Modern Computer Games
COMP 521, Fall 2020

Assignment 3

Due date: Friday, November 20, 2020, by 6:00pm

Note: Late assignments will only be accepted with prior **written** permission of the instructor. You must **explain** all answers and **show all work** to get full marks! Please make sure your code is in a professional style: **well-commented,** properly structured, and appropriate symbol names. Marks will be very generously deducted if not!

Description

This assignment should be done in Unity as a 3D game level.

1. You will first need a basic game level. Build a 3D level with an outer perimeter as shown in the overhead-view sketch below. The level is roughly rectangular with a relatively flat walking surface. Varying size rectangular alcoves are along each wall (3 alcoves on the top, 3 on the bottom, 2 on the left, and 2 on the right). Choose different alcove sizes unique to your implementation, but note that this is static design. Also take note that you will need to know the abstract geometry (coordinates of each corner and their order). Use an overhead camera and relatively even lighting to allow easy visual observation of all parts of the level. **5**

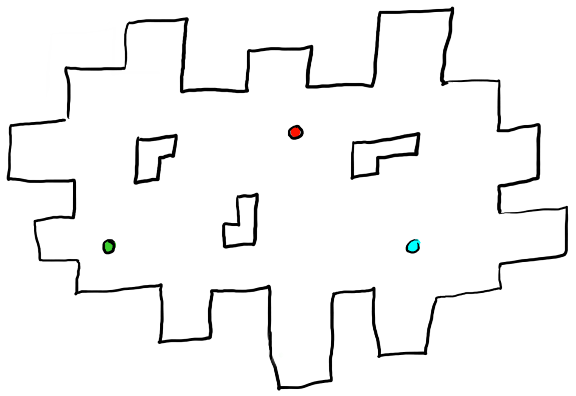

Figure 1: Sketch of level design. Lines are intended to be straight and axis-aligned.

2. Each time the simulation is started, add 3–4 L-shaped obstacles, roughly following the proportions shown in the sketch. Obstacles should have randomized shapes (but still L-shaped), and random locations within the level, but should not overlap with each other or the boundary. The sketch above shows them as axis-aligned, but that is not a requirement (ie your choice). **5**

3. Within this map you will need to path-find for multiple characters. Build a *reduced visibility graph* for the level interior. Each vertex and edge of the graph (other than start/goal nodes) should be represented visually. **10**

4. You will need to add agents to the simulation. An agent should have occupy a non-0 area (shown as coloured dots in the sketch), and must never overlap with other agents, obstacles, or the level boundary. Include an editor control to select the number of agents, and spawn them at random, non-overlapping points in the level. Make agents different colours (as much as possible) so individual agents can be distinguished. **5**

5. Agents should repeatedly move to random destination points within the level. All such destinations should be within the walkable level area—how to guarantee that is up to you. Once they reach their destination they should wait 200-1000ms, and then choose a new destination. **10**

Implement an A* search on your reduced visibility graph to do the pathfinding. You do not need to necessarily guarantee a minimum path length result from each search, but your paths should be relatively sensible. Path planning should be optimistic, in the sense that you plan around obstacles, but ignore other agents.

An agent's goal must be visually represented (eg as a node coloured the same as the agent but shaped differently). It should be visually clear which segment or node of the reduced visibility graph the agent is on at any time when it is pathing.

A single agent should always be able to reach its destination if its destination is reachable.

6. Agent paths will sometimes interfere with each other. Identify situations in which an agent cannot proceed, and use a replanning strategy to help agents eventually get to their destination. An agent identified to have been blocked should discard its current path plan, wait 100-500ms, and then compute and start following a new plan to get to its destination. After 3 failed attempts to reach a given destination an agent gives up and chooses a new random destination. **5**

7. Experiment with an increasing number of agents. Start with 2 agents and double the number until performance or behaviour becomes unacceptable, in each case running the simulation for at least 30s, multiple times (at least 3), and recording the number of paths planned, number of replannings, number of plans successfully reached, and total planning time. **10**

   Provide *as a separate (.pdf) document* a graph or table showing average metric results in relation to the number of active agents. Discuss your results—how many agents can your design effectively support?

What to hand in

Assignments must be submitted on the due date **before 6pm**. Submit your assignment to *MyCourses*. Note that clock accuracy varies, and late assignments will not be accepted without a medical note: **do not wait until the last minute**.

Include all source code necessary to run your simulation.

This assignment is worth 15% of your final grade. **50**