GD2P01 – Artificial Intelligence for Games

Mini-Project (30%)

Component code and name	GD2P01 Artificial Intelligence for Games
Assignment name	Assessment 4 – Mini Project
Weighting	30%
Submission deadline	Week 16
Week issued	Week 12

Brief

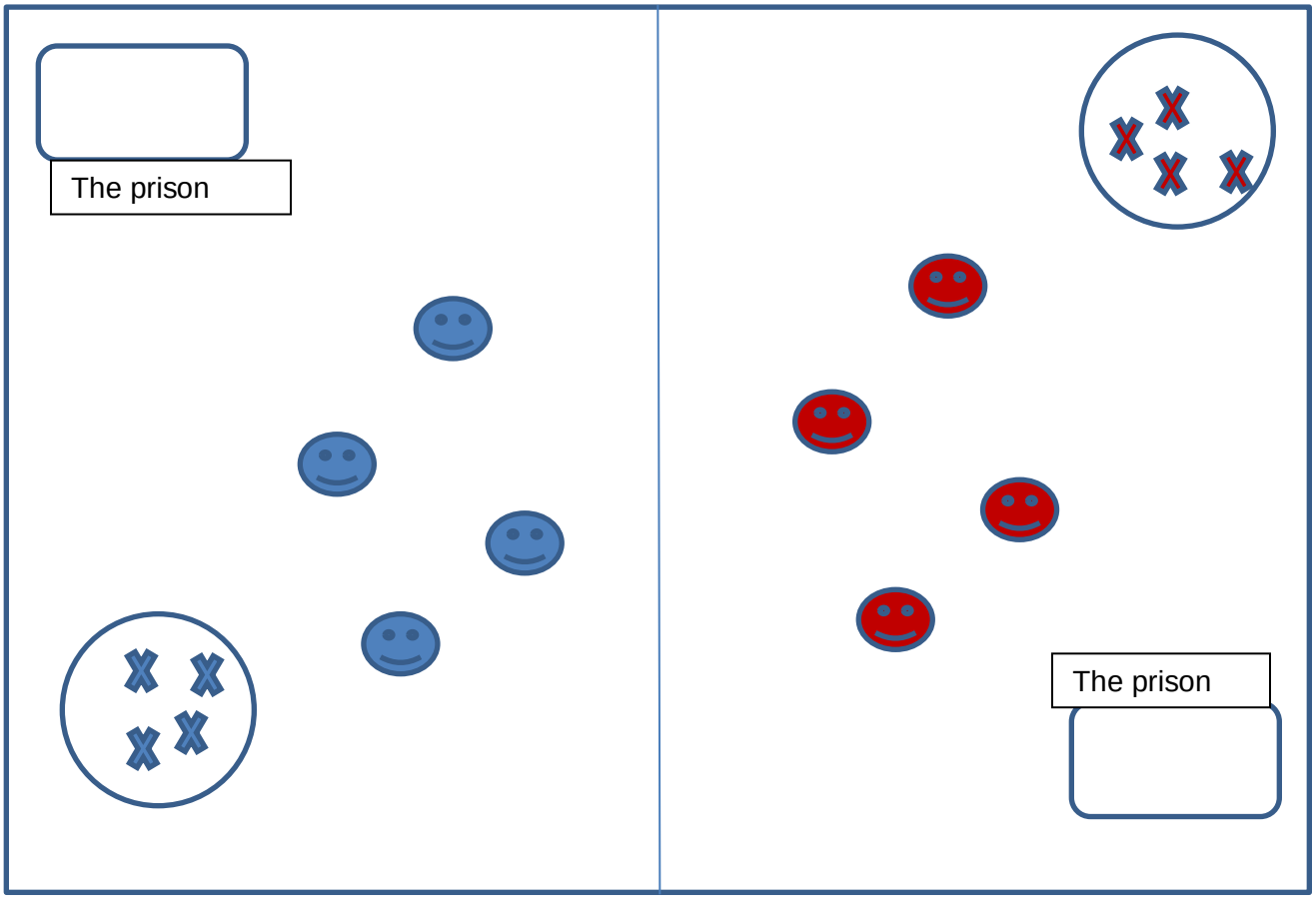
This assessment will demonstrate that you understand the theory and implementation of steering behaviours that can be used by an agent to navigate the environment. You will use the fundamental building blocks of software engineering that you have learned before, together with the steering behaviour algorithms to create an environment that utilises these different behaviours to move an agent/group of agents around.

Please refer to the Task Instructions (below) for further details on how to complete this task.

Instructions/Requirements

1. Using **Unreal Engine 5 or Unity**, you are required to design and implement a simulation of the game Capture the Flag.

Application Design Requirements:



1- The agents should be able to:

   a. Navigate smoothly through the scene.

   b. Be able to capture the flag when it reaches the flag area.

   c. Free another imprisoned agent when it reaches the prison area.

The game has the following rules:

1- The goal is to capture all the other team's flags and bring them to your team's side

of the field (minimum of four flags for each team).
2- An agent can only carry one flag at a time.
3- Both teams have a section of the map that is “their” territory
4- If an agent is tagged by the other’s team’s agent when in their territory:
    a. Any held flag is returned to its original location.
    b. The tagged agent goes to prison (the prison of the team they were tagged by).
5- If an agent is in prison, they can be freed by another agent from their team by tagging them.
6- Only one agent at a time can be freed from prison.
7- Each agent can choose between either capturing a new flag or freeing another agent in prison but cannot do both at the same time.
8- If an agent frees one of their team-members from prison, both need to return to their team’s side of the field before doing any other action.
9- An agent with a flag cannot free an agent from prison.
10- Only one agent from each team is allowed to be in the other team’s territory at a time.
11- Each team has an equal number of agents.

- You need to design the AI to control one team of agents.
- For the other team, one of the agents at a time will be controlled by the player, and you need to design the AI that controls the rest of the agents in the team.
- The player should be able to choose which agent they will be controlling.

For the team that is totally controlled by the AI:

1- The AI should be able to pick one agent to start moving towards the other side of the field (Decision Making).
2- The selected agent needs to choose between either: (Decision Making)
    a. Try to capture a new flag.
    b. Free an agent from prison.
3- The selected agent needs to find its way towards its target (flags/prison) while trying to avoid being tagged by the opponent team members (Path finding and/or steering).
4- The rest of the team should be responsible for protecting the flags by trying to tag the other team’s agents once they have entered their side of the field or after they have reached the flag. They should also protect the prison so that no one gets freed from it.

For the team that is controlled by both the player and the AI:

1- The player should be able to pick which agent it will control and can start moving it around the field.
2- The other agents in the team should have their AI programmed so that they can try and protect the flags from being captured or prevent the players in prison from being freed.

One team wins when they have managed to capture all the flags of the other team.

The game should include (but not limited to) the following functionality.

- Game starts with clear instructions on how the player can play the game.
- Feedback helps the player know what is happening on the field.
- The player should be able to pick which team member they will control.
- A winning condition is present, and a winning/losing message is displayed.
- The player can restart the game after it ends.
- A readme file explains the AI techniques used to implement the game and how they would work together to achieve the required output.

You might need to implement an AI manager or use a blackboard to allow communication between different agents.

Extra features:

- The player can pick the number of agents in each team at the start of the game.
- The environment has some places for all agents to hide behind.
- We can change rule 10 and allow more than one agent to be on the opponent's side of the field but this would require changing the AI for both the player side and the other team's side:
    - For the Player team AI, they need to be able to decide who is going to help the player with the attack and who is going to stay back at the field to help protect the flag. This might include changing the initial decision to attack, into staying to defend, if needed.
    - The other team AI needs to decide who will be attacking and who will be defending the flags as well.

Guidelines

Build Quality:

- The source code is required to display the following qualities:
    - Free of:
        - Build errors and warnings
        - All intermediate files
- Any necessary documentation should be included as a separate Readme.txt file.

Interface Features:

The executable is required to provide an intuitive interface with the following features:

- Provide clear instructions.
- Controls are clearly identifiable and intuitive while playing.
- Design and layout of the UI make effective use of screen space.

Runtime Quality:

The source code is required to display the following qualities:

- Free of:
    - Bugs.
    - Crashes.

Submission Guidelines

Zip files containing the following.

- **Build Zip:** An executable must be zipped and included with the submission.
- **Readme.txt:** with details about the techniques used and any other dependencies.
- **Source Code Zip:** All relevant source code files and project files must be zipped and included with the submission.

An electronic copy of the source folder including only source code are required as described in the BuildQuality section.

Submit the exe zip as described in the executable build section.

Submit the appropriately named zip file to Blackboard with the structure:

Naming & file structure for the zip file.

- Assessment4_StudentName.zip
  - Build.zip
  - SourceCode.zip
  - Readme.txt

Assessment Criteria

- Core features and gameplay (80%)
- Extra features (15%)
- Software organization (5%)

Submission Policy

Please refer to the Submission Policy on Blackboard for further details.