## Guard Detection

(Burns, 2015)

Creating a simple but effective way of having an AI to detect the player through site.

For my Stealth game I want the player to have to manage both their visibility and sound for when they are hiding from the AI. For visibility I have come up with the following ideas and methods of creating it within engine.

## Vision cones

(Cook, 2012)

UE4 already has vision cones built into it. When creating the AI Pawn you can add ‘AI Perception’ which allows you to edit and define what senses and how well they can work.

From this I can give all the AI a set vision distance that they can see the player from.

Once this is done my AI can theoretically be able to see the player and react to them entering their eye site from here they can then sound and alarm or attack the player depending on what the AI is programmed to do after they see them. This is great, however this means that compared to other stealth games the AI doesn’t have a fuzzy period where they’re not sure if they saw the player.

## Detection Range and Charge

(Varanini, 2009)

So, to give the player a chance on avoiding the AI as well as allowing them to bait them out, the addition of a detection meter and a last know position check should be added.

* Creating a more realistic method of Detection for the AI

(Nicol, 2020)
A very simple method of creating more realistic detection for the AI is by breaking down the vision cone of the AI into 2 or more sections. Each section can cause the guard to see the player faster since realistically you’d be a lot better at seeing someone 2 meters in front of you compared to 10 meters.

So, within the yellow zone a detection meter can go up by 5% a second, while orange is 15% and Red could be instant because your so close to the guard.

* How do you do this in the engine?

(Author, n.d.)

When you add the component ‘AI Perception’ to the AI Pawn in UE4 you can add a perception update event.

From this we can take both the AI’s location ‘Receiver Location’ and the targets location ‘Stimulus Location’ and find out what the vector distance between each of them is and call is PlayerLocation. This vector distance will act as our guide to where within the cone the player is being seen and thus how fast the AI should respond to the player. This is where the different sections of the cone come in. We can create 3 different Vector Variables which will then be used against the PlayerLocation to determine the speed of which the detection meter goes up.

* Getting Spotted

If the player is detected through the detection meter filling up, we can then have the AI run after and attack the player character. If the player manages to get spotted and then runs behind some cover to hide, we can use a ‘AI Perception’ variable to help the AI catch the player if they haven’t hidden themselves well enough.

This variable is‘Auto Success Range from Last Seen Location’ this variable will allow the AI to see the target within a specific range from a location as long as they have already been seen.

This means that we can give the AI the ability to instantly see the player if they stay within the area, they were detected in. This shouldn’t mean that the AI can see the player through walls just that they don’t have to think about if they saw the player, they now bypass the detection meter.

* Hunting the player

If the player isn’t spotted outright but the AI thinks they’ve seen something, we can have them go check out the location they thought they saw them. This can be done through the detection meter we set up in the engine. Once the player leaves the sight cone, we can have the AI check if they saw the player long enough to go to said location and do a quick look around.

We can do this by having a branch event that checks if the player isn’t detected but they did get seen 50% more in the detection meter we can have the AI move to the general area of the ‘Stimulus Location’ and then have them do a check event which could be a simple left and right turn check where they look to the right for 4-5 seconds and then look to their right for 4-5 seconds before then returning to their original position or patrol.

* Player hidden

When it comes to how the AI will respond to the player in shadows, we can simple place a check within the perception update event which will ask if the player character is currently registering if it is in shadow. If it is it’ll then choose to either not see the player at all or to only start detecting the player if they’re very close the AI.

# References

Author, U. (n.d.). *AI Perception*. Retrieved September 26, 2020, from Unreal Engine: https://docs.unrealengine.com/en-US/Engine/ArtificialIntelligence/AIPerception/index.html#:~:text=The%20AI%20Perception%20Component%20is,a%20sense%20has%20been%20detected.

Burns, S. (2015, March 31). *‘Metal Gear Solid 2’ Was the Game That Ended My Years As a Fanboy*. Retrieved September 26, 2020, from Vice: https://www.vice.com/en\_us/article/jmamxy/metal-gear-solid-2-sons-of-liberty-was-the-game-that-stopped-me-being-a-fanboy-220

Cook, D. (2012, October 11). *Dishonored vs EA’s Gibeau: a win for new IP*. Retrieved September 26, 2020, from VG247: https://www.vg247.com/2012/10/11/dishonored-vs-gibeau-a-win-for-new-ip/

Nicol, H. (2020). *Detection Cone.* Retrieved September 26, 2020

Varanini, G. (2009, December 18). *Tom Clancy's Splinter Cell: Conviction Multiplayer Hands-On*. Retrieved September 26, 2020, from Gamespot: https://www.gamespot.com/articles/tom-clancys-splinter-cell-conviction-multiplayer-hands-on/1100-6243784/