The NPC AI of The Last of Us

1 Introduction:

The Last of Us is a post-apocalyptic action-adventure game developed by Naughty Dog, set in a world devastated by a fungal infection that has turned most of the human population into grotesque, aggressive creatures called the Infected. The game follows the story of Joel, a survivor, and Ellie, a young girl, as they navigate through the dangerous world filled with both Infected and hostile human survivors known as Hunters.

The AI systems in The Last of Us are designed to create believable, immersive experiences with the Buddies, Infected and Hunters. The game employs a modular AI architecture that allows for easy addition, removal, or modification of decision-making logic. This enables the creation of characters that interact with the world in varied ways while maintaining simplicity and maintainability in the code.

2 Constraints/Challenges in AI System：

The game's AI system is built in three main modules, the buddy system, the infected system and the hostile NPC system. These three AI systems, depending on the game setting, also have their own unique attributes.

The primary challenge faced by developers is to create dynamic gameplay that responds to the player's actions, instead of relying on predetermined scripted events.

3.1 Navigation System in NPC

Each AI system in The Last of Us is built upon several fundamental low-level systems, including a triangulation-based navigation method. Developers also utilize a second channel system featuring a two-dimensional grid centered on each character, which is rasterized with all static and dynamic blockers. This allows for short, high-quality paths, while the high-level system plans the overall route between distant points [1].

A novel system introduced in The Last of Us is the exposure map. During the early project stages, developers discovered that AI needed information on what the player could and couldn't see to make informed decisions about their path. The exposure map represents this information as a two-dimensional bitmap laid over the navigation grid, where 1 signifies visibility and 0 indicates occlusion [1].

Initially, visibility checks were implemented by casting rays from various points along the NPC's path towards the player and using this data to assess path quality. To swiftly calculate the exposure map, a simple height map is embedded within each navigation grid. This height map employs an 8-bit integer to represent the world height of each point on the grid [1].

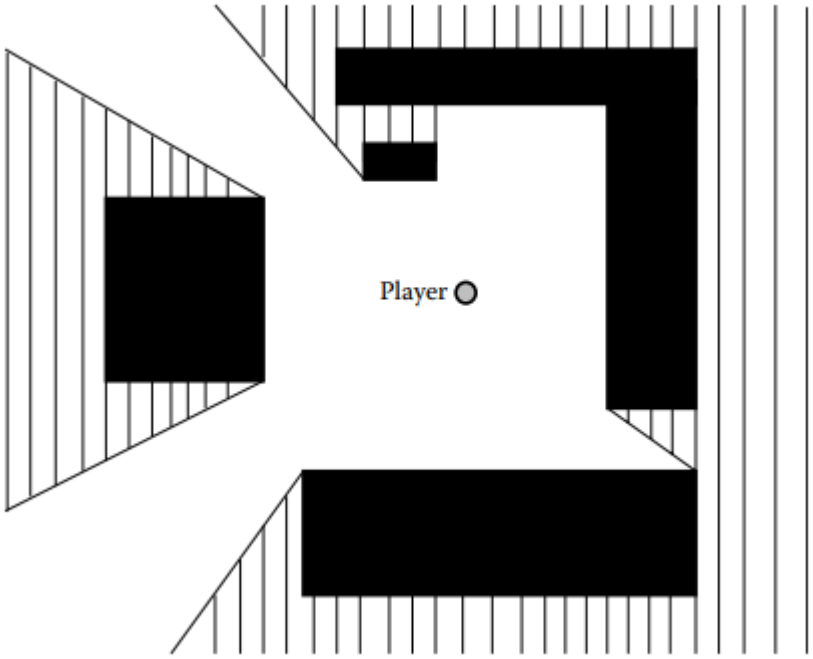

Fig.3.1
The exposure map covers everything an entity can see.

3.2 AI Perception

AI perception is critical to any game, particularly those focusing on stealth. In the Uncharted series, also developed by Naughty Dog, the AI utilizes specific systems to perceive the world. Their line of sight comprises a simple cone and rays to check for obstructions. This system was also initially used in The Last of Us, as illustrated in Figure 3.2.

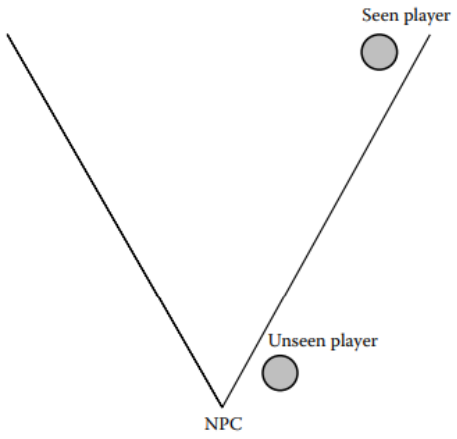

Fig 3.2 The simplest form of perception

However, during playtesting, issues occurred. Players close to NPCs were typically detected, while those farther away were not, due to the cones used for testing not accurately representing actual vision. The fundamental problem was the need for a more significant viewpoint when close and a minor viewpoint at a distance. A simple rule was implemented: the NPC's viewpoint is inversely proportional to distance. As displayed in Figure 3.3, the field of view was redesigned for improved efficiency. Interestingly, a similar rule is applied in the first-person shooter CS GO, where the side far from the cover can see the side close to the cover first.

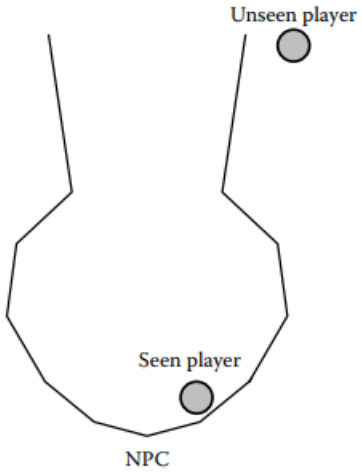

Fig.3.3 This more complex form of vision

When an NPC is not immediately aware of the player's presence, their perception of the player begins at a specific frame. Upon seeing the player, the NPC starts a timer that increments each time the character is visible. For each frame the player is not seen, the timer is decremented. The player is not considered perceived until the timer reaches a predetermined value (approximately 1 or 2 seconds for a typical NPC). In combat, this threshold is much lower; however, when the NPC is unaware of the player during stealth mode, the threshold is much higher [1].

In the original implementation, rays were projected to nearly every joint on Joel's body. Each joint had a weight, and the weighted average was compared to a threshold (usually 60%). If the total was higher, the player was considered visible. While this provided a reasonable approximation of Joel's visibility and eliminated edge cases where only his head or fingers were seen, the complexity of the projection made it challenging for players to predict if specific cover points were safe [2].

After experimentation, developers discovered that a single point on Joel's body could be used instead. The location of this point changed depending on whether the player was in stealth or combat mode, as illustrated in Figure 3.4. In stealth mode, the point was placed at the center of the player's chest. During combat with an NPC, the point shifted to the top of the player's head. This approach favored perception while in stealth, while still ensuring adequate visibility during combat situations [1].

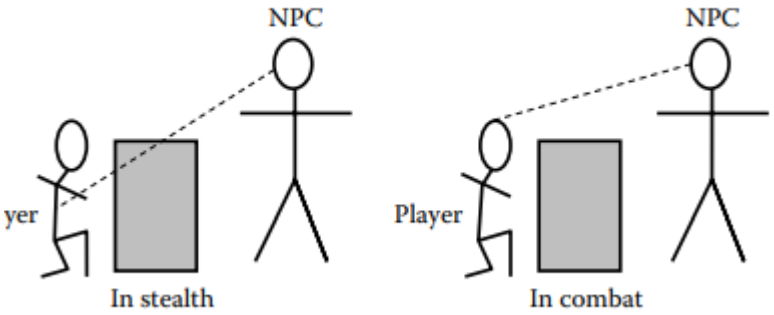

Fig.3.4

4.1 Following System in Buddy AI:

The following system is the core of the buddy AI. Developers hope the buddy AI can follow the player closely, through the implementation of this method can not only solve some of the problems when she encounters the enemy AI, but also enhance the player's playing experience and enhance the significance of ambiance [2].

Developers [2] designed a follow system to enable a buddy character to follow the player or another character smoothly and efficiently. The foundation of this system is a follow region, a torus surrounding the leader character, which is defined by a set of data-driven parameters. Various candidate follow positions are generated within this follow region and subsequently assessed for quality. This is accomplished by casting three sets of navmesh rays, as illustrated in Fig.1:

The initial set of rays extends from the leader's position towards the follow region, providing a clear route for the buddy to trail the player. For each ray that reaches the follow region, a potential follow position is generated (Figure 1a).

A subsequent set of rays is projected forward from each candidate position to ensure the position doesn't face a wall (Fig.1b). Though the developers initially contemplated allowing positions near a wall with the character facing away. However, it was found that having a buddy move directly adjacent to a wall was unnatural.

Finally, rays are cast from the player's location to each "forward" location to ascertain that progressing from this position wouldn't create an obstacle between the player and the buddy (Fig.1c). Although it might appear unnecessary route, this ray was critical during testing as it prevented the buddy from opting to stand on the other side of a doorway or fence, and instead, promoted their presence in the same "room" as the player.

The following positions generated for each frame are then rated an optimal position is selected for the partner. An optimal position is selected for the buddy to move to. These ratings are based on several criteria, including [3]:

•Distance to the leader
- Staying on the same side of the leader
- Visibility of potential targets (either hiding or exposing)
- Not being in front of the player
- Distance to other buddies

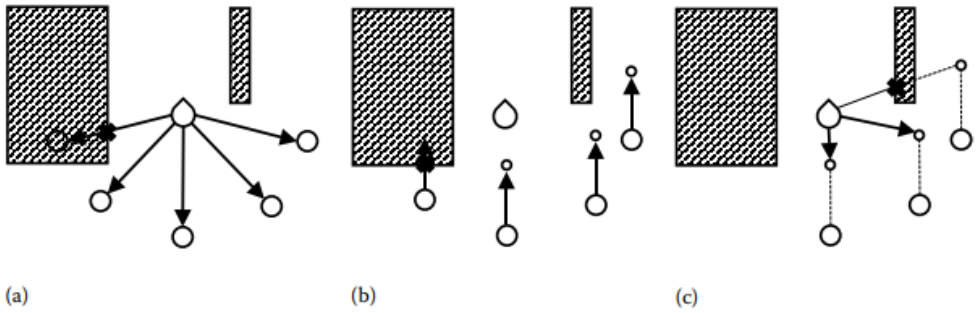

Fig.1 Ray cast design graph

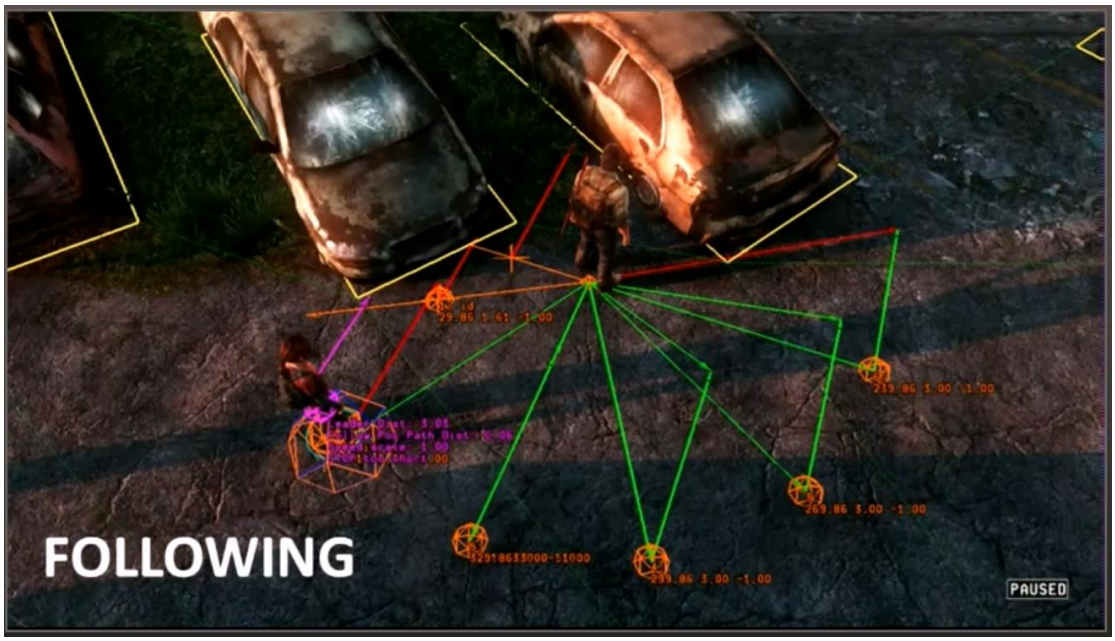

Fig.2 Actual space in the game

4.2 Moving and Dodging in Buddy AI

Once a position was selected for the buddy to move towards, it was crucial to ensure that her movement was believable. Developers observed her following the player around a space, and if a movement appeared unnecessary or unnatural, they paused the game to investigate the cause [4].

Several filters were added to the movement to make it seem more intelligent. Firstly, short-distance moves were limited, only allowing them when absolutely necessary, such as

when her current location was exposed to an enemy. Then, developers prevented her from running past the leader character if possible, unless not doing so meant stopping in an unfavorable location. Allowing her to get "close enough" to her follow position without crowding the player gave her following behavior an organic feel, unlike the rigid behavior she initially exhibited [4].

Each movement mode (walk, run, and sprint) had a single fixed speed, which varied among buddies and was different from the player's movement speed. This resulted in the buddy approaching an ideal distance from the player and then oscillating between running and walking, which appeared unnatural. To resolve this issue, developers allowed the buddy to scale her movement animation speeds by up to 25%, enabling her to maintain a specific movement mode for a longer time. As she neared the threshold to start walking, she would gradually reduce her running speed [4].

Developers assume player will go around a buddy character instead of rushing to them. Make her play a canned animation dodging out of the way at the last second. This approach to dodging makes her has more personality.

4.3 Cover generation in Buddy AI

The Action Pack is an integral part of the AI system and consists of three main packs; Cinematic AP, Traversal AP and Cover AP. Since The Last of us is a third-person perspective based action game, it is a very significant AI logic that partners work with the player to find cover. The developers created Cover AP to enable AI characters to lean against walls or similar obstacles for places of cover.

Developers implemented a hybrid method for generating runtime cover, merging the systems utilized by AI enemy with the dynamic systems used by players. The system used by NPCs, as well as the dynamic system used by players, already include the "edge of cover" feature. These features represent the intersection between wall and floor collision surfaces, accompanied by a marker that indicates height and corner [3].

The specific implementation process is similar to the following system shown in Fig.1, but still slightly different. Firstly, fire ray from the leader and generate the procedural cove features. Then gather the background cover edge features. Last, combining these two features and storing them in a specific data structure is shown in fig.3 [3]. The result of processing and evaluating these data in the game is shown in Figure. 4.

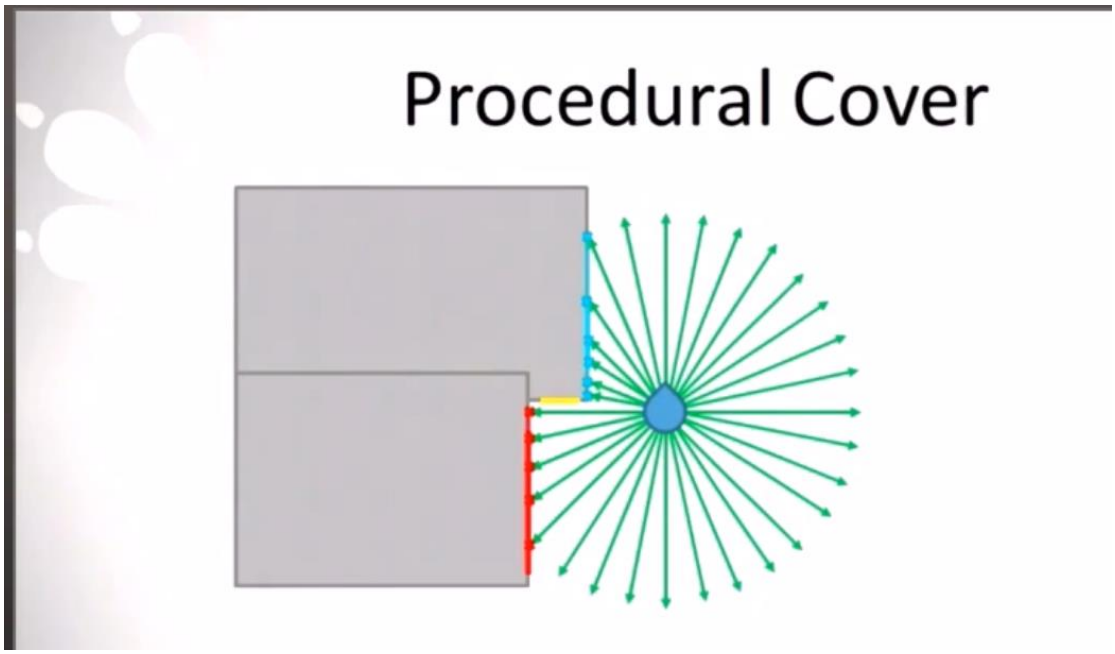

Fig.3 Procedural Cover

Fig.4 Actual procedure cove in game

5.1 Senses of the infected AI

The infected are humans who have succumbed to the parasitic wormwood fungus. The infection destroys the body, leaving no trace of personality, compassion or even self-preservation. The infected are driven only by the instincts of the fungus. There is a fundamental difference between the infected and the hunter. The hunter work in groups,

communicate with words and gestures, and protect each other. Players can see their cold-blooded brutality as a means of survival. By contrast, the infected appear confused and strange [4].

Figure 5.1 outlines the four stages of infection represented by the various Infected types. Runners are the most prevalent, known for their speed and tendency to attack in disorganized groups. They possess vision and, like all Infected, have heightened hearing that makes them equally effective in dark rooms and well-lit environments. Stalkers resemble Runners but prefer hiding in darkness to ambush their victims. Clickers are significantly disfigured by the infection, with fungal growths rendering them blind; they have developed echolocation as compensation. Although slower than Runners, Clickers possess a lethal frenzy attack and can resist unarmed melee attacks. Bloaters are severely disfigured, blind, slow, and heavily armored, making melee attacks futile as they can grab and kill anyone within their reach [4].

The sensory system employed in The Last of Us plays a pivotal role in determining the behavior and reactions of characters in the game. Infected characters primarily rely on their hearing, and logical sound events are designed to control the range and type of sounds detected by different characters. The hearing sensitivity of characters is also adjustable based on their current behavior, allowing for more nuanced gameplay. The sensory system is designed to provide a correlation between logical sound events and actual audio, enabling players to anticipate the reactions of Infected characters. However, there are certain exceptions, such as the logical breathing sound, which propagate over a short range but have no corresponding audio. The system also differentiates between Infected and Hunter characters by adjusting the broadcast radius of logical movement sounds in relation to the player's speed, thereby making Infected characters more challenging to approach stealthily.

Type	Runner	Stalker	Clicker	Bloater
Speed	Fast	Fast	Medium	Slow
Vision	Limited	Limited	Blind	Blind
Rarity	Common	Uncommon	Uncommon	Rare
Combat	Attack in groups	Ambush in dark areas	Melee frenzy, limited melee vulnerability	Armored, ranged attack, invulnerable to melee

Fig.5.1 Characteristics of Infected Character Types

6. Data-Driven Design and Implementation

According to Mark [], in The Last of Us, character code is kept general by defining sets of characteristics for each character type rather than explicitly referring to them in the code. All Infected character types share a single C++ class and are distinguished by skills and tuning variables. This centralized approach gives designers control over character variations, improves modularity and flexibility, and simplifies the process of adding new character types or modifying existing ones. Staying true to design principles and avoiding checks for specific character types is crucial for maintaining code stability and ease of implementing new features.

This approach also made it easier to configure difficulty settings for each character type individually, by modifying the thresholds at which they respond to stimuli.

7.Conclustion

In this analysis of The Last of Us, I have studied various AI systems, such as character perception, navigation, stealth mechanics, and the implementation of character behaviors. These systems have been developed with a focus on keeping the code general, modular, and flexible, which has proven successful in achieving the designers' goals.

Compared to other games in the exact genre, such as Uncharted, the game's AI system is more in line with the developer's initial intentions and has some updates to the original AI. For example, the exposure map was updated from the authentic navigation system in 3.1, and the AI perceptron mentioned in 3.2 was iteratively updated to better supply the game's logic. The Last of Us stands out with its advanced AI systems and the level of detail given to character behaviors and interactions. The seamless integration of AI into the game world and the adaptability of the characters to different scenarios set a high benchmark for future games in the industry.

While the AI systems in The Last of Us are impressive, there are areas that could be improved upon. For instance, the AI could be made more dynamic and adaptive to player strategies, resulting in even more engaging and unpredictable gameplay.

In conclusion, The Last of Us showcases an outstanding implementation of AI systems that essentially satisfy the developers' purposes and outperform many other games. However, there is always space for improvement and innovation, providing a continuous challenge for game developers to create more engaging and immersive experiences.

7 References

[1] McIntosh, T. (2014). The Last of Us: Human Enemy AI. GDC.
[2] McIntosh, T. (2015). Human Enemy AI in The Last of Us. In Game AI Pro 2.
[3] Dyckhoff, M. (2014). Ellie: Buddy AI in The Last of Us. GDC.
[4] Dyckhoff, M. (2015). Ellie: Buddy AI in The Last of Us. In Game AI Pro 2.
[5] Botta, M. (2015). Infected AI in The Last of Us. In Game AI Pro 2.