Complex game design

What the system is.

This system is for AI agent pathfinding with node graph creation from the gpu. The system can create a node graph for pathfinding from user defined objects (E.g., The user can tell the system that any object tagged with “walkable” is to be processed to have pathfinding nodes). It is then able to process AI agent paths from the node graph using the gpu to perform parallel processing to iterate through the node list with the a\* algorithm to find the shortest path from two given points. After this process, the AI agent will be returned with the path to follow and it will move along it till the end destination has been reached.

What is the purpose.

This system’s purpose is to reduce the cpu overhead that is associated with AI pathfinding, as the gpu can have hundreds of parallel threads processing the pathfinding compared to the cpu where an entire thread can be locked up till the processing is done, and if enumerators are used on the cpu to offset a loop to process the whole path, it can take a large amount of time for the path to return. This system aims to have the gpu perform a majority of AI processing to allow for faster pathfinding and to allow a higher amount of AI agents without a drop in performance compared to using a cpu.

What unity libraries will I be using.

This system will be using the following unity engine libraries, the general library, the editor library and the events library. The general library is needed for basic unity functions such as monobehaviour, the editor library is needed as the node graph system will be its own editor object and the events library will be needed as this system will be run on events rather than in an update function.

What mathematical operations would my system use to function as intended.

Basic vector math such as distances between two vector threes will be used for the most part as the system will be based on the distance between position vectors. Float differences will be used for getting height differences between positions.

What advanced algorithms are required to be implemented.

A star pathfinding algorithm will be used as the main pathfinding algorithm. This is because it is faster than using breadth first search and similar simple pathfinding algorithms. This algorithm will have changes made to it to allow for parallel processing of nodes without overlapping.

How will the system be modular.

As this system will be similar in functionality to unity’s inbuilt navmesh system the modularity can be used for many different game genres and ideas without having to change the system itself, there will be many options to change how the system works such as the number of nodes per distance, the height restrictions between nodes and general AI variables such as speed, turning speed and etc to allow the system to be used in more ways such as a car not just a non-playable character.

How will it be integrated into an application.

The system will be a unity package that is downloadable from the asset store, when added there will be a custom inspector added for the node graph generation, in this inspector the user can change what objects are walkable, the object tags that are walkable or not walkable, the number of nodes, the height restrictions between nodes and etc. from there the user can press a bake node graph button to create the node graph for the pathfinding algorithm.

To create the AI agent the user can add a pathfinding component to a game object, this component is controlled through script. The agent will have functions to control its movement, to follow a path and to stop its movement. This is to allow the user to add their own custom logic to the agent without hindrance.