Grand Canyon University

Project 4 – Make a Move 2: Bear the Consequences

Justin Dietrich and Ryan Scott

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

Dr. Ricardo Citro

February 20, 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 4 features (alongside the Bellman equation) attacking enemies and the first pass on basic User Interface for the player. The player (now equipped with either a gun or knife) will move through randomly generated grid-like levels, while fighting off a variety of different enemies that are placed throughout the level with weapons of their own.

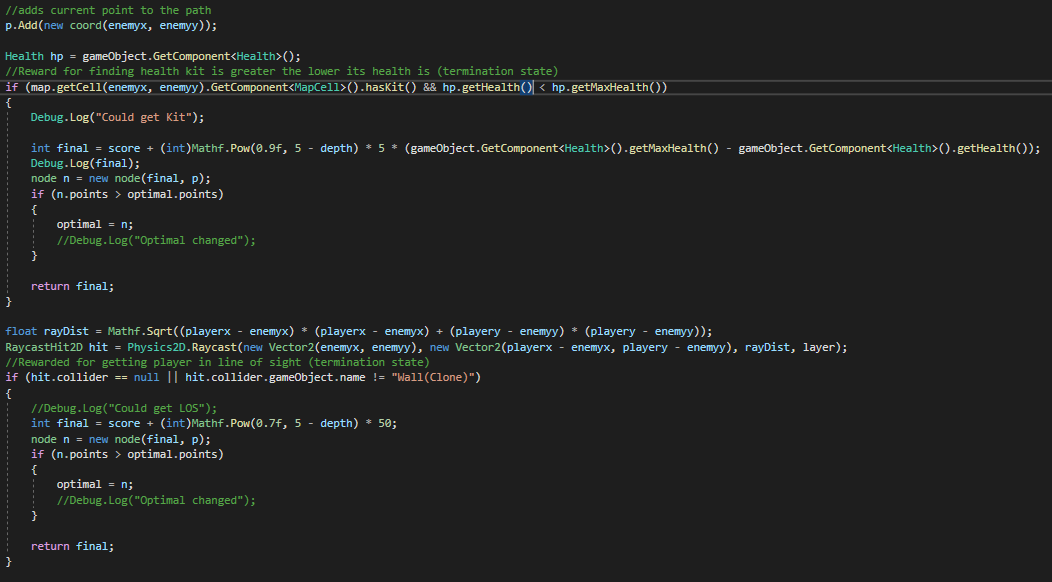
How are the concepts listed above relevant and its purpose? (one paragraph)

1. We use the concept of the Bellman equation with the Brute enemy. The Brute enemy makes a tree of moves recursively, keeping track of the path. When a terminal state is reached, the path and its score are kept track of and compared with other paths. Rewards for state changes are going from searching state to attacking state, and going from low hp to more hp. Each move is costly and has negative reward, encouraging getting rewards sooner than later.

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

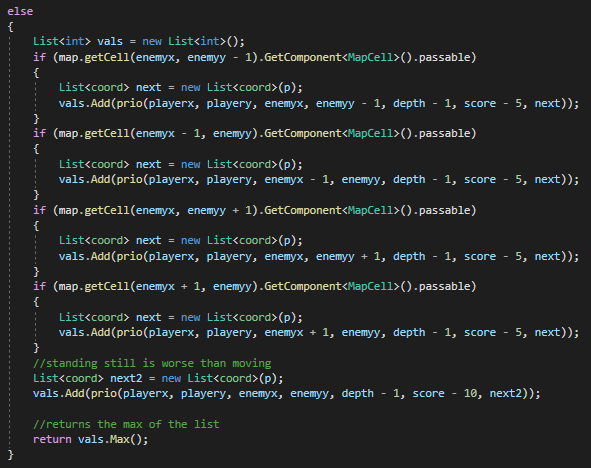
1. The Brute has a function that takes an empty list, creates a clone of the list for each branch, and adds the new tile to the list for each iteration. If the max depth is reached or a terminal state is reached, the cumulative score of the path is calculated, and the path is compared with the current best path to see if it should be replaced. Each move subtracts from the score, as well as the rewards for getting health kits and getting line of sight decay as the depth increases. If the max depth is reached, the brute loses more score the farther it is from the player.

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

Example Scripts

The terminal states. The return statement just ends the branch. The function will compare the path and replace if needed in the small if statement.

All of the next branches if terminal state is not reached. Score loses 5 points per move.

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

1. If what we want didn’t work, we would have just focused on making the equation rather than its functionality in the game. We could make it so that it finds an optimal path, but just make it so that the enemy doesn’t move along with it smoothly. Also, if we can’t get the enemy to function like an enemy would, we could just have an object in place of the enemy follow the path.

How is the project aligned with the current topic objectives?

1. The project shows that we can create a program that can make decisions and find an optimal path based on obtaining rewards and when they are obtained..

| Bellman Equation | A recursive function that calculates point values for state changes is implemented. |
| --- | --- |
| Rewards | Enemy is punished for moving and not changing states. Rewards decay in value over time. |
| States | Scores for state transitions are calculated when going from passive to aggressive and changing hp. Both are terminal states for the path. |

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

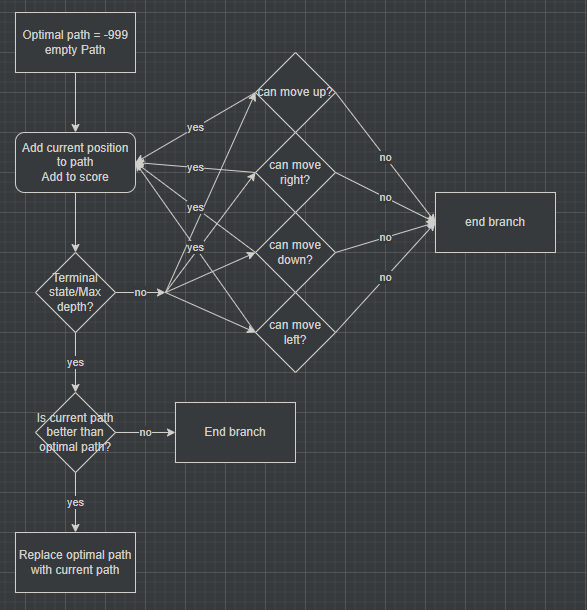
1. Project 4 features a brand new Brute enemy as well as a cool new reload animation for the pistol weapon. The enemy can now shoot projectiles with the pistol and slice with the knife, utilizing more of our state machine systems. The player can also technically “lose” now. Enemies spawn continuously and try to kill the player, while the player tries to survive and accrue as many points as possible.

The first pass of the User Interface now accompanies the systems for visual feedback. This includes ammo, weapon, health, and score. There has also been a new reload animation added for the pistol to indicate that the player cannot shoot during this time.

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 Bellman equation:



Screenshots below:

Showcasing new reload animation and enemy attacks!  
