MSc in Artificial Intelligence and Machine Learning

CS4096 – Artificial Intelligence for Games

Game Development Project Report

A Quiet Place

Game artifact demo link: https://youtu.be/hSoovI5zVWA

Playable game artifact and source code link: https://ulcampus-my.sharepoint.com/:u:/g/personal/23052058_studentmail_ul_ie/EXYXm0t0quNDsEra8Cl95UMBS1vxEheXMPn3l687HsKghA?e=q0J6kd

Project Leader: Gavin Wade

Author: Siddharth Prince

Student ID: 23052058

Word count: 2,281

I. Introduction

A. Inspiration and Game Vision

The idea and motivation for our game project was to try and replicate the world from the thriller/horror movie, “A Quiet Place” [1]. The movie was quite a theatrical experience. There is hardly any sound for most of the scenes in the film, which in turn encouraged the audience to maintain the same level of silence to hold the tension. When there is even a small noise, it heightens the thrill, thereby delivering a unique and thrilling experience. We wanted to capture and deliver that sort of arrested attention and thrill to players of our game giving the jump-scares a little bit more of a kick (and inadvertently cause some rage quits).

B. Literature Review

In terms of games from which we have drawn inspiration from, “Alien: Isolation” [2] and “The Last of Us” [3] would be the closest to our game. Like in these games, there is an enemy game AI character whose motivation is to kill the player (in brutal ways).

Our game is set in a open world like in “The Last of Us” just without the top-notch level design. However, like in the “Quiet Place” movie, it is not sight that the aliens rely on to find and kill you. Rather, it is just sound since they do not have sight like the “Clickers” [4] from “The Last of Us”, but with literal otherworldly levels of hearing. Hence, the thrill is still delivered since you never can really tell when random interactions with the environment could make a slight sound. The moment you hear something on your earphones or speakers, you can bet the aliens are right around the corner barreling their way towards you. If you start running (making a lot of noise), it makes it even worse because you are now a beacon, projecting your location to the aliens who are much faster and stronger. Hence the open world gives the player an illusion of freedom. But the player is just as doomed to their fate as in the movie.

There is another recent game, “Lethal Company” [5] that has similar aspects to this concept and demonstrates a potentially more engaging approach to this genre of games. This is a multiplayer game and there is a remarkably interesting game mechanic where actual player-to-player voice communications can be “heard” by the monsters in the game compelling players to use comms carefully. This, however, is beyond the scope of our game project, but is useful to think about for a full-blown version of the game.

C. Basic Gameplay Premise

The general story for the game starts with our protagonist, the main playable character, finding themself in a world where aliens have invaded Earth. The game world is set up where the main character can move around the world by walking, sprinting and jumping. Ideally, there would be the crouch/stealth mode which would be the state (machine) where the player would make no noise whatsoever. But this would mean the character would move about the world very slowly which would not be good from a gameplay perspective. Hence for the demo, the character does not make any noise when in the state of walking. Sprinting does cause noise.

The AI gameplay aspect of the game is that aliens NPCs are spawned into the world and “patrol” areas of the map. If it hears a noise from the player because it is in audio range, it proceeds to navigate to the player’s last known location and investigates/attacks the player.

II. Analysis

A. Gameplay mechanics overview

The game demo has a playable character who can move in the game world using the usual directional keys of W/A/S/D. He can sprint or run by holding down the Shift key along with the W key (use the mouse to direct your running). He can also jump, but the jump animation has not been set up. The game world comprising of terrain map and world assets have been downloaded and imported from the “Flooded Grounds” free terrain pack from the Unity Asset Store [6] and the base game play up to building the character controls was done by following the YouTube tutorial playlist, “Unity 3D Zombie Apocalypse & Zombies Survival Horror Game” [7].

For the purposes of this project where the subject is about building and demonstrating game AI, the primary focus of this report is on the Alien game object which is this game demo’s “game AI”. There are two alien objects within this game world. The alien is just a generic unity 3D capsule object. The script, “AlienMovement.cs” contains the movement and behavioral logic for the Alien game AI which is also attached to the alien object as a component in Unity. The gameplay mechanics and game AI aspects for the game artifact entails three main parts. They are the alien AI’s navigation, its sound perception and the alien AI’s switching of state machines to act on the info. We shall discuss in detail each of these game AI aspects in sections II-B (State machines), II-C (Navigation) and II-D (Perception system) below.

B. State machines

The game artifact has 2 main game objects, the playable character and the alien non-playable character (game AI).

The main states the playable character can switch to are “Idle”, “Walk” and “Running”. For the idle and walk states, the character does just like their respective state names suggest. For the running state however, there is an additional component of sound that is played from an audio source component attached to the player object. The other states do not have any audio source attached to them. There are also two more states that were added as part of following the video tutorial series [7] of “RifleWalk” and “IdleAim”, but that is not relevant to the game AI discussion here.

For the alien, there are only two main states which are “patrol” and “chase”. When in patrol mode, the alien moves along a set path of way points using Unity’s Nav Mesh AI. When in chase state, the alien navigates at a significantly fast pace to the player’s last known location.

C. Navigation

For the chase mechanism, a number of YouTube tutorials were referred which were all mostly line of sight based [8]. Considering that we wanted to use a large, imported terrain, we used Unity’s inbuilt navigation AI, Nav Mesh Agent to lay the groundwork for the alien game AI to move around the world. The Nav Mesh Agent component is added to the “Alien” game object. In the script, there are methods for patrol and chase which are both called within the “update” method. However, the AI behaviour defined in each method will only trigger based on Boolean values. These Boolean variables are m_PlayerInRange (chase state if true) and m_InPatrolMode (patrol state if true).

By default, when the game is initialised, the alien object is in the patrol state. There are manual waypoints (empty game objects) that have been placed around the terrain. The transform attributes of these objects are added to a “waypoints” array associated with each alien. So, when in patrol mode, the alien will know to which positions on the map it has to keep going to.

Based on the trigger of sound made by the player within the listening radius of the alien, it will change its state to chase and have the player’s last position set as the current destination via the “SetDestination” method of Nav Mesh Agent. This triggers the alien to navigate to this new location. On arriving there, if the player keeps making noise, the player position value for the alien to navigate to keeps updating and hence the alien essentially chases the player. Also, the alien has a higher movement speed than the player so it can close the gap quickly.

If the player’s state is not “running” and he goes to being idle or just the walk state, there is no more audio played. Hence, the new player position is not updated, and the alien stops at the last known player position. Once, in this state, it waits for a bit after which the

AI sets the new nav mesh destination as the next point in the predefined waypoint array to which position the alien starts navigating to. Note that at this point, the “m_PlayerInRange” variable will be set to false i.e. the alien will no longer be in chase mode and the “m_InPatrolMode” variable is set to true making the AI follow the preset route of patrolling.

D. Perception system

The perception system is in the “DetectNoise” method in the AlienMovement script. A list of Collider objects is created by using Unity’s “Physics.OverlapSphere” method [9]. The method essentially creates an invisible sphere centered around an object (here, our alien) with a given radius value and returns a list of all objects that the sphere collides with defined by a layer mask. We create a “player” layer mask that we assign to the player object alone and associate this with the script component in the alien object’s inspector menu. We loop through all the objects in the collider array (in this case, it should just be one object which is the player object) and check if the player object’s AudioSource component is enabled or not.

The player object has another script associated with it called “PlayerAudio.cs”. This is a very simple script which checks if the current state is “running” by checking if the “sprint” key-bind and “W” key are pressed. If so, it sets the audio source’s “enabled” property to true and false otherwise.

Back to the DetectNoise function, if the player audio is enabled, the state variables are updated where the “m_PlayerInRange” variable will be set to true changing the alien’s state to “chase” and the “m_InPatrolMode” variable is set to false. The previously declared variable of “m_PlayerPosition” is updated to the player’s current position determined by the collider. This same variable is used in the chase method described in the previous section to send the transform object to Nav Mesh.

III. Reflection

A. Results

While the game demo developed is by no means polished or good enough to be playable by an end-user, we were successfully able to implement our versions of game AI that could “intelligently” detect a player’s presence and chase them based on a trigger mechanism. In my implementation of the enemy AI, it triggers the chase state based on sound because I wanted to stick to the theme of “A Quiet Place”. Thus, we do not need ray tracers to check whether the player is in sight of the alien and makes it a bit simpler and unique at the same time. When play-testing the demo, there was a feeling of eerie satisfaction seeing the enemy close in on you from behind suddenly when you made a sound.

B. Challenges

The biggest challenge by far for us as a team was getting our work synchronised. Given that we had large game assets, it proved difficult to get our work onto a VCS and had to manually transfer our work. This is also the main reason our game AI implementations differed towards the end because we both started working on them individually.

A personal challenge I faced was with getting familiar with the game engine, Unity. It took considerable time getting used to the workflows and nuances of it like knowing when to drag objects where, associating object masks, etc.

Another challenge initially was to get into the mindset of thinking how to make aspects of the game intelligent and thinking about the logic of the AI mechanisms in terms of code. It seems so trivial and intuitive as a player and consumer of such games, but this perspective from the other side was a revelation. Also, art and animation are fun but very tedious.

C. Future ideas

During the course of this semester, I've thought about various ideas and gameplay mechanics that would make a game set in the "Quiet Place" universe fun and engaging. Some of them are as follows.

Weaponise the aliens! Like in "the Last of Us", we could have a survival loot system. Say we had a brick or soda can and chuck it towards human enemies like looters. The aliens would annihilate them for us!

Small nuances to the game by playing with sound would make a big difference. For example, if your current shoe gets wet, it could start squeaking making you a sitting duck and it would catch the player by surprise. Creaking floorboards in abandoned houses would create tense encounters and is another good example.

Like in the movie with it being okay to make noise near a louder ambient sound source like a waterfall, we can have spots like these across the map as safe havens for players so that they don't always have to be in a tense state.

As the game progresses and the player gets to collect loot from the world, they can start adapting spaces as quiet places. For example, spread sand along common pathways so that their footsteps do not make sound which is also another idea depicted in the movie.

IV. Conclusion

In conclusion, this project has helped us understand what “AI” means from a game development perspective instead of the popular interpretation of AI in the mainstream. Instead of trying to make AI artifacts that have a semblance of intelligence and autonomy, game AI is usually parts of a game that project intelligence with some often predetermined and programmed logic. If the player has the illusion that the game AI is competent and engaging within the scope of the game, the game AI is well built.

V. References

[1] – A Quiet Place (2018) – Movie reference: https://en.wikipedia.org/wiki/A_Quiet_Place

[2] – Alien: Isolation – Game reference: https://en.wikipedia.org/wiki/Alien:_Isolation

[3] – The Last of Us – Game reference: https://en.wikipedia.org/wiki/The_Last_of_Us

[4] – Clickers in The Last of Us – Enemy game AI reference: https://thelastofus.fandom.com/wiki/Clicker

[5] – Lethal Company – Game reference: https://en.wikipedia.org/wiki/Lethal_Company

[6] – Flooded Grounds – Free 3D environment package from Unity Asset Store: https://assetstore.unity.com/packages/3d/environments/flooded-grounds-48529

[7] – Building the base game -Video tutorial reference: https://www.youtube.com/watch?v=tSkKIqvDTEM&list=PLA-xaldQ72ryGL-DyIGasa0qa6mIMcic6

[8] – Chase game AI mechanism – Video tutorial reference: https://youtu.be/ieyHlYp5SLQ?si=34MlybzUJKMBCoTW

[9] – Unity Documentation – Physics.OverlapSphere: https://docs.unity3d.com/ScriptReference/Physics.OverlapSphere.html