Modern Computer Games  
COMP 521, Fall 2019  
Assignment 3  

Due date: Monday, November 18, 2019, 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

In this assignment you will develop a Unity simulation of an AI agent, mainly focused on using a search-based planner. You must implement the AI decision making process yourself; do not use any built-in or external asset tools or systems for AI development or implementation. This does not extend to pathfinding, for which you should be using Unity’s pathfinding to navigate characters from one place to another.

The basic design is inspired by the board game “Century: The spice road”. Your AI-controlled character is a spice trader, and the task is to create a planner that allows your character to trade spices. The main goal is to fill a caravan with 2 units of each of 7 spices, Turmeric, Saffron, Cardamom, Cinnamon, Cloves, Pepper and Sumac. In order to do this your character needs to visit traders who each trade one specific spice for another, in various quantities, and deal with a roving thief.

1. Build a game level following the overhead view of the design shown below. This consists of a large area, with a caravan in the center (surrounded by 3 small walls), and 8 traders in surrounding alcoves.

![Diagram of the game level](image)

Note that this is a 2D representation. You can construct this as either a 2D or a 3D environment, positioning the camera for a full overhead view, but you must use Unity for this assignment.

Include (simple!) representations of the traders, caravan, and two dynamic agents, the player-agent and thief. Traders and the caravan are static scene elements, and thus do not need to move. Trader locations are randomized on each play-through. To help see who is who, visually tag your traders with numbers 1 through 8.

The player-agent and thief should be contained to the level area, and not able to pass through walls, each other, or other obstacles (including traders and the caravan). To move the player-agent around the space you must use Unity’s NavMesh pathfinding. Scale player-agent movement speed so that at default simulation speed it takes around 2s to cross the area diameter (i.e., movement is quick, but still easily discernible).
To help with grading (and debugging), add controls for the simulation:

<table>
<thead>
<tr>
<th>Control</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spacebar</td>
<td>pause/unpause the simulation</td>
</tr>
<tr>
<td>+</td>
<td>increase simulation speed by a factor of 2</td>
</tr>
<tr>
<td>-</td>
<td>decrease simulation speed by a factor of 2</td>
</tr>
</tbody>
</table>

2. The ultimate goal of the player-agent is to store 2 of each spice in the caravan. Spices are acquired by trading different spices between the traders. Each trader accepts a type of spice and returns another type. To carry out a transaction, the player-agent must navigate to a trader and wait 0.5s (at normal speed) for the transaction to complete.

(a) Trader 1: Gives you 2 turmeric units.
(b) Trader 2: Takes 2 Turmeric units and gives you 1 Saffron unit.
(c) Trader 3: Takes 2 Saffron units and gives you 1 Cardamom unit.
(d) Trader 4: Takes 4 Turmeric units and gives you 1 Cinnamon.
(e) Trader 5: Takes 1 Cardamom and 1 Turmeric and gives you 1 Cloves unit.
(f) Trader 6: Takes 2 Turmeric, 1 Saffron and 1 Cinnamon and gives 1 Pepper unit.
(g) Trader 7: Takes 4 Cardamom units and gives you 1 Sumac unit.
(h) Trader 8: Takes 1 Saffron, 1 Cinnamon and 1 Cloves unit and gives you 1 Sumac unit.

The player-agent begins next to the caravan, and has an inventory that allows carrying at most 4 units of spice at any one time. The caravan has unlimited capacity, and traders have unlimited supply.

In addition to trading and pathfinding actions, and while at the caravan, the player-agent can either transfer any number of items from their inventory to the caravan, or vice versa.

The player-agent’s behaviour must be completely defined by an AI planner. Implement a GOAP-based approach, defining a world state, one or more goals, actions, and a suitable approach to plan search. Your player-agent should demonstrate reasonable behaviour and it must be evident that it will achieve the main goal, but it does not need to compute an optimal solution.

Note that your planner can assume knowledge of the current full game state space, including specific trader abilities and locations, and the thief’s state. Plans, however, must be fully computed at runtime.

As a separate document provide a description of your world state, your goal(s), and the actions you defined.

3. To track progress, your interface should also show the number of each spice within the inventory and caravan in a tabular form, dynamically updating it as the simulation progresses:

<table>
<thead>
<tr>
<th></th>
<th>Tu</th>
<th>Sa</th>
<th>Ca</th>
<th>Ci</th>
<th>Cl</th>
<th>Pe</th>
<th>Su</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inventory:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Caravan:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

There must also be a dynamic rendering of the player-agent’s current plan. Show the planned sequence of actions on screen somewhere, using meaningful textual labels for each action you define.

4. Additional planning complexity occurs from a thief. Every 5s (at normal speed), with a 33% chance, the thief chooses to either intercept the player or go to the caravan. In the latter case, once they reach the caravan they remove a random item (if any are there). Once they reach the player, they remove a random item from the inventory (if any are there).

The thief begins at a random location, moves at the same speed as the player, and when not on a thieving mission wanders randomly. Once committed to an action the thief does not change their mind, and once accomplishing an action they return to wandering. The thief only steals at most 2 items, after which they will only wander randomly.

You may use any AI system you want to implement the thief.
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**.

For the Unity questions, hand in an exported project containing all files needed in order to reconstruct and run your simulations.

For non-Unity questions, submit either an ASCII text document or a .pdf file *with all fonts embedded*. Do not submit .doc or .docx files. Images (plots or scans) are acceptable in all common graphic file formats. You may submit a single document for such questions, as long as each answer is clearly delineated.

This assignment is worth 15% of your final grade.