Project Plan

Robbery Simulation using AI techniques

(Custom Project Plan – Task 18)

Name: Chathil Vithanage

Student ID: 104195065

Introduction

This document outlines my plan for the custom project in COS30002 - AI for Games. The goal here is to clearly explain what I'm aiming to build, how I plan to do it, and what kind of outcomes I'll produce to show what I've achieved. It's not just about writing code it's about showing how different AI techniques come together to solve a game-related problem in a smart and effective way.

For this project, I'm creating a robbery simulation where a Thief agent tries to steal an item and escape without getting caught by a Guard. The Thief will learn how to act using reinforcement learning, adapting based on how the Guard behaves. The Guard will use a Finite State Machine (FSM) to patrol, chase, or investigate depending on what's going on. I'll also use steering behaviors like seek, flee, and wander to make both agents move more naturally through the world.

This intro sets the stage for the rest of the plan, which covers what I'll document, what I'll build, and what I'll show in the final interview to prove that the project meets the goals of the unit.

Project Description

In this Project we aim to build a simulated Heist or a Robbery Scenario in a 2D graph-based environment where an Autonomous character "Agent (Thief)" will have to navigate through a set of terrains to steal an item and **escape** while avoiding the security agent guarding the item.

The Thief agent will be actively learning optimal strategies to avoid the Guarding agent. The Guarding agent will be a FSM which performs dynamic behaviors like **patrolling**, **chasing** and **investigating**. Movement throughout the terrain map will be guided through **seek**, **wander** and **flee** to ensure random and realistic interactions.

Learning Outcome Approach Strategy

ULO1 - Discuss and implement software development techniques to support the creation of AI behavior in games, focusing on non-obvious problem-solving approaches.

- In this project we aim to integrate both FSM and Learn based AI to address on the navigation and path planning showcasing hybrid approach on this project

ULO2 - Understand and utilize a variety of graph and path-planning techniques.

- We will be implementing a custom navigation graph that is built to represent the Path layout, where both the agents will be traversing.

ULO3 - Design and create realistic movements for agents using steering force models to simulate natural and responsive behavior.

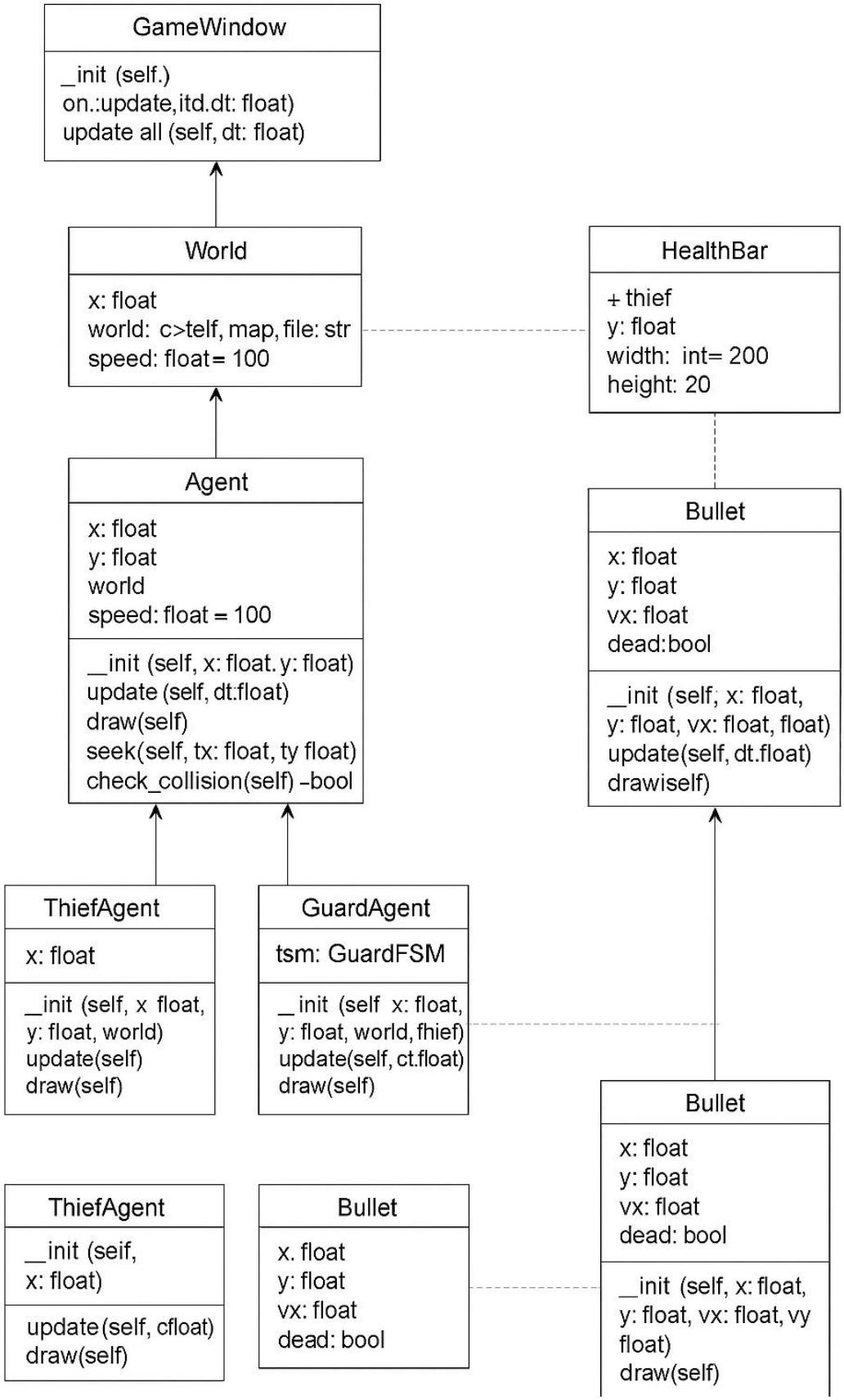
- For utilization of the behavioral pattern of the agent we will be using Seek, Flee and Wander which will guide the agent movement which create a more dynamic simulation

ULO4 - Design and develop agents capable of planning actions that require adaptive problem-solving techniques and innovative solutions.

- We plan on using reinforcement learning to adapt decision to be based on the environment which will have outcomes based on the past interaction to finding the best times for decision.

ULO5 - Combine multiple AI techniques to develop sophisticated game AI capable of solving intricate, multi-layered problems that arise in dynamic game environments.

- As we plan on implementing FSMs, Steering and graph-based movement , we will be integrating all these techniques into the same environment as the final output.


GameWindow
_init (self.)
on.:update,itd.dt: float)
update all (self, dt: float)
World
x: float
world: c>telf, map, file: str
speed: float= 100
HealthBar
+ thief
y: float
width: int= 200
height: 20
Agent
x: float
y: float
world
speed: float = 100
__init (self, x: float. y: float)
update (self, dt.float)
draw(self)
seek(self, tx: float, ty float)
check_collision(self) –bool
Bullet
x: float
y: float
vx: float
dead:bool
__init (self; x: float,
y: float, vx: float, float)
update(self, dt.float)
drawiself)
ThiefAgent
x: float
_init (self, x float,
y: float, world)
update(self)
draw(self)
GuardAgent
tsm: GuardFSM
_init (self x: float,
y: float, world, fhief)
update(self, ct.float)
draw(self)
Bullet
x: float
y: float
vx: float
dead: bool
__init (self, x: float,
y: float, vx: float, vy
float)
draw(self)
ThiefAgent
__init (seif,
x: float)
update(self, cfloat)
draw(self)
Bullet
x. float
y: float
vx: float
dead: bool

What I will document

As I go through the project, I'll document everything important mainly how the AI systems are set up, how they work together, and the key changes I make along the way. The aim is to keep things clear and easy to follow, both for me and for showing what I've done during the interview.

Here's what I'll include:

- Design and Implementation planning: explains how the FSM, RL, and steering behaviors are built and how they interact.
- Feature List: a simple breakdown of all the main things I'm planning to implement.
- Architecture: Sketches quick diagrams showing the structure of the code
- README Files: Basic instructions on running the Code
- Testing Results: Testing results on different Scenarios

All documents will be included in the GitHub Repository.

GitHub : https://github.com/ChathilJay/Custom-Project---Heist-Simulation

What I will make

The plan is to build a working 2D simulation of a robbery scenario where two agents interact in a smart, game-like environment. I'll create a Thief agent that learns how to act using reinforcement learning, and a Guard agent that's built using a Finite State Machine (FSM) to switch between patrolling, chasing, and investigating based on what's happening.

The whole thing will run on a graph-based map, which both agents will use for navigation and path-planning. To make their movement feel realistic , I'll use steering behaviors like seek, flee, and wander. These will help the agents respond naturally to each other and the environment.

Here's a breakdown of what I'm building:

- A graph-based 2D map layout.
- FSM-driven Guard agent with patrol and reactive behaviors.
- Thief agent that learns escape strategies using RL.
- Steering logic for agent movement (seek, flee, wander).
- Pathfinding system for navigating the map.
- A visual demo to show it all working.

Everything will be done in Python using Pyglet, NumPy, and possibly PyTorch for the RL side. The final goal is to have an interactive and intelligent simulation that shows off the AI systems working together.

Conclusion

This project is my way of putting together everything I've learned in this unit into one solid simulation. I'm combining FSMs, reinforcement learning, steering behaviors, and graph-based navigation to build a working system that actually shows intelligent agent behavior. It's not about building a full game it's about showing that the AI works, and that I understand how to make these systems interact in real time.

Everything from the code to the documentation will be kept in the GitHub repo, so it's all tracked and easy to follow. Whether I finish the full simulation or just get the core parts running, I'll be able to explain the process and show clear progress during the final interview.

At the end of the day, this plan helps me stay on track and make sure I've got something to show that ticks all the boxes for this unit.