Grand Canyon University

Project 3 - Make a Move 1: At Your Own Risk

Justin Dietrich and Ryan Scott

CST-415: AI in Games and Simulations Lecture & Lab

Dr. Ricardo Citro

February 6, 2022

A brief description of the game or simulation (one paragraph)

1. Our game will be from a top-down view, set in an environment that has paths, walls, and other types of cover or obstacles. Project 3 features (alongside the ‘minimax algorithm’) numerous gameplay updates. The player (now equipped with a gun he can aim, shoot, and reload with) will move through randomly generated grid-like levels, trying to find or complete the objective(s) while fighting off a variety of different enemies that are placed throughout the level.

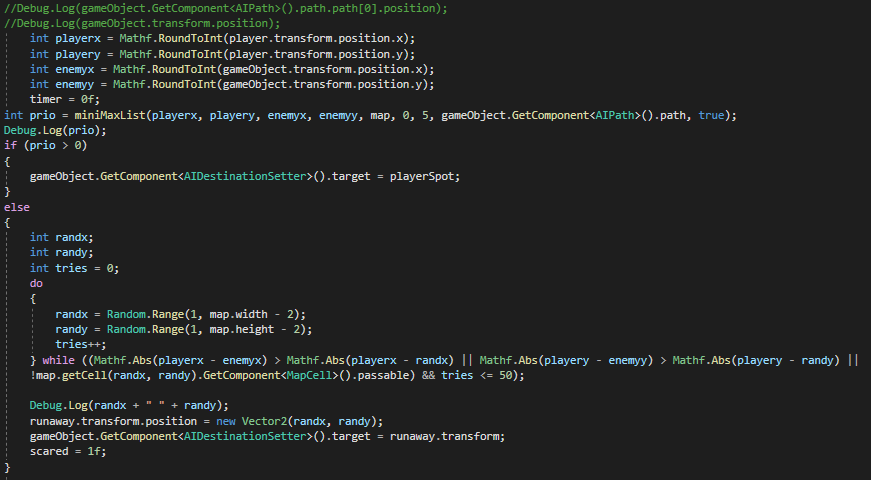
How are the concepts listed above relevant and its purpose? (one paragraph)

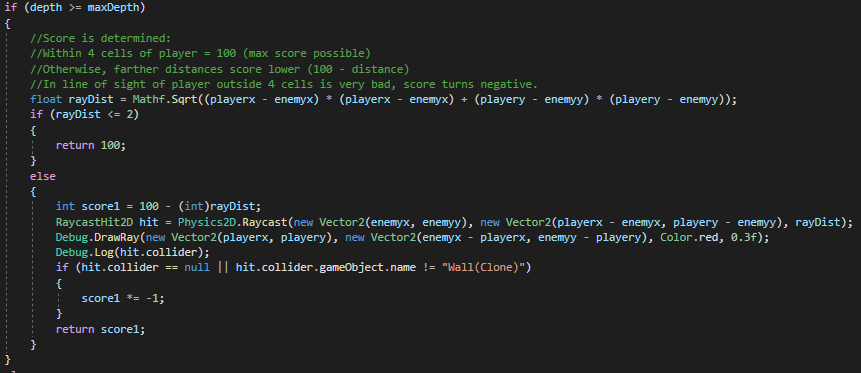
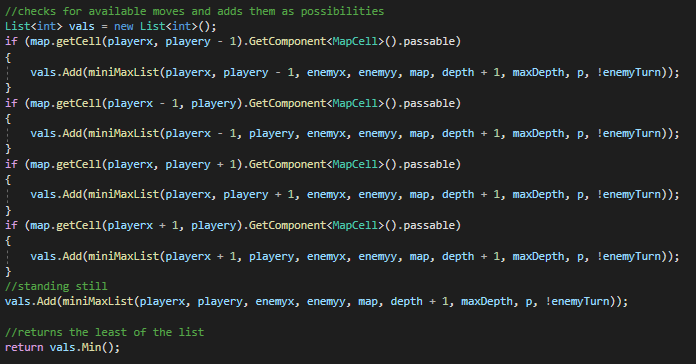
1. The game uses the concept of ‘minimax algorithm’ in the runner enemy. The runner has a melee attack, so is easily beaten using ranged weapons at a distance. To counter this, it using a minimax algorithm to see if it should pursue the player or run into cover and hide.

Which search method will be used? (one paragraph and bullet points outline)

1. The algorithm makes a tree of all possible moves for a set amount of turn cycles. The player can move in four directions or stay. Depending on how many walls are around the player, there can be 1 - 5 moves for the player per state. Currently, the enemy can only pursue, so their nodes only have 1 child. A scoring system is implemented based on the distance between the player and enemy and whether there is a line of sight between the two.

Github: <https://github.com/AsePlayer/CST-415>

Example Scripts

The enemy uses the minimax algorithm to decide if it should run away. If the result > 0, it pursues the player. Otherwise, it finds a spot on the map to run away to that is not in the direction of the player.  
If the depth of the algorithm has reached its max, it calculates a score and returns it. This is the bottom of the tree, where there are no more children.  
The player, if playing optimally, will choose the minimum value returned by the functions here. A list of possible scores is made, and the smallest value is returned.

How will you overcome unforeseen obstacles during implementation? What is your 'plan B'?

1. If the minimax algorithm ends up ultimately not making much sense on an enemy, we could always repurpose it for some broader or niche purpose. This could include a grouping system that determines if the enemies should attack or wait for more reinforcements, etc.

How is the project aligned with the current topic objectives?

1. The project shows that we can create a program that can see all possible moves in the future and choose the best one, assuming both players play optimally.

| Minimizer | The player AI is the minimizer, and chooses the score worst for the enemy. |
| --- | --- |
| Maximizer | The enemy is the maximizer, and chooses the score that works best for it. |
| States | Scores for each state are calculated based on distance between the two and whether there is a line of sight. |

What will appear on the screen: animation, user interactions, information dashboards, UI elements, etc.

1. Project 3 features a visual update to the map tiles as well as a mechanical update for the player. The player now shoots projectiles which instantiate bullets and activates the muzzle flash for visual feedback. The user can click to shoot and press R to reload. In this game, your ammo will carry over between excessive reloads because it’s more fun that way.

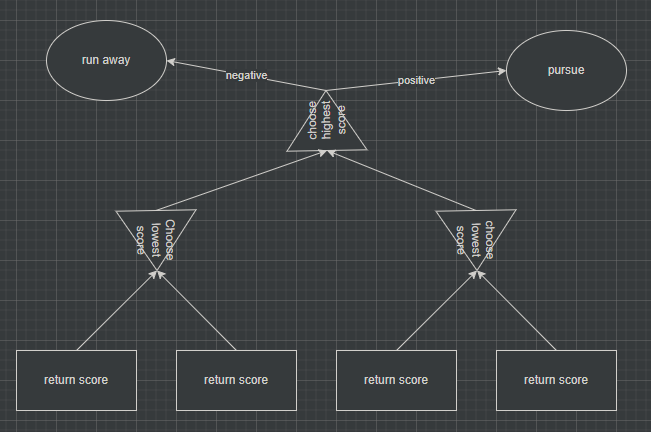
We have opted for a compositional approach, and that is reflected in the new scripts that have been implemented (Weapon.cs, Ammo.cs, and Bullet.cs, Health.cs). This system should allow us to swap in and out scripts with ease and hopefully, we can get more bang for our buck with features in the limited time that we have. An example of this is being able to slap on the Health Component onto any game object, including walls or other projectiles like a rocket. Just by virtue of possessing the Health component, they are susceptible to bullet damage and can die. Most importantly, this interaction is self-contained in its own bubble, allowing us to make changes with ease.

UI will eventually accompany the systems soon for maximum visual feedback. This will include ammo, weapon, health, score, and potentially more.

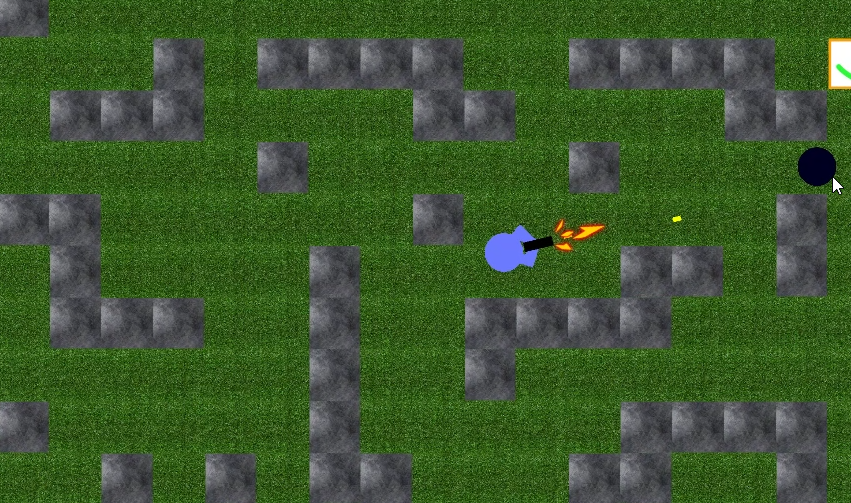
List the platform and software tools you plan on using

1. Stuff we will use:  
   Unity (with C# scripts)  
   Adobe Photoshop  
   Adobe Illustrator  
   MS Paint  
   Audacity

Flowchart for minimax



Screenshots below:

Visually Updated Gameplay Screenshot:

Reloading Sequence:

