set of pre-determined actions set by the developer (Sehrwat and Raj, 2018). Through assuming manual control of the AI agent, the consistency of the action can be guaranteed as it will no longer be up to the decision-making system to perform them, where there is a level of inherent uncertainty. During gameplay, this method is quite limiting as it is highly reliant on the developer having the foresight to script out every potential action that the AI could perform (Roberts, 2024). Despite this, scripting does have a place as it is best used in conjunction with another decision-making system, this will be used for a majority of, if not all, the gameplay sections, and it will then switch to scripted AI during cutscenes as the outcome of the scripted actions and the cutscene can be guaranteed (Rabin, 2013).

## Behaviour Trees

Behaviour trees is another common approach to decision-making, even often considered a staple in game AI. Behaviour trees are comprised of a root node, from which the decision-making begins from, composite nodes and decorators (Roberts, 2022). Composite nodes come in the form of selectors and sequence nodes, which return success, failure or running depending on the result of their child nodes, referring to the nodes owned by it. Selector nodes when run, iterates through all its child nodes and executes the first child node that is successful and returns success, but if its children fail to do so, then the selector will also return failure to its parent node (Millington, 2006). Sequence nodes work in the opposite way, it iterates and executes all of its children nodes until one of them returns failure, if it iterates through all the children nodes and none of them fail, it returns success to its parent node, however if one of them fails during the iteration, then it returns failure (Johansson, 2012). Decorators are attached to singular nodes to add another form of filter in the form of a condition, this adds another layer of control to nodes (Rabin, 2013). The main advantage of behaviour trees is its simplicity, the base algorithm is straightforward, and it is highly modular (Rabin, 2013). Adding new nodes to the tree can be swiftly done and unlike FSMs, the entire structure of the tree does not need to be tailored for the new nodes and old nodes can be easily reused under different contexts (Rabin, 2013).

## Movement, Avoidance, Routing, Planning and Orders (MARPO)

Movement, Avoidance, Routing, Planning and Orders (MARPO) is a system for AI control that arranges actions to be done into tasks and completes them based on a given priority. MARPO completes this by having several stacks with set levels of descending priority where tasks are pushed onto these stacks for the AI to complete by following the order of priority (Roberts, 2024a). Typically, there are 3 stacks and the stack that has the lowest priority usually contains the overall

goal of the AI or longer-term tasks, such as moving to a far location, or idle tasks, like wandering around. The stack with the second highest priority contains tasks which are temporary actions that take precedent over, like getting dressed or eating a snack. The last stack is reserved for emergency actions that must be done immediately, this can be actions like reloading a weapon or running away from an opponent (Roberts, 2024a). During gameplay, due to how the stacks can receive tasks at any moment, the AI is constantly checking the contents of these stacks in order of highest to lowest priority, which makes the AI agents highly reactive and able to have their current task interrupted, then resume the task once the higher priority stacks are empty. The system itself is quite modular as well because to add more actions for the AI agent to perform, all that must be done is to push the new task into the relevant stack (Roberts, 2024).

## Reactive Behaviour Trees

Reactive Behaviour Trees (RBT) are an improvement upon behaviour trees where it uses MARPO inpsired logic to produce goal-centric game AI. RBT works using only two stacks, where one has higher priority than the other, instead of the 3 MARPO uses to reduce processing power and streamlines the process, additionally, instead of tasks being pushed into the stacks, behaviour trees are pushed into them (Roberts, 2024a). The AI will similarly check the stacks in order of priority and will run any behaviour trees pushed into the stacks, it will run the behaviour tree with the lowest priority and if at any point a behaviour tree is pushed into the stack with the highest priority, it will pause the current process to pursue and complete the behaviour trees with higher priority, similar to MARPO. The monitor pushes these behaviour trees into the relevant stacks of the relevant agents, allowing for stacks to contain behaviour trees tailored to the role of the AI NPC. RBT provides a system with small, manageable and reusable behaviour trees that is also modular and efficient. However the main downside is that it can be quite easy to develop large and unwieldy behaviour trees to push into the AI's stacks, which defeats the overall purpose of RBT (Roberts, 2024a).

## Influence Maps

Influence maps are an approach to AI decision-making that uses the agent's perception of the world around it as an input, they require the pre-requisite that there is a changing, quantifiable value that can be extrapolated from the environment (Rabin, 2015). This value can be things such as opponent NPC positions and their power over an area. Using this information, the AI agent's behaviour changes via spatial analysis, where it uses queries to query its environment and values stored in them. These values are propagated through the use of placement functions, which store the values, and diffusion functions, which smear and decay values over distances (Rabin, 2015). The queries represent the goals of the AI agent and must be relevant to the value extrapolated (Roberts, 2022), for example if the query is to find a safe area to hide from opponent AIs, the value would be opponents influence over an area, in which the AI will find the area with the least opponent influence. Although this assists the AI NPC in making highly informed decisions, the limitation comes in the representation of the influence map as a whole. Having a large grid with a fine resolution to represent the game world can be quite taxing on the system memory-wise, as the propagation of cells mandates the updating of each relevant cell along with the 8 cells surrounding it at the minimum, however if the resolution were to get coarser, then the details of the influence map is lost along with the size (Rabin, 2015). It becomes a balancing act of choosing the right size for the environment of the AI.

## Director AI

Director AI is a form of decision making that has one larger more complex AI control smaller AI entities, this format has been used in games like Payday 2. The director AI system is comprised of 2 parts, the micro AI, the smaller entities that are seen in-game, and the macro AI, an omniscient AI that controls the micro AI, he macro AI is omniscient in regards to how it works to control the micro AI, it has knowledge of all the actions done by the player and adjusts the behaviour of the micro AI accordingly (Thompson, 2020). Alien: Isolation uses this system as a way to have an intense and engaging opponent in the Xenomorph. At the start of the game, the xenomorph's behaviour tree has access to a small number of nodes, in reality it has over 100 nodes but is limited by the master AI in what it can do. However, as the player progresses throughout the game, the master AI slowly unlocks more and more behaviours for the xenomorph to do, giving it the illusion of being able to learn from interactions with the player and improving its plan to attack them the next time (Thompson, 2017). Left 4 Dead (Valve, 2008) uses the master AI system to manage its zombie horde, in a dynamic and engaging way. The team has a collective "emotional intensity" value that builds and decays throughout the level and from the actions of the players (Booth, 2009). The director monitors this value and the higher it goes the more difficult it gets to fend off the slowly growing zombie horde, when it reaches the peak, the director summons an all out zombie assault, and if the players survive, they are rewarded with a cooldown, where no zombies are spawned for a certain time (Thompson, 2022).

## Utility Theory

Utility theory refers to the evaluation of possible actions and going with the most option that has the best benefit over cost (Roberts, 2024a). Evaluating the actions relies on extrapolating a number from the current possible actions by judging how relevant or useful those actions would be within a given context, this is referred to as its utility and is distinct from its value, other factors can additionally contribute to the overall utility of something (Rabin, 2013). Calculating the utility of a potential actions can vary, but commonly uses different curves on a graph, however what it ultimately boils down to is understanding the relationship between the benefit of the one action compared to another and its relation to the curve (Rabin, 2013). Utility theory can be used to control an AI where the actions being evaluated are instead substituted for behaviours, for example if a quadratic curve is used to calculate utility, health was on the X axis and willingness to chase an opponent was on the Y, the lower the AI's health number is, the less likely it would be to chase the opponent. This is limited in potential as to get more complex behaviours from the AI NPC, it requires more variables and behaviours to be considered, which is typically illustrated as the addition of more axes, however at some point it becomes difficult to manage (Roberts, 2024b). One method of circumventing it is through the addition of hierarchies, dividing the utility systems by function and behaviours makes it so when comparing benefit over cost of the behaviours, it will only have to do it against the behaviours within the same band instead of every other potential behaviour (Roberts, 2024b).

## Infinite Axis Utility Systems

Infinite Axis Utility Systems (IAUS) is a form of utility theory where it produces an outcome based on weighting all the potential actions against how confident the AI is in performing them. This process is

completed through the use of 4 concepts, reasoners, actions, considerations and axis (Roberts, 2024a). Reasoners refer to the process of recognising all possible actions at the current moment in time and grouping them to be evaluated which is closely linked with the actions concept, this refers to the actual behaviours to be executed by the decision-making system. The potential actions are then scored against the curve on a graph, referred to as the axis concept, and the score returned is the consideration concept. The consideration score is then later weighed against other actions' consideration scores where the one with the highest consideration score has its action run (Rabin, 2013). An example of this in action would be an AI NPC contemplating whether to sleep or to do work. The actions themselves would be sleeping or doing work, the reasoner would encapsulate the 2 currently possible actions of the NPC, these actions would then be scored against the axis curve, and the score returned would be how much more compelling doing work is compared to sleeping for the NPC.

## Character Utlity Behaviour Graphs

Character Utility Behaviour Graphs (CUBG) is a concept born from IAUS and hFSM to create a complex character decision-making system (Roberts, 2024a). In CUBG, the reasoners contain its own FSM group, and the reasoners are interconnected like a hFSM, organising them in such a way that the AI agent will move through them going through the highest level states first and ending with the lowest level states. Considerations, in regard to CUBG, can have special properties like allowing the action its related with to run regardless of the utility theory evaluation, this allows for the AI NPC to be running multiple states at once. This makes the system similar to fuzzy logic in practice and allows for the system to emphasise the AI entity's character (Roberts, 2024a), the NPC has multiple modules that use CUBG to dictate how it should act to emulate emotions like fear or happiness. These modules are made up of the Maslow, personality, core behaviour, events and smart objects, communication between these modules occurs through a blackboard. The Maslow module dictates the basic needs for the AIs survival, while personality dictates the traits of the NPC, where it uses Openness, Conscientiousness, Extraversion, Agreeableness and Neuroticism (OCEAN) to refer to when calculating the emotional offsets for fear, trust, joy, surprise, anger and disgust. During events, the personality module will assist in creating varied reactions by adjusting the reaction of the AI agent (Roberts, 2024b).

## Goal Oriented Action Planning

Goal Oriented Action Planning (GOAP) was developed in the early 2000's by Orkin as a method of AI agent control for the game F.E.A.R. and is a goal-based method of decision-making that assigns tasks to the relevant AI agents for them to complete (Orkin, 2006). This is completed through the use of a planner and tasks. Tasks refer to the actions that need to be completed themselves, they also have preconditions that need to be fulfilled and this information is pushed along with the action itself into a stack, the planner is an algorithm evaluates the game world and monitors the stack for any content, if there is, it will look for available AI agents and give them the task to complete. An important part of the planner is that it tests the feasibility of the available AI completing the task in the current state of the game, and if it cannot it attempts to fulfil the preconditions. The initial task passed in can be interpreted as the overall goal, and the preconditions themselves can also be tasks with their own set of preconditions (Rabin, 2013), resulting in what is known as a Directed Acyclic Graph (DAG). For example, if an AI agent was given the goal of eating at a restaurant, the feasibility

of it is checked through the preconditions in reverse order, has the AI agent eaten yet? Have they ordered and paid for the food yet? Have they reached the location of the restaurant? Do they have enough money for it? The tasks would then result in the AI agent having money for it, then travelling to the restaurant, ordering the food and paying for it, then actually eating it. By modelling the tasks like a DAG, it prevents any loops (Roberts, 2024b) and ensures the AI agent completes the tasks in an order that makes sense , using the previous example, the AI agent logistically cannot order and pay for the food if they are not at the restaurant yet and have no money for it. By using this structure, it provides a system where more tasks can be swiftly added without much maintenance to previous tasks, making it highly modular in practice. Orkin additionally used A*, a form of graph searching algorithm, to help GOAP complete goals efficiently. A* interprets the game world as nodes with actions affecting the state of it and it then changes nodes, if a goal had to be completed A* would check the cost metric of potential methods of reaching a game world state where the goal has been completed and select the method with the lowest cost for completion (Orkin, 2006).

Despite the advantages it provides, there are some limitations to GOAP, with one of the main issues being reactivity. GOAP is goal-based and is centred around the planner which requires time and processing power to plan the next course of actions (Roberts, 2024a), this means that when an event suddenly happens that need to be resolved immediately, it will have to drop the current task then re-evaluate the feasibility of the goal to ensure that sudden event has not prevented the completion of the goal. Another limitation is relative scale to the project, GOAP is more suited to projects with multiple AI agents, as using GOAP for a single AI agent would be highly inefficient (Roberts, 2024b)

# Research Methodologies

## Artefact Creation

To effectively investigate the development of companion AIs for FPS games, primary data will be gathered on one of the methods of AI decision-making. An artefact will be developed using Unreal Engine version 5.4.4 (UE5) as a base platform to test the effectiveness of Behaviour Trees in a recreated FPS environment. By using UE5 as the platform of development, it circumvents much of the initial preparation stage for conducting this format of research, the game engine provides lots of powerful, built-in tools for developers which include, but is not limited to, a 3D space renderer, blueprints to represent modifiable entities and accommodation for both visual scripting and C++ scripts. The most relevant tools provided by UE5 are what it provides in terms of AI, present from the start is pathfinding via navigation meshes and A*, crowd avoidance, blackboards and a behaviour tree system.

Beginning the process of development, the player character needs to be made first, as it will allow for player to interact with the game world and the AI. When creating entities with C++ functionality, a C++ class is required and is composed of a cpp file and a header file. C++ classes contain functions and act as templates for any UE5 blueprints made from it, and blueprints born from it share the same functions written in the C++ class. The controls of the player's character will be created using this and will mirror those commonly used by first-person games, where WASD controls the X and Z axis movement, mouse movement to rotate the camera of the player's character, the left mouse button to shoot, which is purely cosmetic, and space to jump. By using familiar controls, players will be able to quickly grasp how to play it and therefore more time can be spent interacting with the AI. Additionally, the player will be able to place and delete ping blueprints into the world using C and X respectively. This ping system will be the method chosen to improve communication between players and AI companions, allowing it to be controlled to do certain actions under specific conditions. When using the ping system, there will be 3 types of pings and placing them will be contextual to what the player is looking at. There will be a location ping, an enemy ping and a companion ping, and context is found by casting a ray and reading the first thing that comes into contact with it. Location pings are intended to be placed when the player pings an area on the floor, enemy pings are when the player is looking at an opponent and companion pings are when the player is looking at the AI companion.

The companion AI will be developed next and creating the AI companion requires multiple C++ classes, one to represent the entity and the second to represent the AI controlling the entity in game. The blueprint for the entity will carry a reference to the AI C++ class within its AI Controller component along with references to the behaviour tree created for it. UE5s behaviour tree system is quite robust and highly modular as the basic nodes, like decorator and composite nodes, have already been implemented along with other useful tasks such as a wait task and a move to location task. The systems modularity comes in the form of how easy it is to add and remove custom nodes without it heavily affecting the existing tree. The blackboard is a component that stores values and allows for scripts linked to it to modify and update these values. The AIs behaviour tree will be broken down into sections with a natural priority, so during gameplay it checks the behaviours with the highest priority first then flows down to check the sections with lower and lower priority, similar to MARPO. The section with the highest priority would represent emergency tasks such as protecting an endangered player, the second highest priority would consist of activities to do with the ping

system, such as watching over a location or going to a location, and the lowest would be idle tasks like following the player at a distance. When a location ping exists, the AI will look in that general direction and will follow the player, but when both a location ping and companion ping exist, the companion will go to the specified location. When an enemy ping exists, the AI will watch the enemy and follow the player, and when the companion ping is present, the AI will move towards the pinged opponent. The blackboard of the companion AI will hold variables that assist the behaviour tree in its decision making, such as the distance between it and a target, whether a type of ping exists in the world and whether it has line of sight with the opponent AI and the player. The aforementioned opponent AI will be a simple AI that has a random wander function in its behaviour tree and looks at the player if they are within its line of sight.

## Data Collection Methods

The type of data that will be collected will be mixed methods, a combination of both qualitative and quantitative data, where the method of gathering the data will be through questionnaires. Willing participants for the project will be found and will need to complete a few prerequisites before interacting with the questionnaire. Firstly, the participants will need to have read the information sheet relevant to the investigation and sign a consent form which ensures that the participant is informed on what type of investigation they will be partaking in, where the information that will be extrapolated from their questionnaire responses will be used to evaluate the effectiveness of behaviour trees as a method of decision making for an AI companion. After, a unique identifier will be assigned to each of the participants, and this will be marked on their consent form to match it with their questionnaire number. Doing this maintains the participants' anonymity and their right to remove their data, until the 18th of February, where if they wish to do so, the relevant document are identified through the number and destroyed. The willing participants will then be able to play with the artefact and once that is finished, they are to then fill out the questionnaire.

The questionnaire will consist of 7 questions with an extra one dedicated to the unique identifier. The first 4 questions are aimed towards gathering quantitative information, where the questions will be presented to participants with answers on a Likert scale. The first question will be how responsive the AI was to the ping system, with the choices being unresponsive, mostly unresponsive, somewhat responsive, very responsive and highly responsive. The second will be how the AI acted when the ping system was not active, with the answers being, unsupportive, mostly unsupportive, somewhat supportive, very supportive and highly supportive. The third will ask the participant on if the Ai behaved as expected when using the ping system, with the potential answers being, did not behave as expected, behaved slightly as expected, behaved almost as expected, behaved as expected and behaved more than expected. The fourth and final Likert based question will ask how the participants found the potential options for AI commands within the context of an FPS, the options consisted of too many unconcise command prompts, an okay variety of command prompts, a decent variety of command prompts, a good variety of command prompts and not enough command prompts. The last 3 questions aim to gather qualitative information from the participants' experiences of interaction with the AI companion, it asks which of the command prompts were more relevant within an FPS environment with elaboration, which of the command prompts were less relevant within an FPS environment with elaboration and finally how the participant found the AI as a replacement for a teammate with elaboration.

The reason for the investigation using mixed methods research is because it allows for concise but also more meaningful conclusions to be extrapolated from the data. When conducting research to gather purely quantitative data, the answers are simplified down to hard numbers which leads to the data being concise and easily analysed. This could also be interpreted as a limitation, as due to the simplification, it becomes difficult to capture the overall context and complexity of the data, and methods of answering like the Likert scale, partially manipulate the data to be one of the predetermined categories. On the other hand, using qualitative data yields results that are high quality, detailed and allow for elaboration on the topic, but in exchange, the data is harder to analyse and generalise. By conducting mixed methods research, both data types' advantages are present in the data and can complement each other, and the inherent disadvantages have much less precedence.

Mixed methods are the ideal for the project as it allows for participants to provide hard numbers while elaborating on them. The difficulty of mixed methods is the analysis of the data, as it is more difficult and requires more attention, for the purpose of investigating the effectiveness of behaviour trees as a method of AI companion control, reoccurring trends will be extrapolated from the answers and compared against other trends for impact and meaning on the investigation.

## Results and Findings

2. How responsive was the AI to the command prompts?


Figure 1

The graph shown in Figure 1 displays the results of how responsive the AI was to the command prompts. We can observe from the experiences of the 5 participants, that 80% of the time, the AI was very responsive and 20% of the time it was only somewhat responsive.

3. How did you find the AI when it was not directly commanded?


Figure 2

The graph shown in Figure 2 displays the results of how the participants found the AI when it was directly commanded. We can observe from the 5 participants that 40% believed that the AI companion was very supportive, while 60% of them believed that the AI companion was only somewhat supportive.

4. When directly commanded, did the AI behave as you expected?


Figure 3

The graph shown in Figure 3 displays the results of what the participants thought of the AI's behaviour when it had been commanded. 60% believed it behaved as expected, 20% believed the AI companion behaved almost as expected and another 20% believed it behaved slightly as expected.

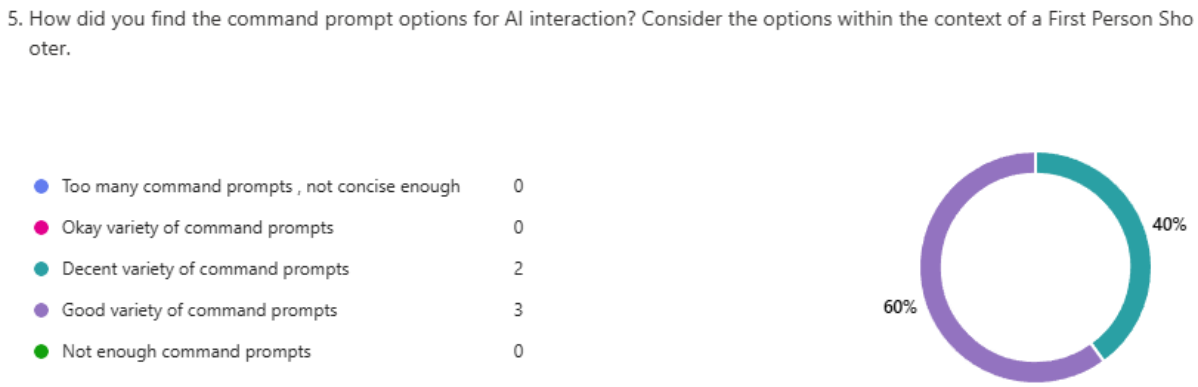


Figure 4

The graph shown above in Figure 4 displays the results of how the participants found the command prompt options for AI interaction within the context of an FPS. 60% believed that there was a good variety of command prompts while 40% believed that there was a decent variety of command prompts.

The graphic shown in Appendix 1 displays the results of which of the command prompts the participants thought were more relevant within an FPS setting with elaboration. One participant said Get behind cover lets the player know when they are exposed to the enemy and the enemy spotted prompt lets the player know of nearby enemies. Another participant believed the prompt to attack the enemy was the highest priority. Two participants believed the enemy ping was the most important prompt and one participant believed the go to location prompt was the most important as it lets players set up the AI for a flank.

The graphic shown in Appendix 2 displays the results of which of the command prompts the participants thought were less relevant within an FPS setting with elaboration. One participant said shooting at you. Another participant believed the prompt that said get behind cover wasn't so clear on its purpose. One participant said the move to a location, while another said the move to enemy prompt was less relevant because within an FPS setting, the AI would most likely just be gunned down if you only had one. The last participant said none of the prompts were less relevant in an FPS setting.

The graphic shown in Appendix 3 displays the results of how the participants found the AI as a replacement for a missing teammate with elaboration. One participant said that it's hard to tell as there isn't much gameplay around the project. Another participant said that it felt like a good companion who you could have fun with, while a different participant said that it was a fun alternative. The fourth participant believed that it was good as a back up, but as for a missing

teammate, more functionality is needed to replicate the missing team's potential ability. The final participant believed the AI was too clingy, but otherwise okay.

# Discussion and Analysis

The overall goal of the project is to investigate the development of companion AIs for an FPS environment to address the common issues players have with AI. To assist in the process of research, an artefact was created to test the effectiveness of a common AI decision-making system, the system being behaviour trees, by getting willing participants to play the artefact and fill out a questionnaire.

## How Responsive was the AI to the Command Prompts?

The first question of the questionnaire asked the willing participants how responsive the companion AI was to the command prompts. This question is aimed to gather quantitative data on the overall quality of the part of the behaviour tree that responds to the ping system, the participants were presented with a Likert scale to choose answers from to provide generalised concise data. This question is important to the project as in games with controllable AI, it is important for the AI to be responsive to commands from the player in a way that is obvious but also makes sense for the games context while mitigating player frustration with AI. The results of the data show that 80% of the participants believed the companion AI to be very responsive whenever the ping system was used, this could be due to how the implementation of the companion AI's behaviour tree handles interactions with the ping system. The companion AI's behaviour tree has nodes structured in such a way that it has a natural priority flow of priority during traversal that somewhat mimics MARPO.

During runtime, the behaviour tree checks for any events and interactions considered as emergencies that need to be dealt with over anything, such as the wandering opponent AI seeing the player, similar to MARPO. It then goes to the second layer of checks where it handles interactions with the ping system, where it looks at the direction of a location ping or an enemy ping, and actually travels to those locations when combined with a teammate ping. Additionally, within the second layer there is also a small hierarchy of what interactions the AI chooses to prioritise as all three types of pings can exist at any moment, where the AI will prioritise going to the location of the wandering opponent to attack them. The final layer handles the idle state of the AI which is just following the player. Implementing the behaviour tree in this MARPO-like format makes the AI highly reactive, interrupting its current actions to execute behaviours with higher priority.

Despite this, the data from the first question also shows that 20% of the participants believed that the AI was only somewhat responsive to the command prompts of the player while using the ping system. A potential reason for this option could be related to the MARPO-like structure of the behaviour tree, where it chooses to deal with events considered as emergencies over the ping system or the idle behaviour. The AI companion is constantly checking if the player is under threat from the wandering opponent, which constitutes the player being within its line of sight to mimic being shot at, during which it goes to the location of the enemy to mimic attacking it and then goes to the location of the player to mimic healing them. This means that if there is a location ping that the AI is currently travelling towards, it will ignore the command in favour of protecting the player. Another reason for this choice could be due to the format of the controls and the participants unfamiliarity with them. Due to the type of ping placed being contextual to what the player is looking at, it can be somewhat difficult to get the correct type of ping the participant intended, additionally, the controls of how to properly use the ping system may not have been clear.

These potential limitations however, open up new avenues for improving on the implementations of behaviour trees as a method of AI control. For the first potential reason, a method to deal with it is to change the order of priorities, where instead of prioritising behaviours that protect the player if they are threatened by the wandering opponent, the behaviours relating to the ping system will be prioritised. Naturally, the behaviours that protect the player will be second in terms of priority and the last will be its idle follow player behaviour. This means that if there are no commands and the AI companion is running its idle behaviour of following the player, if the player goes into line of sight of the wandering opponent, it will go to attack them but if at any point during that process the ping system is used, the process will be interrupted to pursue the commands instead.

## How did you find the AI when it was not directly commanded?

The second question posed to the willing participants was how they found the AI when it was not directly commanded. This question seeks to gather quantitative data on the quality of the behaviours in the behaviour tree that do not interact with the command prompts, and the answers the participants could choose were generalised using a Likert scale. It is important to ask this for the project as games with NPCs that support the player need to have a use to that is clear to the player and assist them in a way that does not obstruct the player. The results of the data show that 40% of the participants believed that the companion AI was very supportive when the ping system was not used to control its actions, this is potentially due to the behaviours the exuded by the companion AI at the top layer of priority and lowest layer of priority.

The companion AI's behaviour manages when to follow orders from the player and when to use its autonomous behaviours. Its autonomous behaviours reside on the first layer of priority checks and the last layer, where it checks if the player is threatened by the wandering opponent and follows the player respectively. When the player is threatened by the wandering opponent, it means that they have entered the wandering opponent's line of sight and when that happens, the AI companion goes to the location of the wandering opponent to mimic attacking them, once the player breaks the wandering opponent's line of sight, the companion AI will travel to the player's location to mimic healing them. When the player is not under threat and there are no pings currently present, the behaviour tree will default to its idle behaviour of following the player.

From the results however, 60% of the participants believed that when the ping system was not used to control AI, its behaviour was somewhat supportive. A possible reason for this is due to the implementation of the companion AI's autonomous behaviours. When the AI intends to protect the player by attacking the wandering opponent, all it really does is go to the wandering opponent's current location and stand there, the same case is applicable for when it returns to the player to heal them. The issue is that there is not a lot of telegraph as to what the AI companion is actually doing other than a debug message being sent in the top left of the screen.

The method for tackling this potential issue is to add more functionality into the artefact. Due to the nature of project where research on how to utilise Unreal Engine 5 and it's C++ scripts, the potential of what could be done was limited, leading to a lot of details being stripped and sacrificed to

streamline the process. Adding visual animations and sound cues would assist in providing the players multiple methods of feedback to clue in what the AI is doing.

## When directly commanded, did the AI behave as you expected?

The third question of the questionnaire asked the participants if the AI behaved as they expected when using the ping system. This question is aimed to gather quantitative data on the overall quality of the behaviours concerned with the ping system and presents the participants with generalised answers in a Likert scale fashion. This is important as AI NPCs that can have their actions be influenced by the player requires the influenced behaviour to fulfil the expectations of the player, as it can lead to disappointment and a break in immersion for the player if it only partially does the action. The results from questionnaire reveal that 60% of the participants behaved as expected, which could be due to the content of the behaviours themselves.

How the behaviours regarding the location ping work is that, on the second layer of checks for the behaviour tree, if the location or enemy ping exists, it will look in the direction of the in-game ping but will still physically follow the player. This works by using a dedicated node for checking the status of any pings, where if one exists, it will change a Boolean dedicated to it from false to true and will store its in-game location to a vector to the blackboard of the AI companion's behaviour tree. The next node then moves the AI towards the ping, in the case for moving towards the wandering opponent, the distance between them and the AI is calculated and while the distance between them is above a certain threshold, the AI will move to a point that is halfway between them.

Curiously, 20% of the participants believed that the AI believed almost as expected and the last 20% of the participants believed that the AI behaved slightly as expected. A possible reason for this judgement is that it could stem from the way the ping system was implemented and participants' unfamiliarity with it, as to influence the actions of the AI, the player must combine a teammate ping and a location ping or an enemy ping, where it will go to the location, and if all three types exist at once, the AI companion will prioritise going towards the enemy over the location. Because of the way that the ping types are placed in the world through context, there is a chance that the type of pings placed by the user were unintended and require some getting used to, in order to be proficient using it. Another potential reason could be due to the wording of the answers to the question being not as well worded and difficult to work with.

To address the potential issues posed, there could be text on screen to remind the player of what combinations of pings do what and the question could also be worded better where instead of options such as behaved almost as expected, it will get the participants to choose a number where 0 represents the AI not behaving as expected and 5 being the AI behaved as expected.

## How did you find the command prompt options for AI interaction?

The fourth question of the questionnaire asked the willing participants how they found the ping system, within the context of an FPS. This question is aimed to gather quantitative data on the quality of the ping system and the options presented to the player. This is important as it is the

method of communication between the player and the companion AI, and questions if the implementation had precedent to be implemented into an FPS environment. The results of the data show that 60% of the participants believed it had a good variety of prompts and this could be due to the way that the ping system is controlled by the player.

As previously stated, the way that the ping system works is that it is contextual and decides what type of ping to place in the world depending what the player is looking at when they press C. To do this, when C is pressed, a ray is cast and it gets the first actor that comes into contact with it, and depending on the actor type, it places the ping most relevant to it. The type of behaviours that the AI exudes when a ping exists includes, look at this direction which is useful for spotting enemies at a distance, look at this enemy which is useful in FPSs for notifying the presence of an opponent, go to location which makes the AI companion go to the pinged location and go to opponent which makes the AI companion go to the location of the opponent to mimic attacking.

The other 40% of the participants however, believe that within the context of an FPS, there were only a decent variety of ping options. A possible reason could stem from a similar sentiment of a lack of functionality, as in the artefact, there is no presence of actual shooting between any of the NPCs and the player. This limitation in functionality means that it was difficult to telegraph what was meant to be happening between the staring contests of the Ais and the player.

These limitations however give a clear goal of how to improve upon the project, where if more functionality was added, such as a proper shooting mechanic, it would allow for more options such as distinctions between a look at opponent ping and a shoot at opponent ping, a throw grenade at this area option. More options such as these would allow for an overall better experience for participants and more interactions between the AI and the player.

### Which command prompts did you think were MORE relevant in an FPS setting?

This is the fifth question of the questionnaire, and it asked the willing participants which of the command prompts they thought were the most relevant for an FPS. This question is aimed to gather qualitative data to get more context on the quality of the ping system as participants were able to elaborate on their answers. A question like this is important to the project, as it is not only surveying the types of pings the player can do but also the quality of their implementation.

60% of the participants mentioned the enemy ping being the most relevant command prompt. This is potentially due to the environment that the investigation is being conducted in, FPS games are also commonly known for being team centric and through the enemy ping it lets teammates become notified of the presence of nearby opponents, including the AI companion. 20% of the participants specifically mentioned the attack enemy command, which occurs when the player pings both the companion and the enemy, saying it was definitely a priority. This answer could share the same line of thinking as the other 60% who mentioned the enemy ping, where the context of the investigation must be considered. This is because FPS games are typically centered around eliminating opponents, therefore by performing the set of actions to usher the AI to attack the opponent, players not only

communicate to teammates the presence of an enemy, the companion is also going to be sent to attack them. 20% of the participants believed the go to location prompt was the most relevant in an FPS environment, reasoning it with the fact that it allows the AI to be set up to flank opponents. This could be because of the potential impact of flanking not just in shooters but also team based games, flanking is a form of attacking opponents in a way that is powerful as it is a sneak attack to catch the opponents by surprise, and it also splits the attention and attacking power to be directed to an additional direction, which could potentially lead to the opponents' team getting overwhelmed.

Curiously, one of the participants that said that the enemy ping was highly relevant in an FPS environment also argued that the get behind cover quip of the companion AI was quite important for FPS games. The mechanic that they are referring to however is not considered within the command prompts that the player is capable of, but instead they are talking about one of the autonomous behaviours of the AI. The AI companion checks for whenever the player has entered the wandering opponent's line of sight, which is considered as the player being under threat, and if that occurs it defies any commands given by the player and proceeds to attack the opponent. The participant that who stated this could have potentially mentioned due to the lack of information given by the artefact and the question as to what was considered a command prompt and what was not.

To prevent any confusion in naming conventions to occur again, what is considered a command prompt must be clearer to participants, potentially within the artefact itself such as text that denotes them, and also within the content of the questionnaire.

### Which command prompts did you think were LESS relevant in an FPS setting?

The sixth question of the questionnaire attempts to gather qualitative data on which of the available command prompts the willing participants were less relevant within the context of an FPS. This question was made to target information on which of the available command prompts would see less or even no usage. A question like this is important to the project as due to all types of ping being tied to a single button that searches for context, it is imperative that only the most useful commands are available for it, by removing the commands that were consdiered less useful, it would allow for better options to be added in it's place.

20% of the participants believed that none of the command prompts that they had access to were less relevant within an FPS setting, which could potentially be attributed to the basic but still flexible and applicable behaviours that the AI could exude when commanded by the player. Location pings have their use in directing the companion AIs attention towards a certain direction or by getting them to travel to that location. On the other hand, enemy pings have their place in diverting the companion AIs attention towards the opponents and developing it into a potential attack on them. 20% of the participants argued that the move to location command was less relevant while another 20% argued in that the move to enemy command being less relevant, elaborating on it where the participant said that the companion would most likely get gunned down. Both lines of thinking could stem from the lack of base functionality, where although there is space for the behaviours to be slotted in neatly, all it currently boils down to is pretending through staring competitions, the answer could potentially change if a proper shooting mechanic was implemented with feedback.

40% of the participants argued that mechanics not even considered under the command prompts were less relevant. Half of the participants in the bracket of outlier answers stated the quip said by the wandering opponent to denote that the player was within the opponents line of sight, while the other half said the companion AIs autonomous behaviour to go attack the opponent whenever the player was endangered didn't have a clear purpose. Both of these could be attributed to the lack of communication in both the artefact and the questionnaire as to what mechanics counted as a command prompt.

The aspects of the project that must be improved would be to add more functionality to the base game, such as shooting, to make the move to location more relevant, and also to communicate to the participants better as to what the questionnaire was looking for.

## How did you find the AI as a replacement for a missing teammate?

This was the final question asked by the questionnaire for the willing participants to answer, which which asks how they found the AI companion as a replacement for a missing teammate. This is an important final question as it calls into question the participants entire experience with the artefact to judge not just the companion AI, but also its decision-making system, its ability to communicate and respond to the player's commands, it's overall implementation and it's quality.

40% of the participants had generally positive answers mentioning the fun of an AI companion, half of them regarded the AI companion as a fun alternative while the other half said the AI felt like a good companion who you could have fun with. This could potentially be due to the effectiveness of the implementation of behaviour trees, making the AI highly responsive and reactive to the player and the world around it. Another 20% of the participants had a positive response to it with some criticism, where it was deemed good as a back up. But, they also did point out the lack of functionality where they agree with another 20% of the participants that said it was hard to tell if the AI companion was good as a teammate substitute due to the minimal gameplay, this insinuates that for what the project was it was good and fun, however more functionality was required to develop the potential of the AI.

The final 20% of the participants said that the AI was too clingy, but otherwise was okay. What could be extrapolated from this comment is that despite being okay, there were some critiques to the idle follow player function where it could potentially be physically obtrusive to the movement of the player. This critique could potentially be circumvented through slight changes to the distance the companion AI follows the player. Currently the AI has a distance check between its current position and the player's position, the change would come where if there is much more open space, the AI companion would float around the player at a further distance, but in the event that it follows the player into a small area, the follow distance would shrink accordingly.

Overall, what can be deduced from the questionnaire data is that the behaviour trees as a method for controlling AI companions within the context of an FPS environment is largely effective. The data suggests that it was effective at performing the behaviours relevant to what was happening in the

game world, and the inclusion of the command prompts and ping system provided a form of influence over the AI. There are some limitations to this, with the main critique being the quality of implementation, as despite participants calling it a fun and having potential, the environment the companion AI was implemented into was largely barebones and missing some key features of FPS games, such as a proper shooting mechanic, restricted a lot of the judgement of the participants and the potential.

# Conclusion

An investigation into the development of companion AIs for FPS games was launched to confront the common issue of player frustration with AIs. Conducting such a study could potentially assist AIs by reducing playre frustration and allow for more companion AIs to become more common in modern FPS games akin to Apex Legends and Call of Duty, most likely as substitutes for teammates. Over the course of the investigation, important aspects of AI creation were researched, which include the importance of believability and decision-making techniques. Believable AI was covered due to its importance in maintaining player immersion and expectations in such a way that it does not ruin them, with techniques such as communication and. Decision-making techniques was inherently the most relevant aspect of AI creation to this study, different systems and techniques were researched and evaluated with many of them such as FSM, GOAP and behaviour trees being considered staples of the topic. To assist in the investigation, an artefact was created with the sole purpose of implementing one of the methods of decision-making into an FPS environment to gather data on different aspects of it. The chosen method of AI control was behaviour trees due to its popularity and it being considerd a staple of AI decision-making. Additionally, a popular method of communication, in the form of a ping system, was implemented as a way to control and influence the AI's actions through commands.From the investigation, behaviour trees can generally be considered effective as system for AI decision-making, especially when integrated with a system of AI control such as commands. Behaviour trees are very simple system that is also highly modular and can be easily tailored for the environment of the AI and the overall goals of the AI.

Overall, many of the initially set objectives have been met, as lots of research went into decision-making where new, unfamiliar methods were learnt about and researched, while original understandings of previously known systems for AI control was greatly improved and deepened. The documentation of the in-depth research was highly successful along with the implementation of behaviour trees into the artefact. Another aim was met where an originally unfamilar engine was researched and used to create the artefact, where a familiarity with Unreal Engine 5 and its C++ integrations was well developed and can now be comfortably used. Despite this there were some aims that were attempted that were not met by the end of the study. Although time and research was put into learning how to use Unreal Engine 5 and its C++ integration, the aim of creating an ideal FPS environment was not met, with the ramification of it affecting some of the primary data gathered on behaviour trees and ping-based communication systems.

## Recommendations

In the event that this investigation was to be launched again or improved upon, one of the main aspects that needs to be addressed is the lack of functionality in the artefact as a whole. Despite it meant to be a representation of an ideal FPS artefact, it is missing the main feature of shooting. Adding a proper shooting mechanic for the player, the companion AI and the wandering opponent would assist in telegraphing what is meant to be happening better to participants who are unfamiliar with the code.

Additionally, the artefact is heavily reliant on Unreal Engine 5's built-in behaviour tree system. If another attempt was to be made on this investigation, a change would be to make the behaviour trees no longer reliant on it and purely C++ script based, this would allow for the process to be largely replicable on different C++-based programs.

Another change that could be made would be the method of AI control. The behaviour tree in the artefact was built in such a way that it was similar to MARPO, where there were inherent priorities to the layers of nodes. Ideally implementing MARPO would be better, as it is a goal-based decision-making system that allows for the AI to also be reactive due to the way it is set up to have the emergency tasks be dealt with first, current tasks be dealt next and idle tasks to be completed last. It would fit in with the overall vision of a supportive companion AI, where it prioritises tasks related to the ping system, then autonomous behaviours like protecting the player if they were in danger and the idle behaviour with the lowest priority would be the follow player task.

# References


*Alien: Isolation*. (2014). PlayStation 3 [Game]. Tokyo, Japan: Sega

*Apex Legends*. (2019). PC [Game]. Redwood City, CA: Electronic Arts.

Baglin, S. and Morgan, J. (2017). *Random Numbers and Gaming Random Numbers and Gaming Sinjin Baglin*. [online] Available at: https://scholarworks.sjsu.edu/cgi/viewcontent.cgi?article=1006&context=art108.

Booth, M. (2009). *The AI Systems of Left 4 Dead*. [online] Available at: https://steamcdn-a.akamaihd.net/apps/valve/2009/ai_systems_of_l4d_mike_booth.pdf.

Bouquet, E., Mäkelä, V. and Schmidt, A. (2021). Exploring the Design of Companions in Video Games. *Academic Mindtrek 2021*. doi: https://doi.org/10.1145/3464327.3464371.

*Call of Duty 4: Modern Warfare.* (2007). PlayStation 3 [Game]. Santa Monica, CA: Activision.

*Counter Strike: Global Offensive.* (2012). PC [Game]. Bellevue, WA: Valve Corporation.

Coviello, H. (2023). *Reinventing the (Ping) Wheel: How Apex Legends revolutionized communication in battle royale games*. [online] Available at: https://www.loopbreak.gg/features/apex-legends-ping-system-deep-dive/.

Dawe, M., Gargolinski, S., Dicken, L., Humphreys, T. and Mark, D. (2013). *4 Behavior Selection Algorithms An Overview*. [online] Available at: https://www.gameaipro.com/GameAIPro/GameAIPro_Chapter04_Behavior_Selection_Algorithms.pdf.

Dill, K. (2013). *What Is Game AI?* [online] Available at: http://www.gameaipro.com/GameAIPro/GameAIPro_Chapter01_What_is_Game_AI.pdf

*DOOM*. (1993). PC [Game]. University of Wisconsin, WI: id Software.

*Duck Hunt*. (1984). Nintendo Entertainment System [Game] Kyoto, Japan: Nintendo

Emmerich, K., Ring, P. and Masuch, M. (2018). I'm Glad You Are on My Side. *Proceedings of the 2018 Annual Symposium on Computer-Human Interaction in Play*. doi: https://doi.org/10.1145/3242671.3242709.

Grimshaw, M. (2007). *Sound and Immersion in the First-Person Shooter*. [online] *wlv.openrepository.com*. University of Wolverhampton, School of Computing and Information Technology. Available at: https://wlv.openrepository.com/handle/2436/35995.

Johansson, A. (2012). *Affective Decision Making in Artificial Intelligence : Making Virtual Characters with High Believability*.

Kessner, T.M. and Cortes, L.P. (2023). Mechanics and Experience in *Call of Duty: Modern Warfare*: Opportunities for Civic Empathy. *Simulation & Gaming*, 54(2), p.104687812311561. doi: https://doi.org/10.1177/10468781231156187.

Korotin, A., Stepanov, A., Lange, A., Dmitry Nikolaev and Andrey Somov (2021). Assessment of Video Games Players and Teams Behaviour via Sensing and Heterogeneous Data Analysis: Deployment at an eSports Tournament. *Lecture Notes of the Institute for Computer Sciences*, [online] pp.409–421. doi: https://doi.org/10.1007/978-3-030-76063-2_28.

Leavitt, A., Keegan, B. and Clark, J. (2016). Ping to Win? Non-Verbal Communication and Team Performance in Competitive Online Multiplayer Games. [online] doi: https://doi.org/10.1145/2858036.2858132.

*Left 4 Dead*. (2008). PC [Game]. Bellevue, WA: Valve Corporation.

Lewis, M. (2013). *Plumbing the Forbidden Depths Scripting and AI*. [online] Available at: http://www.gameaipro.com/GameAIPro/GameAIPro_Chapter16_Plumbing_the_Forbidden_Depths_Scripting_and_AI.pdf.

Maclean, E. (2021). *Fast Paced and Action Packed: The Temporality of Masculinity in Shooter Videogames*. [online] Available at: https://digraa.org/wp-content/uploads/2021/02/DiGRAA2021_paper_29.pdf.

Mccabe, A. (2023). *Community Perception of Gunplay in Modern FPS games within Call of Duty: Warzone and Rainbow Six: Siege Faculty of Arts Department of Game Design*. [online] Available at: https://www.diva-portal.org/smash/get/diva2:1771241/FULLTEXT01.pdf.

Mendel, J.M. (1995). Fuzzy logic systems for engineering: a tutorial. *Proceedings of the IEEE*, 83(3), pp.345–377. doi: https://doi.org/10.1109/5.364485.

Millington, I. (2006). *AI for games*. Boca Raton: CRC Press.

Orkin, J. (2006). *Three States and a Plan: The AI of F.E.A.R.* [online] Available at: https://gdcvault.com/play/1013282/Three-States-and-a-Plan

*Payday 2.* (2013). PC [Game]. Calabasas, CA: 505 Games

Pretty, E.J., Fayek, H.M. and Zambetta, F. (2022). A Case for Personalised Non-Player Character Companion Design. *SSRN Electronic Journal*. doi: https://doi.org/10.2139/ssrn.4123761.

Rabin, S. (2013). *Game AI pro : collected wisdom of game AI professionals*. Boca Raton: Crc Press/Taylor & Francis Group.

Rabin, S. (2015). *Game AI Pro 2 : collected wisdom of game AI professionals*. Boca Raton: Crc Press.

Rabin, S. (2017). *Game AI Pro 3 : collected wisdom of game AI Professionals*. Boca Raton, Fl: Crc Press, Taylor & Francis Group.

*Rainbow Six: Siege.* (2015). PC [Game]. Montreal, Canada: Ubisoft

*Resident Evil 4*. (2005). GameCube [Game]. Osaka, Japan: Capcom

Roberts, P. and Nicholas Dent (Illustrator (2022). *Artificial Intelligence in Games*. CRC Press.

Roberts, P. (2024a). *Game AI Uncovered: Volume One*. CRC Press.

Roberts, P. (2024b). *Game AI Uncovered: Volume Two*. CRC Press.

Roberts, P. (2025). *Game AI Uncovered: Volume Tree. CRC Press.*

Sehrawat, A. and Raj, G. (2018). *Intelligent PC Games: Comparison of Neural Network Based AI against Pre-Scripted AI*. [online] IEEE Xplore. doi: https://doi.org/10.1109/ICACCE.2018.8441745.

Thompson, T. (2017). *The Perfect Organism: The AI of Alien: Isolation*. [online] Game Developer. Available at: https://www.gamedeveloper.com/design/the-perfect-organism-the-ai-of-alien-isolation.

Thompson, T. (2025). *AI 101: Managing the Experience with Director AI*. [online] Aiandgames.com. Available at: https://www.aiandgames.com/p/ai-101-managing-the-experience-with

Tremblay, J. and Verbrugge, C. (2013). *Adaptive Companions in FPS Games*. [online] Available at: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=d9f0b72174229a15d04d79d4ec0316ba5e5405c4.

Voorhees, G.A., Call, J. and Whitlock, K. (2012). *Guns, Grenades, and Grunts*. Bloomsbury Publishing USA. Available at: https://pdfcoffee.com/qdownload/guns-grenades-and-grunts-first-person-shooter-games-pdf-free.html

Warpefelt, H. (2016). *The Non-Player Character -Exploring the believability of NPC presen- tation and behavior*. [online] Available at: https://su.diva-portal.org/smash/get/diva2:912617/FULLTEXT01.pdf.

Wetzel, B. and Anderson, K. (2017). *4 What You See Is Not What You Get Player Perception of AI Opponents*. [online] Available at: http://www.gameaipro.com/GameAIPro3/GameAIPro3_Chapter04_Player_Perception_of_AI_Opponents.pdf

Wuertz, J., Bateman, S. and Tang, A. (2017). Why Players use Pings and Annotations in Dota 2. doi: https://doi.org/10.1145/3025453.3025967.

Young, S. (2021). *The Aquila Digital Community Dissertations Spring 2021 Professional Counter-Strike: An Analysis of Media Objects, Esports Culture, and Gamer Representation*. [online] Available at: https://aquila.usm.edu/cgi/viewcontent.cgi?article=2996&context=dissertations.

Zamedianskii, D. (2023). *3D Movement shooter with spider mechanics*. [online] Available at: https://dspace.cvut.cz/bitstream/handle/10467/108643/F3-BP-2023-Zamedianskii-Dmitrii-3D_Movement_shooter_with_spider_mechanics.pdf?sequence=-1&isAllowed=y.

Zheng, K., Stein, B. and Farzan, R. (2022). Use Ping Wisely: A Study of Team Communication and Performance under Lean Affordance. *ACM transactions on social computing*, 5(1-4), pp.1–26. doi: https://doi.org/10.1145/3557022.

Zhu, C. and Zhang, Y. (2023). A First-Person Shooter Game Designed to Educate and Aid the Player Movement Implementation. *Machine Learning and Soft Computing*, [online] pp.37–48. doi: https://doi.org/10.5121/csit.2023.130203.

# Appendices

## Appendix 1

6. Which command prompts did you think were MORE relevant in a First Person Shooter setting? List which prompts and explain why.

| Responses |
| --- |
| Get behind cover - tells the player they are exposed to the enemy, Enemy Spotted - Communicates to nearby players that an enemy has been spotted |
| Attack the enemy is definitely a priority |
| ping enemy |
| Defo the Go Towards prompt as you can set up the AI for a Flank |
| Pinging an enemy |

## Appendix 2

7. Which command prompts did you think were LESS relevant in a First Person Shooter Setting? List which prompts and explain why.

| Responses |
| --- |
| shooting at you |
| The get behind cover wasn't so clear on its purpose |
| move to location |
| Maybe the move to enemy, in an FPS setting, the AI would most likely just be gunned down if you only had 1 |
| None |

## Appendix 3

8. How did you find the AI as a replacement for a missing teammate? Explain your thoughts.

| Responses |
| --- |
| its hard to tell as there isn't much gameplay around the project |
| Felt like a good companion who you could have fun with |
| A fun alternative |
| It was good as a back up, but as for a missing teammate, more functionality is needed to replicate the missing team's potential ability |
| Too clingy, but otherwise OKAY |