Spike: Task 18

Title: Navigation with Graphs

Author: Sam Huffer, 101633177

# Goals / Deliverables

* Create a navigation graph simulation demonstrating:
  + A game world divided into tiles, and that supports a navigation graph structure.
  + Agent path planning based on current environment, using cost-based heuristics.
  + Four or more agents of at least two types that can follow their own independent paths.

# Technologies, Tools, and Resources Used

* SublimeText (for editing, executing and testing the code)
* Learning materials on Canvas (for instructions and sample code)

# Tasks Undertaken

* I started by copying the project from Task 17: Graphs, Paths and Search and the agent, weapon and projectile classes and the spike report from Task 16: Goal-Oriented Action Planning into the Task 18: Navigation with Graphs folder, stripping the spike report down to what was needed for this task.
* Box world rendering a stationary agent.
* Start no longer gets a path, agent calculates path to target. R key randomises agent position. On resize, agent stays in its current box.
* Agent moves to next node in path, checks which box it is in and updates its box field. Rate of movement is scaled along each axis by the ratio between the original width and height of the window and the current width and height of the window.
* Multiple agents moving to separate targets, R key now randomises target positions. Removed start. BUG: sometimes path nodes seem to get strapped to agents and the path goes all over the place.
* All agents wandering. AStar is the default search method. Fixed bug where couldn’t switch path lines off.

# Instructions for Operating the Code

* Left mouse click: place or clear a block, or move the start or target positions.
* 1: mouse clicks now clear blocks.
* 2: mouse clicks now place mud.
* 3: mouse clicks now place water.
* 4: mouse clicks now place walls.
* 5: mouse clicks now move the start position.
* 6: mouse clicks now move the target position.
* M: scroll forward through the list of search algorithms.
* N: scroll backwards through the list of search algorithms.
* Space: plan a path using the current search mode.
* B: alternate thickness of box lines.
* C: toggles markers of the centre of boxes.
* E: toggles displaying of movement network edges.
* L: toggles box labels.
* O: toggles highlighting of the optimal path in red.
* P: (un)pause the simulation
* R: randomise the position of the target(s).
* T: toggles display of graph options that were considered but did not end up being the optimal path.
* U: toggles circle markers of boxes considered during the search.

# Code Snippets

# In-Simulation Screenshots

# What I Found Out