Spike: 13
Title: Tactical Steering (Hide!)

Author: Hoang Bao Phuc Chau, 103523966

Goals / deliverables:
Create a hunter-prey agent simulation for two or more agents, in which "prey" agents avoid "hunter" agents by
concealing themselves behind objects in the environment. The simulation must:
• Include several "objects" that prey can hide behind (simple circles).
• Show a distinction between the "hunter" and "prey" agent appearance and abilities.
• Show an indicator ("x" or similar) to indicate suitable "hide" locations for prey to select from
• Prey agents must select a “good” location, and head to it, based on tactical evaluation.
• Do NOT hide “inside” objects - rather find a location outside (behind).

Technologies, Tools, and Resources used:
List of information needed by someone trying to reproduce this work

- Pycharm
- Python 3.12
- Pyglet 2.0.15
- ChatGPT AI
- Microsoft Copilot AI

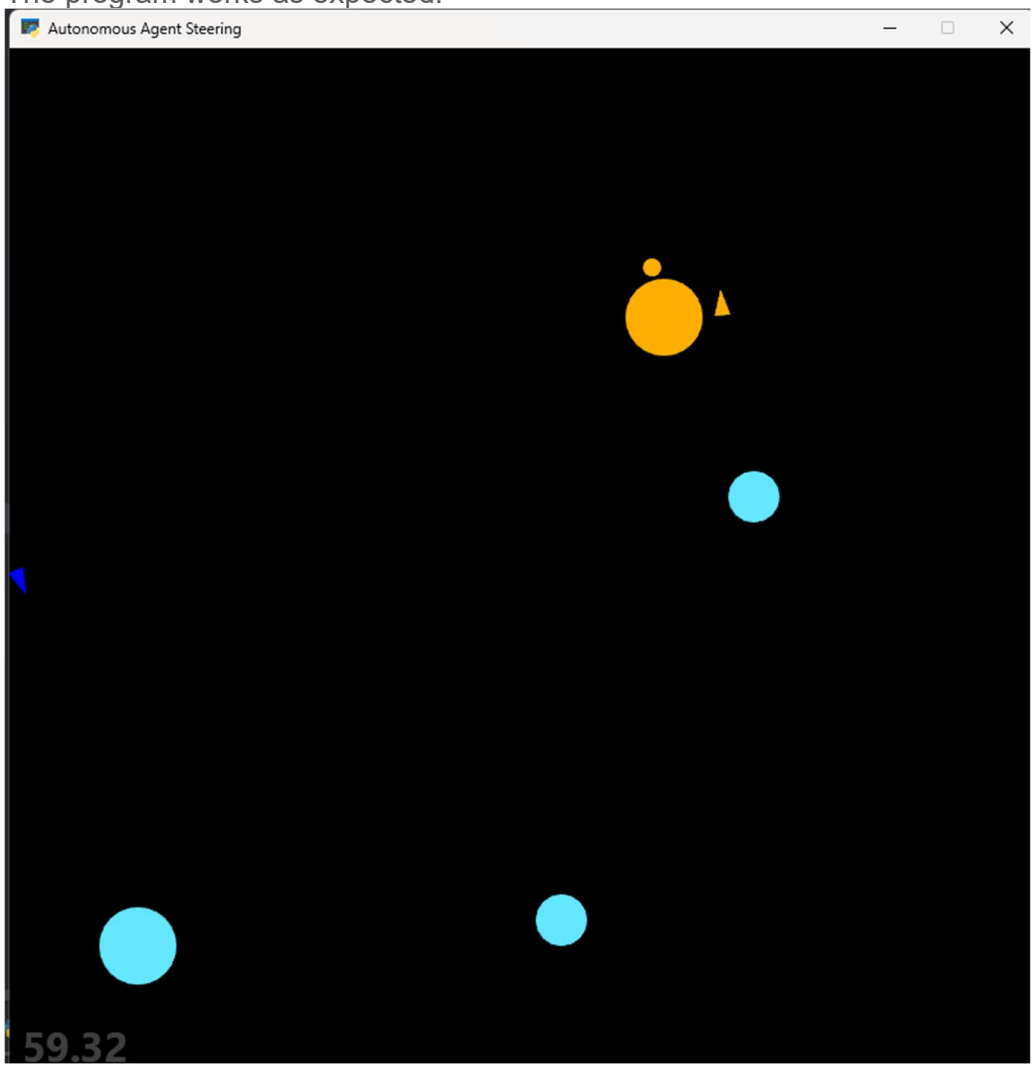
Tasks undertaken:

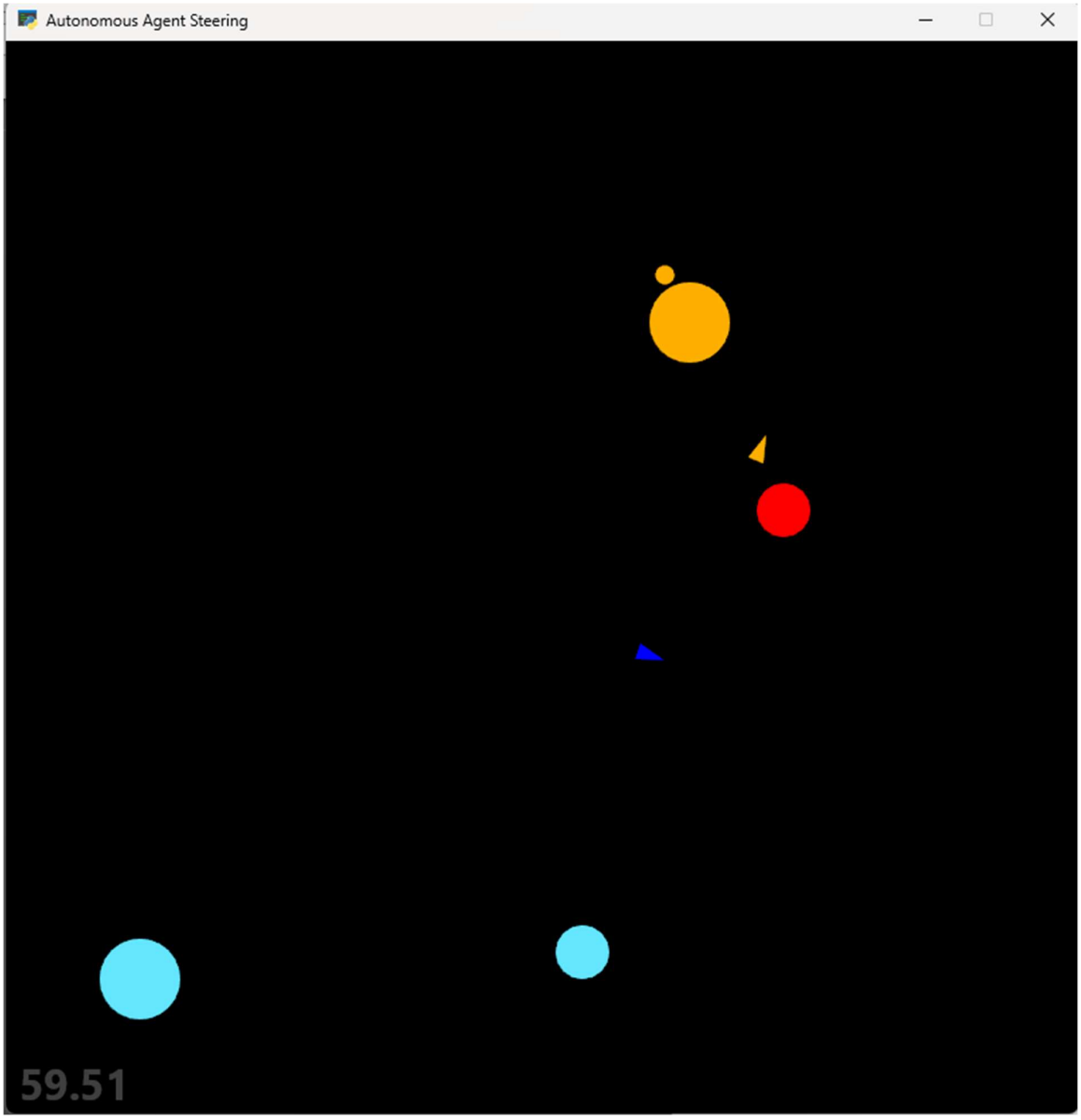
- Copy codebase from task 12
- Since this program doesn’t have the ability to append agents, change mode. So it only have the option for pausing the screen, which can be done by pressing P.
- Create a new class named Obstacle inside a separate file (obstacle.py)
- Inside Obstacle class, initialise it with a vector has position (100, 100), radius = 30. Also, initialise it with 3 types of circle: main circle, target circle, and emphasised circle. The last 2 one have the colour of INVISIBLE and the 2nd one have a radius of 7.
- Declare a method named is_safe to check if the distance from the circle to the hunter is safe (safe distance depends on each person’s idea, I took 50 + self.radius * 2)
- If a circle is safe, it will have aqua color, else it will have red color
- Update method will call is_safe
- Reset method will make the circle teleports to a random position
- Back to agent.py, create 2 classes that inherited from Agent class.
- Inside Agent class, everything is almost the same, except for some changes:
    - Delete all the codes related to Path class
    - Inside calculate() method, delete all codes and alert the system that this method must be overridden
    - Write a function to check if a selected obstacle is nearby us, alert distance is obstacle radius + 25.
    - Write a function to check for nearby obstacles

- Inside Hunter class, do the following:
  - Initialise it with scale = 20, mass =5, color = ‘BLUE’
  - Copy all wander details and wander function from Agent class
  - Inside calculate method, set mode to wander and acceleration type to wander. If the hunter is near the obstacle, flee from it.
- Inside Prey class, do the following:
  - Initialise it with scale = 30, mass =1, color = ‘ORANGE
  - Write a boolean function to check if there is a Hunter nearby.
  - Create a function to check for the best obstacle to go to, using is_safe. Loop through all the obstacles, then choose the one having the furthest distance.
  - Inside calculate() method, if the prey is not alerted, become idle, else calculate the position of the nearest obstacle to go.
  - If the circle become the best circle, it will have ORANGE colour
- Inside world.py, initialise it with Prey and Hunter instead of using only Agent.

What we found out:

The program works as expected:





Open issues/risks [Optional – **remove** heading/section if not used!]**:**
List out the issues and risks that you have been unable to resolve at the end of the spike. You may have uncovered a whole range of new risks as well.

- eg. Risk xyz (new)

Recommendations [Optional – **remove** heading/section if not used!]**:**
Often based on any open issues/risks identified. You may state that another spike is required to resolve new issues identified (or) indicate that this spike has increased your confidence in XYZ and should move on.