ECS7016P Interactive Agents and Procedural Generation
Coursework: Unity Project

Deadline: 11am, Friday 12th April 2024
Weight: 40% of the module

For this assessment you should create a Unity project that implements an **agent simulation using behaviour trees** within a 2D level that is **procedurally generated using cellular automata**.

Core Requirements

The agents and generator should be of your own design, satisfying the following requirements.

1) The project should contain **a single scene** (Assets/Main.unity). When played in the Unity editor, it should generate a level and spawn the agents within it.

2) The level generator should be a constructive pipeline with **three stages**:
    a) Generate an initial 2D grid of filled and empty cells.
    b) Apply cellular automata to the grid.
    c) Post-process the grid to generate the final level.

3) There should be at least **two types** of agent (Agent A and Agent B), controlled (perhaps in-part) by behaviour trees implemented using the NPBehave library. There may be more than one instance of each agent type spawned in a level.

4) The level design and agent behaviour should realise your **allocated scenario** (see below).

Beyond these requirements, this assessment provides you some creative freedom. The design and implementation of the agents, and each stage of the generator, is up to you. You may include additional elements of your choice. In particular, you are free to use:

- Any algorithms in stages a and c, and any kind of CA rules in stage b.
- Any other algorithm for agent control, in combination with BTs.

Submission

You should submit a Unity project directory, **compressed as a zip file**, containing all your source code and assets. In addition, that directory should contain a **README.md** file which documents your project, including:

- A detailed description of the design of your generator and agents.
- A link to a video hosted online (e.g. YouTube) of a screen recording of your project running. At least one minute in length, showing at least three generated levels.

There are additional submission requirements given at the end of this document.

Scenarios

Each student will be allocated a scenario that serves as a “design brief” for their project. Your scenario should guide you in the design of the level environments and the two types of agent. Submissions which ignore the specified scenario will be heavily penalised.

You are free to innovate within this scenario, e.g. decide how particular agents should behave and interact with each other and objects. You will not be penalised for including additional design elements, e.g. different tile types, objects in the environment, a third kind of agent.

You will be allocated one of the following scenarios:

Scenario 1: Troll Lair
Environment: Caves
Agent A: Troll
Agent B: Thief

Scenario 2: Zombie Apocalypse
Environment: Ocean Islands
Agent A: Survivor
Agent B: Zombie

Scenario 3: Undersea Explorers
Environment: A Undersea Reef
Agent A: Diver
Agent B: Mermaid

Scenario 4: Haunted Forest
Environment: A Forest
Agent A: Adventurer
Agent B: Forest Spirit

Code Reuse

The design of the generator and the agent behaviours should be your own work and unique to your project. You may of course take inspiration from existing games.

The submitted code is expected to be your own work, apart from the NPBehave library.

You may choose to **reuse other code from the labs** under strict conditions:

- It is code provided for module labs, or a code library used in a lab, e.g. MovementAI.
- It is clearly separate from your own work, i.e. in separate source files or clearly indicated with code comments.
- Its reuse is documented in your project README.md.

Assessment Criteria

Marks will be awarded for the design, the implementation, and output quality of the generated levels and agent behaviour. Implementation includes code comments.

Level Generator (60%)
- The design of the three generator stages a, b, and c.
- Correct and readable implementation of appropriate PCG algorithm(s).
- The quality and diversity of the generated levels.

Agents (40%)
- The design of their behaviours.
- Correct and readable implementation of appropriate agent behaviour algorithms.
- The quality and diversity of the observed behaviour.

Assessment will not consider any graphical, audio or interactive elements of the project. (You can include them e.g. to make it a portfolio piece, but you won't gain or lose marks for those aspects.) It does not have to be a playable game.

A fail (less than 50%) will have not met many of the core requirements.
A satisfactory pass (50-59%) will have met most of the core requirements, with some errors.
A good pass (60-69%) is expected to have met all the core requirements, with distinctive elements in the design or implementation or output of either generator or agents.
An excellent pass (70-79%) will have gone beyond the basic requirements, with excellent design and implementation of either the generator or the agent behaviours.
An exceptional pass (80-89%) will be excellent across the entire project.
An outstanding pass (90+%) is for exceptional work which demonstrates particular originality, quality and/or depth. (From previous years, some students do achieve at this level.)

Submission Requirements – at risk of penalty!

You should submit a Unity project directory, **compressed as a zip file**, containing all your source code and assets. It should be compatible with Unity 2021 or 2022.

The compressed project **MUST** be less than 50MB in size. To keep your project small, do not import unnecessary files, e.g. large art assets. You may omit files and directories that the editor does not need to open a project, as described by the standard Unity gitignore.

The Assets directory should organised as follows:
- A **single scene** Main.unity which runs without errors (warnings are acceptable).
- A directory Generator containing only the C# scripts for your generator.
- A directory Agents containing only the C# scripts for your agents.
- Any other assets should be organised sensibly.

Remember to include a README.md with documentation (see "Submission" above), including a screen recording hosted somewhere online, e.g. GDrive, Youtube.