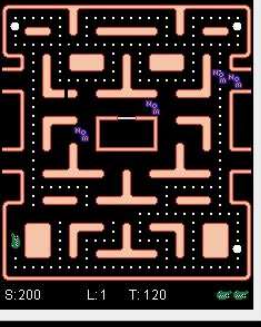


PROJECT POST-MORTEM


Project Title: GatorRaider Project

Date Prepared: 11/2/19


Project Overview:



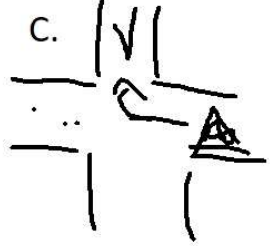
A.



B.



C.



$\Delta = A + K$
 $\bigcirc = Def$
 $V = \frac{Vul.}{Def}$

A. Attacker main focused is to avoid the nearest defender by fleeing in the opposite direction. This is done by comparing the distance between the attacker and the defenders. If the closest defender is within a set unit of the attacker, then the attacker flees.

B. The attacker second focused is prioritizing power pill over regular pills.

C. Once picking up the power pill the attacker focused on eating defenders.

Since the attacker pick up pills while performing task A, B, C. The last priority was for the attacker to pick up the left over pills, while maintaining task A of avoiding defenders.

Key Accomplishments:

The attacker was able to perform the basic functions listed in the overview. The attacker was able to change direction when it encounters a defender within 11 units of itself. In addition, the attacker successfully prioritizes on pathing towards power pills while continuing to avoid the defenders. Once obtaining the power pill, the attacker was able to path towards the closest defenders. Overall, the basic functions that was set out at the start of the project was accomplished.

Key Problem Areas:

There are many issues with the design, especially when it comes to situational occurrence within the game. The attacker sometimes walks into a wall and stay there for a while and I couldn't figure out why. In addition, the attacker fails to correctly responds to situation where it is pincer from multiple directions at a junction. Instead of taking the only path out, the attacker continuously moves back and forth. This is probably due to the flaw design of making the attacker flee from the closest defender. Therefore, it can be assume that when the attacker encounters with 2 or more defenders that are within 11 units of the attacker, the attacker can't properly flee from the defenders even if there is a possible path to escape because the attacker

PROJECT POST-MORTEM

prioritized fleeing from alternating closest defender rather than prioritizing a possible escape path. Overall, the program is too reliant on the attacker to make its own pathing decision. The lack of specific pathing instruction is most definitely the cause of many of the issues, such as running into walls or the attacker making questionable pathing decisions that result in deaths.

Project Reflection

Going into this project it was important to have a detail plan before starting to implement the program. I began this project with an overview plan of how I want the attacker to behave. However, I should have taken into consideration of implementing a specific pathing behavior rather than being reliant on the “flee” mechanic from `.getNextDir()`. If I could have successfully implemented this then I might not have to worry about the attacker putting itself into obvious inescapable scenarios. At the start of the project, the first line of code I wrote was able to make the attacker path towards the closest pill it can find. This single line of code alone was able to get ~4000 points. Unfortunately, the more I tried to complicate things in order to make the attacker smarter, the more dumber the attacker becomes. Therefore, I decided to go for a simpler design and implement the most basics of functions. However, there are a lot of room for improvement. In addition, there were many functions that I didn’t use that could have been helpful for the attacker to make smarter pathing decision. For example, I could have use `getPossibleLocations` for defenders and maybe implement a way for the attacker to avoid those locations when the defenders are within a certain range of the attacker. Therefore, if I am to take on another project of similar nature to this one in the future, then I’ll make sure to implement more specific behaviors into my A.I. by utilizing a greater number of methods.