Teddy, Jason, Matthew, John
Professor Jenkins
CSC 470
23 October 2022

Requirements

Business

- Make an entertaining product
- Visualize how an artificial intelligence learns through generations based on feedback loops

User

- User should be able to observe how ai ‘think’ in a basic environment or be able to be entertained by watching ai solve a maze
- The user will be able to generate mazes, run an AI through that maze through multiple generations to see how the AI learns to solve it

Functional

- User should be able to click a button to generate a random maze
- User should be able to click a button to have the AI runthrough the maze
- User should be able to click a button to run the next generation of the AI through the maze
- Users should have an option to have a different random maze if they don’t like the one that was generated the first time and so on.
- User should have the option to adjust just how many agents are running at one time
- Making the program in a format that is easy to install onto new computers with at least windows operating systems

Non-funtional

- The program should generate a 20x20 randomized maze within a few seconds
- The program should be able to support running up to 100 individual AI agents at the same time

Implementation

- Training the users on how to use the program

Use Case Name: Entertainment	ID: 01	Importance: High
Primary Actor: User	Use Case Type: Business	
Stakeholders: Team Monkfish		
Brief Description: A user wanting to run our maze software for entertainment purposes will be able to do so simply and efficiently		
Trigger: Users utilizing the software for entertainment	Type: **External** / Temporal	
Relationships: The user runs the AI in the name of entertainment.		
Normal Flow of Events: A user will start by launching the software and generating a maze. The agents will generate to the display and begin navigating the maze as intended. The user will have options to change parameters as they see fit, including agent count, maze size, or other variables.		
Subflows: The user changes parameters to get different results.		
Alternate Flows: If the software were to run into runtime errors or limits due to application complexity, an error will be displayed and the application will quit.		

Use Case Name: Testing the Algorithm	ID: 02	Importance: High
Primary Actor: User	Use Case Type: Business	
Stakeholders: Team Monkfish		
Brief Description: The user will be able to test the general AI algorithm and how it changes in runtime.		
Trigger: Users looking to observe AI algorithm	Type: **External** / Temporal	
Relationships: The user would want to observe the AI. This use case relates to the education use case as a student can learn more about AI.		
Normal Flow of Events: A user will run the application as normal, noting how the artificial intelligence navigates the generated maze. The user may run the application again to note differences in algorithm runtimes, in which case the maze and agents will be reset.		
Subflows: None		
Alternate Flows: If the software were to run into runtime errors or limits due to application complexity, an error will be displayed and the application will quit.		

Use Case Name: Variables effect on Performance	ID: 03	Importance: Medium
Primary Actor: Programmers	Use Case Type: System	
Stakeholders: Team Monkfish		
Brief Description: We will test how different variables change the performance of the AI in terms of how quickly it learns to solve the maze.		
Trigger: Programmers wanting to know the effect of variables on performance Type: External / **Temporal**		
Relationships: Programmers will get to see the results of performance based on different variable values. This use case has a relationship with the education use case as a student will be able to learn more about how an AI works.		
Normal Flow of Events: The user will be able to change variables prior to running the program. From there, they will run the program to completion and notice how each of the variables affects the performance of the AI.		
Subflows: None		
Alternate Flows: If the software were to run into runtime errors or limits due to application complexity, an error will be displayed and the application will quit.		

Use Case Name: Education	ID: 04	Importance: Medium
Primary Actor: Programmers	Use Case Type: Business	
Stakeholders: Team Monkfish		
Brief Description: The program will have the opportunity to help students learn exactly how an AI works.		
Trigger: Programmers want to help teach students	Type: **External** / Temporal	
Relationships: Team Monkfish wanted users to learn more about AI. This use case is related to use case 2, 3, and 5 as they are all somewhat related.		
Normal Flow of Events: A student will use the program to learn more about AI and how different variables affect how quickly it can solve the maze.		
Subflows: None		
Alternate Flows: An alternate flow to the normal flow of events would be that the student does not learn from the AI that we programmed.		

Use Case Name: Hardware Limitation	ID: 05	Importance: Low
Primary Actor: Programmers	Use Case Type: System	
Stakeholders: Team Monkfish		
Brief Description: To find hardware limitations while running the application, the parameters will be turned to their maximum and performance will be monitored.		
Trigger: Programmers wanting to test hardware capabilities	Type: External / **Temporal**	
Relationships: The programmers wanted to test hardware capabilities and just how much an AI can stress a system. This use case is related to the education use case as we would learn more about how the program stresses out hardware.		
Normal Flow of Events: The application will launch as normal, generating a maze and agents to a screen. The available parameters will be increased to an exceptionally large value, and runtime will be monitored. As the software or hardware are expected to overperform, limits will be measured and recorded.		
Subflows: None		
Alternate Flows: If the software were to run into runtime errors or limits due to application complexity, an error will be displayed and the application will quit. If the software does not produce an error, the parameters will be tuned until limitations are found.		