CSE 701 Project Proposal

NPC Race

A computational comparison of non-playable character pathfinding algorithms.

The goal of the project is to test various pathfinding algorithms to not only see how fast they are but on how hard they to implement.

Project constraints

* Project must be implemented completely in C++ and GCC
* project has to contain only your code (no libraries)
* project must be somewhat related to your research or physics

Terms:

AI – Artificial Intelligence, with respect to video games the branch of computation related to mimicking certain human behaviour in a game environment.

NPC – nonplayable character, with respect to video games a character that is not directly controlled by the player.

Project idea:

The basic idea is to compare pathfinding algorithms.

Background - Context

Non-playable characters (NPCs) or artificial intelligence (AI) in video games make up a significant portion of the game experience of many games. To simulate intelligent navigation AI generally will move towards a specific target, say for example towards the character. To do this they implement a form of pathfinding to calculate where to navigate. Video games are an interesting application because there are timing requirements that state when the program should produce a result. For example, if a game is frame controlled, running at 144 Hz (144 frames per second) you only have 6.94 milliseconds to complete each frame. For computers that do operations on the nanosecond timescale this may seem like an eternity but when you also consider you may need to do all the computational expensive game logic, physics, and graphics there is very little time leftover for the AI. This means that we need to be able to be as computational efficient as possible when it comes AI pathfinding.

Background – Theory

Algorithms for pathfinding have been well established since the dawn of computing. Dijkstra’s algorithm [Algorithms] is a simple algorithm for tree traversal. As discussed before video games has its own set of standard algorithms that have been used for many years [AI book]. Some common algorithms include A\* path finding, [continue list]. The computational runtime of these is well known as well as numerous issues. Here the goal is to learn C++ so we’ll be computational verifying these runtimes.

[needs citations from algorithms book and AI book]

Procedure

To test compare the computational efficiency we’ll be running a series of trial runs with varying parameters. An AI agent will be placed in a fixed location and a target location will be randomly generated. Then a test will be run using A\*, \_, \_, and \_ algorithm.

Parameters:

- Size of the space

- Speed of the agent.

- Target location.

• Static target vs dynamic (randomly?) Moving target.   
• Run many thousands of times to get averages.

• Stretch: have ASCII display AI vs target and compare.  
• Stretch: could write in Python to compare  
• Good project because I could know how to do this it's just testing my C++ skills  
• Need to chart out details and timeline to be realistic and specific as possible.  
• Run in discrete space or float space?  
• Also could include going up levels

OR

Could learn and do graphics algorithms

Pick a couple algorithms and try to do ones that are faster