Group Members: Max Rider, Alp Yuksektepe, Ethan Karner

Title: Dungeon Crawler

Problem Statement:

- Enemy Pathfinding to player while player is moving

- AI Difficulty Scaling in Relation to Player Victories
    - https://spronck.net/pubs/SpronckGAMEON2004.pdf

- Using AI to avoid Deadlock scenario in PGC game (Bonus)
    - https://aquila.usm.edu/cgi/viewcontent.cgi?referer=https://www.google.com/&httpsredir=1&article=1307&context=honors_theses
    - https://www.sbgames.org/sbgames2019/files/papers/ComputacaoFull/198359.pdf

Problem analysis:

- State Space Formulation: We will have a tile based state space that generates a finite MxN rectangle. When the game starts or when the player progresses a level, dungeon space is procedurally generated on the map. The player has a FOV that is a light that surrounds a couple blocks. The player is expected to find the end goal which is to get to the next level by exploring the map and revealing what type of dungeon space has been generated.
    - The state space is reasonable for a 2D game design as it's similar to pacman and Legend of Zelda. The 2D environment is easier for AI to navigate.
- State Transition Function:
    - Enemy Pathfinding: The enemy AI will change states (x,y position) based on if the tiles next to it are "legal" moves (i.e. not walking through walls) that move it closer to the players current position.
    - Difficulty Scaling: Based on how well the player is performing on each level (i.e. how many enemies they have killed) The enemies on subsequent levels become more difficult to kill (More health point, armor, stronger attacks, etc)
- Evaluation Function
    - The goal state for the enemy AI is to be on the closest tile to the player for it to attack and kill.
- Partially Observable: Both player and enemies have a limited FOV
- Multi Agent: Multiple enemies plus player
- Deterministic: The only uncertainty comes from the player movement
- Episodic: Enemy actions won't necessarily affect future decisions
- Dynamic: The player can move while the AI is deciding the best path to get to them
- Discrete: There are a finite number of states in the environment and a finite number of things to do in the environment

Data set/source materials:

- https://aquila.usm.edu/cgi/viewcontent.cgi?referer=https://www.google.com/&httpsredir=1&article=1307&context=honors_theses
- https://www.sbgames.org/sbgames2019/files/papers/ComputacaoFull/198359.pdf
- https://spronck.net/pubs/SpronckGAMEON2004.pdf

Deliverable/Demonstration:

We're making a playable game that can be opened via an executable file

Eval of Results:

We will know that we are successful if the game is actually playable and the AI respods as expected (i.e. seeks player, grows stronger, etc)

Major Components + schedule:

The game is going to be built using PyGame
Our schedule for the major components:

1) Fully implement sprites and graphics (for the floors, walls, monsters, players, etc)
2) Player and enemy movement (general movement+pathfinding)
3) Make a procedurally generated map
4) Game mechanics
5) Add in game items

References:

- https://aquila.usm.edu/cgi/viewcontent.cgi?referer=https://www.google.com/&httpsredir=1&article=1307&context=honors_theses
- https://www.sbgames.org/sbgames2019/files/papers/ComputacaoFull/198359.pdf
- https://spronck.net/pubs/SpronckGAMEON2004.pdf