**Spike:** 15

**Title:** Agent Marksmanship

**Author:** Ben Holmes, 103024841

**Goals / deliverables:**

Create an agent targeting simulation with:

(a) an attacking agent (can be stationary),

(b) a moving target agent (can simply move between two-way points), and

(c) a selection of weapons that can fire projectiles with different properties.

Be able to demonstrate that the attacking agent that can successfully target (hit) with different weapon properties:

(a) Fast moving accurate projectile. (Rifle)

(b) Slow moving accurate projectile. (Rocket)

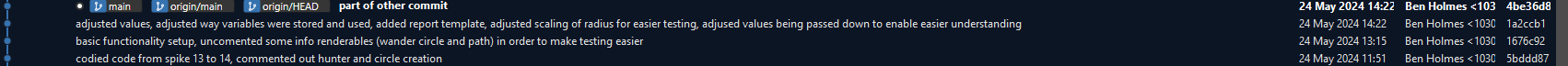
(c) Fast moving low accuracy projectile (Hand Gun)

(d) Slow moving low accuracy projectile (Hand grenade)

**Technologies, Tools, and Resources used:**

* Visual Studio Code
* Python 3.12.2
* Pyglet

**Tasks undertaken:**

****

* Copied code from spike 13
* Commented out circle creation code
* Created the patrol functionality with preplace patrol points and added a target agent set to patrol

Agent.py

A computer code on a black background

Description automatically generated

World.py

A computer code with numbers and symbols

Description automatically generated with medium confidence

* Created the variables for bullet mode, speed and mode selection

A computer screen shot of a program code

Description automatically generated

* Adjusted the hunter agent to start in a hold mode (no motion) in the centre of the screen and created an empty array for bullets and a world bullet mode storage for switching between

A screen shot of a computer code

Description automatically generated

* Created the bullet class in separate file bullet.py with a vehicle of a smaller red arrow and a render of the predicted position as a green cross

A screenshot of a computer with Marfa lights in the background

Description automatically generated

A screen shot of a computer program

Description automatically generated A screen shot of a computer program

Description automatically generated

* Added a shoot mode and function to the agent and adjusted the math around it to improve its aim including adding an iteration method to improve the predicted position

A screen shot of a computer program

Description automatically generated

* Added the mode change buttons and shoot button to the world input\_keyboard





A screen shot of a computer program

Description automatically generated

* Added update, check\_hit and check\_ lifetime to bullet to enable removal of bullet after set time (based on speed) and ability to check hit on bullet level

A computer screen shot of a black screen

Description automatically generated

\_\_init\_\_

A black background with white text and numbers

Description automatically generated

* Setup hit check and bullet life check in update of world

A screen shot of a computer program

Description automatically generated

* Added hit functionality to agent.

Init

A black screen with white text

Description automatically generated

A computer screen shot of a black background with white text

Description automatically generated

* Added inaccuracy to bullet in bullet init

A screen shot of a computer

Description automatically generated

**What we found out:**

Patrol was incredibly easy to implement, as was the bullet class and basic functionality of getting it to shoot at a point. Getting the hit accuracy when the bullet was accurate was difficult however.

It was adjusting the code to have an iterative calculation and adjusting the pursuit provided calculation to use distance divided by only bullet speed that allowed the accuracy to be good, it has trouble with the accelerating target but once the velocity is stable its perfectly accurate.

To help with the accuracy while accelerating I adjust the agent to be a little slower and doubled the speed of the bullets that I originally selected.

Figuring out how to implement inaccuracy was a little difficult as adjusting the angle of the vector directly was too tricky so instead I added a 1-10% strength positive or negative perp of the bullet vector to add the inaccuracy which can be seen by looking at the predicted green star

**Testing buttons:**

Numpad minus shoots a bullet and numpad:

1: rifle (fast accurate)

2: rocket (fast inaccurate)

3: pistol (slow accurate)

4: grenade (slow inaccurate)