2023 IEEE Symposium Series on Computational Intelligence (SSCI)
Mexico City, Mexico. December 5-8, 2023

Comparing Behaviour Tree and Hierarchical Task Network Planning methods for their impact on Player Experience

Alexander Kedalo*, Andrey Zykov†, Hamna Aslam ‡, and Manuel Mazzara§
Artificial Intelligence in Games Development Lab and Institute of Software Development and Engineering
Innopolis University
Email: *a.kedalo, ‡a.zykov, §h.aslam, ¶m.mazzara{@innopolis.ru}

***Abstract*—The AI in games has a large impact on the player's experience, but the large variety of available AI implementation methods makes it difficult to determine which one(s) to use in any particular project, and the differences in their impact on players are mostly unstudied. This paper presents a comparative study to analyse the effects of Behaviour Tree AI and Hierarchical Task Network Planning AI on players experiences. The study participants (players) were given two prototypes of a third-person shooter game, each utilising different AIs, to play and give feedback on. According to the results obtained, players did not notice any major differences between the two prototypes, leading us to believe that the Behaviour Tree AI may be a better solution in most cases, as it is easier to implement.**

***Index Terms*—artificial intelligence, player behavior simulation, effects on player experience, behavior tree, hierarchical task network planning**

I. Introduction

The gaming industry is relatively young and extremely fast-growing. It is expected to reach a 339.95 billion market value by the year 2027 [3] and can be divided into the following segments : PC, Console, Social/Casual (Mobile); with Mobile having the largest share of the market and set to have the highest growth [14].

Those segments target different customers and utilize specific sets of technology. However, artificial intelligence (AI) is one of the things they all have in common.

To cover all the responsibilities of AI, multiple AI methods have been invented over the years of game development. Those methods range in complexity from simple finite-state machines that change their state in response to the environment [13], to real-time planning that formulates and executes a plan for an AI agent [10] [11], development of AI players [9] [5], machine learning to train virtual racers based on the players performance [17], and evolutionary computation which generates a population and only allows pops with desirable traits to reproduce [15]. There have been studies conducted on the performance and applications of various AI implementation methods [1] [16]. However, few studies have done methods' comparisons, with the closest found example being a comparison of three games with AI planning [4], and a study on extending a planning AI with behavior tree [12]. Hence, this paper compares and contrasts two AI implementation methods and established a foundation for further research. For the purposes of the study we have selected two Ai implementation methods most often used today in first- and third-person shooters - Behavior Tree and Hierarchical Task Network Planning. Both methods are utilized in shooter games, have similar features, have noticeable difference in implementation complexity and aim to provide comparable experience for the player.

II. Methods Overview

A. Behavior Tree

A behavior tree is a tree data structure that consists of different nodes [8]:

1) Root Node - execution starting point
2) Leaf Node - node that contains AI behavior
3) Selector Node - node that decides which child node to execute based on outside conditions
4) Sequence Node - executes all child nodes in sequence
5) Decorator Node - manipulates the existing logic of a node

The tree works by saving a reference to an active node, and having this node process the tick. The tree reevaluates its actions if the behavior of the active node reaches its end or if it is influenced by outside factors, such as player actions or changes in the environment. Additionally, behavior trees utilize "blackboards" which store data relevant to the tree decision-making process in order to avoid recalculation and can be shared between multiple trees. The behavior tree provides a streamlined overview of the behavior logic thanks to its tree structure, thus making it easier to scale. This also results in increased ease of use by designers, as they can easily see the structure of the tree and trace any bugs and unwanted behavior. The behavior trees can be constructed based on already existing nodes, with programmers required only for additional behavior. In addition, trees provide high reusability as nodes can be created to account for multiple possibilities. Lastly, the utilization of blackboards and the absence of a requirement to update every frame reduces computation overhead and memory utilization.

Behavior Trees are commonly used in large-scale projects that have a large number of entities present and interacting between themselves at any point in time. They have effectively replaced finite state machines (FSMs) as the prime method of implementing an AI behavior. Nevertheless, in some cases, FSMs or a combination of both [7] can still be an optimal choice.

Examples of behavior trees can be found in Tom Clancy's The Division, Halo 3, Bioshock Infinite, and Alien: Isolation.

B. Hierarchical task network planning

Hierarchical task network planning (HTNP) is a method of AI implementation that relies on creating and executing a plan for an AI agent. It was created as a less resource-demanding and more designer-friendly alternative to real-time planning. HNTP consists of a planner, a domain, and a world state [7].

The world state stores information about the surrounding world. It is updated via the NPC's sensors and stores only the information required for the HNTP decisions. The domain contains tasks. Tasks can be compound or primitive. Primitive tasks are simple sequences of steps that can be performed by an NPC. They have a set of conditions required for the task's execution and a set of results that describe how the execution of the task will affect the NPC's world space. This allows the planner to "see" the future and create more optimized plans. Primitive actions are further divided into operators. Operators are the most basic actions that can be performed by an NPC, such as walking to a point or executing a light attack.

Compound tasks are high-level tasks that can have multiple ways of being achieved. They include several methods that can be chosen by the planner depending on the world state. The method itself can include a set of conditions for execution and a set of tasks (both primitive and compound). Compound tasks form a hierarchy that forms a domain.

The planner formulates a plan from the tasks present in the domain. It starts with the main compound task and decomposes it into smaller tasks. The planner creates a copy of the world state in which it simulates the effects of the selected tasks. The planner utilizes a depth-first search to formulate a plan and goes through the hierarchy, checking conditions and reverting to the nearest compound task if necessary. The end result is a set of primitive tasks or a conclusion that no plan is available. The plan is reformulated if the NPC fails or finishes its current plan, if it has no plan, or if its world state is changed via sensors.

HNTP is able to react to world changes during planning with the help of expected effects. Expected effects allow designers to execute specific tasks if some condition is satisfied without re-evaluating the whole plan.

HNTP allows designers to exercise some level of control over the NPC's actions and patterns of behavior instead of allowing NPC's to fully govern themselves. It also shows better CPU performance in comparison to RTP and provides longer plans faster. HNTP can be found in Transformers: Fall of Cybertron, Killzone 3 [4].

III. Study Design - Goal and Setup

The goal of this study is to determine the impact of the two aforementioned AI methods on the player's experience and perception of the AI entities.

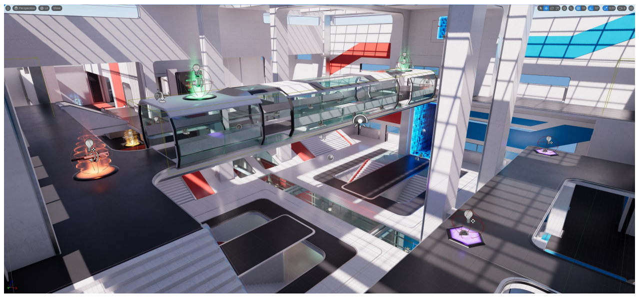

Fig. 1. BT Arena

A prototype of a third-person shooter game is created [1], along with a set of scenarios to be completed by the AI controlled character. Both the prototype and the recordings of the scenarios are provided to the test group of players without notifying them of the AI implementation methods used. Players were asked to interact with the prototype and study the recording. After that, players were given a questionnaire to fill out. The questionnaire is as follows:

1) Does AI in games impact your immersion and overall enjoyment of the game? If yes - how strong of an impact does it have in comparison to core gameplay, graphics, sounds, etc.?
2) On a scale of 1 to 5, how would you rate the AI's likeness to a human player? Where 1 - the AI clearly follows a set of instructions and is extremely predictable, and 5: the AI can be mistaken for a human player.
3) Can you name any noticeable patterns that the AI follows regularly?
4) Can you name some circumstances (e.g., being shot at, being low on ammunition, etc.) that influence the AI's decisions?
5) On a scale of 1 to 5, would the circumstances from the previous question affect your decisions? Where 1 - I would disregard those circumstances, and 5 - those circumstances will dictate my actions.
6) Can you propose any improvements that will make the AI's behavior more human-like and believable?

IV. Preparation

A. Behavior Tree Prototype

We utilised an asset base provided by Lyra Starter Game [2]. The game action commences in a multilayered arena with several weapon and health spawns, teleports and booster platforms. The arena is shown in Figure 1

The AI works based on the behavior tree created through the use of the built-in Unreal Engine toolkit. Behavior Tree requires a blackboard to function. Blackboard stores the data and variables required by the behavior tree. In our case, the blackboard contains the following keys:

1) SelfActor
2) TargetEnemy

[1]link to game build, https://shorturl.at/bijqU and https://shorturl.at/giEYZ

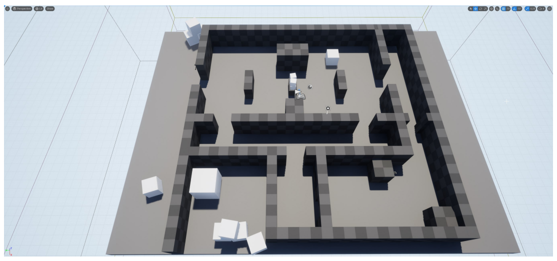

Fig. 2. HTNP Game Map

3) MoveGoal
4) OutOfAmmo

The used behavior tree provides the AI with several options in terms of available actions (BT graph tree link [2]):

1) Find and move to the best position while firing at the target.
2) Look for weapons if no target is detected or low on ammo.
3) Look for targets if no target is detected and has enough ammo.

B. Hierarchical Task Network Planning

For the prototype, we utilise the Hierarchical Task Network Planning AI [6] asset that provides us with a code base for implementing HTNP based bots.

The game's action commences in a simple arena with one main area, additional pathways, and movable obstacles. The HTNP arena can be seen in Figure 2.

The AI will work based on a Hierarchical Task Network created through the use of features provided in the HTNP AI plugin. The overall structure of the task network is similar to behavior tree, and it also requires a blackboard to operate. The keys used in our blackboard are:

1) SelfActor
2) SelfLocation
3) MovementTargetLocation
4) CurrentEnemy
5) AmmoCount
6) HeldFirearm
7) Actor
8) IsBeingDamaged
9) IsShooting

The used task network provides the AI with several options in terms of creating and executing plans:

1) Find cover if being damaged or near a grenade.
2) Find and shoot targets.
3) Throw a grenade if the target is obscured.

Those plans consist of sub-plans and tasks. The AI creates plans, evaluates them, executes the most prioritized plan or the least costly one, and replans if some variables change. Decorators check if the conditions for plan execution are in place (HTNP game tree link [3]).

[2]https://shorturl.at/uX579
[3]https://shorturl.at/ipzGY

V. Data and Results

The study participants are recruited via a voluntary call at an IT university and through an internet channel called Yandex surveys. The survey has been completed by 51 people, with an additional 12 people answering only part of the questions. At this stage, we have not taken into account participants' gaming experiences. It is a significant factor and will be studied in future extensions of the work. Answers form questions two and five have been collected and represented as graphs that can be found later in this paper. Answers to the other questions were provided in a free form, so they were analysed by looking for commonly encountered keywords to determine which aspects of the game where most often noted by the players. The analysis of the survey results revealed the following common patterns:

1) Most players consider the AI to be an important part of the gaming experience.
2) Players often consider the byproducts of the main logic execution as the core features of AI.
3) While the AI nature of the opponents was clear to the players in both cases, the behavior tree version was rated to be slightly more human-like.
4) Despite the previous point, more players consider the HTNP AI to make more relevant decisions in regards to circumstances.
5) Most of the proposed improvements to AI were quantitative in nature.

A. Survey Statistics

The survey consisted of one general question and 5 questions for each build.

1) Does AI in games impact your immersion and overall enjoyment of the game? If yes - how strong of an impact does it have in comparison to core gameplay, graphics, sounds, etc.?
   61 percent of participants consider AI to be an important part of gaming experience, with an impact comparable to that of other aspects. Players also note that the AI impact changes depending on the genre of game, being most noticeable in shooters and action games.
2) How would you rate the AI on it's likeness to a human player? Please see graph in Figure 3.
   The scale goes from 1 - the AI clearly follows a set of instructions and is extremely predictable, to 5 - the AI can be mistaken for a human player.
   Both prototypes have a similar distribution of votes. The BT AI was voted to be more human-like. One player compared BT bots with low-ranked CS:GO players.
3) Can you name any noticeable patterns that the AI follows regularly?
   Most players noted oblivious patterns of behavior. Such as grenade throws, taking cover for HTNP bots and weapon pickups for BT bots. Exceptionally two players noted teleport usage by BT bots. In both cases players attributed AI with actions that were not programmed.

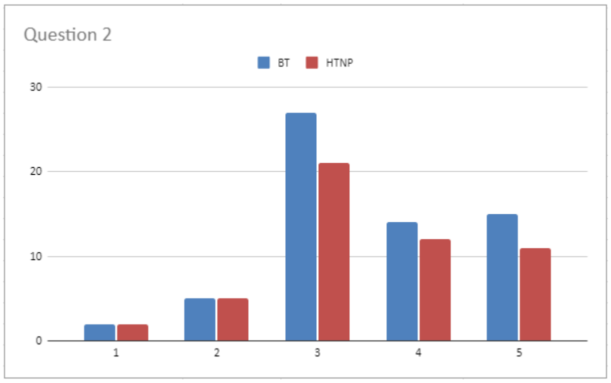

Fig. 3. How would you rate the AI on it's likeness to a human player?

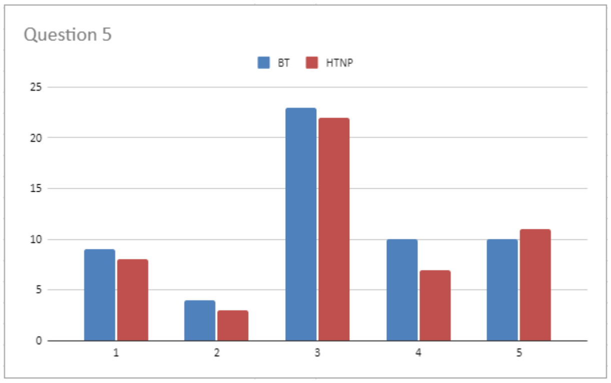

Fig. 4. Would the circumstances from the previous question affect you own decisions?

Several players said that the BT AI "baited" its teammates to get a kill and took cover when shot at, when no such logic was ever programmed. "Baiting" means letting an ally go first and take the most damage. Players attributed teamwork to HTNP AI, despite the fact that each bot always acts on its own.

4) Can you name some circumstances that influence the AI decisions?
Most players have correctly noted the opponent's location and line of sight to said opponent as the main deciding factors for both prototypes. Few of the players (approx. 10) said that the AI considered the map to make decisions, which is only partially true. The AI considers the map only for path-finding, the map does not affect its decisions in both prototypes. In addition, two players said that HNTP bots considered the location of their teammates.
5) Would the circumstances from the previous question affect you own decisions? (Graph in Figure 4)
The scale goes from 1 - I'd disregard those circumstances, to 5 - those circumstances will dictate my actions.
As in Question 2 the percentage distribution is similar in both cases. However, players consider HNTP bots to make decisions based on more relevant circumstances. Note that players assigned the score based on their own perception of the AI, not the real AI programming.
6) Can you propose any improvements that will make the AI behavior more human-like and believable?
In both cases players proposed quantitative improvements to the AI. The most common suggestion is to provide the AI with more options to choose from. None of the players made any suggestions regarding the decision-making process itself.

VI. Discussion and Conclusion

Currently, the AI in games has to keep up with the ever increasing scale of games and players' expectations of AI complexity and specialization. Fortunately for the AI and the game developers, the processing power of personal computers is also increasing rapidly, allowing for the creation of sophisticated and intelligent AI. Nevertheless, the large variety of games and genres requires a number of specialized AIs to perform different roles and actions, that each impact the overall experience of the person playing the game.

In this study, we have studied the impact of behavior trees and hierarchical task network planning AI implementation methods on the player experience in a limited environment of a third-person shooter game. The results of the study reveal that, despite being significantly harder to implement, on account of requiring a special framework to operate in comparison to being a default feature of the game engine, such as the case with behavior tree, the HTNP based AI did not have any noticeable impact on the players' gaming experience in comparison to behavior tree implementation. In addition, players attributed the AI of both implementations to properties not programmed into them. This leads us to the conclusion that players neither notice, nor appreciate the complexity of AI and its decision making process, as long as the AI acts within the expectation of players. Players focused on the end goal of the AI instead of the ways that it tried to achieve it. And as long as the AI did not break down or encounter any errors, the players were satisfied with its performance in the context of the game. Overall, the presented facts allow us to conclude that while AI is an important aspect of game development, it is mostly viewed superficially by players, and thus more complex implementations should only be employed in cases where no viable alternative is present or the showcase of AI complexity is the goal by itself.

In future works, we propose an expansion of this study to cover additional genres of games such as RTX, 4X, RPG, etc. to gather data for each individual case and compile a comprehensive study of AI impact on players experiences across all genres of games.

VII. Study Limitations

During this study, several limitations became apparent. Each prototype utilizes its own custom made environment, which might have impacted the perception of the AI by the players. The AIs represented in the prototypes are simplistic in comparison with AIs from modern games of the same genre. This may lead more experienced players among the participants in the experiment to have inflated expectations of the AIs' abilities. In addition, we would like to consider participants' game play experience as an impacting variable for this study.

REFERENCES

[1] Xueman Fan, J Wu, and L Tian. A review of artificial intelligence for games. In *Artificial Intelligence in China: Proceedings of the International Conference on Artificial Intelligence in China*, pages 298–303. Springer, 2020.

[2] By Epic Games. Lyra starter game. Accessed on 14.04.2023 [Online]. https://www.unrealengine.com/marketplace/en-US/product/lyra.

[3] By Mordor Intelligence. Gaming industry size & share analysis - growth trends & forecasts (2023 - 2028). Accessed on 29.10.2022 [Online]. https://www.mordorintelligence.com/industry-reports/global-gaming-market.

[4] Éric Jacopin. Game ai planning analytics: The case of three first-person shooters. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 10, pages 119–124, 2014.

[5] Egor klementev, Arina Fedorovskaya, Farhad Hakimov, Hamna Aslam, and Joseph Alexander Brown. Monte carlo tree search player for mai-star and balance evaluation. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 686–691. IEEE, 2020.

[6] Maks Maisak. Hierarchical task network planning ai. Accessed on 24.04.2023 [Online]. https://www.unrealengine.com/marketplace/en-US/product/hierarchical-task-network-planning.

[7] Stanescu Marius, Buro Michael, et al. Combining scripted behavior with game tree search for stronger, more robust game ai. In *Game AI Pro 3*. AK Peters/CRC Press, 2017.

[8] David M Mount. Cmsc 425 game programming1. 2015.

[9] Tamirlan Omarov, Hamna Aslam, Joseph Alexander Brown, and Elizabeth Reading. Monte carlo tree search for love letter. In *EUROSIS*, 2018.

[10] Jeff Orkin. Agent architecture considerations for real-time planning in games. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 1, pages 105–110, 2005.

[11] Jeff Orkin. Three states and a plan: the ai of fear. In *Game developers conference*, volume 2006, page 4. CMP Game Group SanJose, California, 2006.

[12] Ricardo Palma, Pedro Antonio González-Calero, Marco Antonio Gómez-Martín, and Pedro Pablo Gómez-Martín. Extending case-based planning with behavior trees. In *Twenty-Fourth International FLAIRS Conference*, 2011.

[13] AF Pukeng, RR Fauzi, R Andrea, E Yulsilviana, S Mallala, et al. An intelligent agent of finite state machine in educational game "flora the explorer". In *Journal of Physics: Conference Series*, volume 1341, page 042006. IOP Publishing, 2019.

[14] Simon Read. Gaming is booming and is expected to keep growing. Accessed on 29.10.2022 [Online]. https://www.weforum.org/agenda/2022/07/gaming-pandemic-lockdowns-pwc-growth/.

[15] Reed Simpson. Evolutionary artificial intelligence in video games. *University of Minnesota*, 2012.

[16] Penelope Sweetser and Janet Wiles. Current ai in games: A review. *Australian Journal of Intelligent Information Processing Systems*, 8(1):24–42, 2002.

[17] Peter R Wurman, Samuel Barrett, Kenta Kawamoto, James MacGlashan, Kaushik Subramanian, Thomas J Walsh, Roberto Capobianco, Alisa Devlic, Franziska Eckert, Florian Fuchs, et al. Outracing champion gran turismo drivers with deep reinforcement learning. *Nature*, 602(7896):223–228, 2022.