Spike Outcome Report

Number: 9
Spike Title: Agent Markmanship
Personal: Michael Williams (7668481)

Goals:

- Create an agent targeting simulation with:
  - an attacking agent (can be stationary)
  - a moving target agent (can simply move between two way-points)
  - 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:
  - Fast moving accurate projectile. (Rifle)
  - Slow moving accurate projectile. (Rocket)
  - Fast moving low accuracy projectile (Hand Gun)
  - Slow moving low accuracy projectile (Hand grenade)

Technologies, Tools, and Resources used:

- Python IDE
- Sample Lab Code
- Lecture Material
- Various online resources (physics)

Tasks undertaken:

1. To make this spike simpler, instead of starting from scratch, we used on of the previous spikes as the base. We chose Spike 7, as this already featured all of the basic functionality we would need and could be adapted quickly to suit this purpose.

2. We started by refactoring the agent code into 3 distinct classes:
   a. The shooter
      This class does not need to do anything except sit in a stationary location
   b. The target
      This class just moves at the end of the screen in a sort of shooting range.
   c. The bullet
      This class needs to be able to move from the shooter to the target location.
3. We worked on getting the agent and the target working first adding variables to the game world where necessary before moving on. Once this was functional, we added the ability for the shooter to shoot a bullet using a weapon factory.

   ```
   def fire_weapon(self):
       weapon = WeaponFactory().GetWeapon(self.weapon)
       weapon.Fire(self.world,self.pos)
   ```

   This action spawns a bullet and fires it at the target.

4. To see this all working, we added the key bindings to fire the weapon.

5. Now that everything was working there were two major components left.
   a. Making the bullet hit the target
   b. Checking for a collision

To make the bullet hit the target, we used some online references and tutorials to aid us with the physics, but we understood exactly what needed to happen. As a result, our final code to aim at the target looked like this:

```
def target(self, target):
    ''' move towards target position '''
    target_offset = self.pos - target.pos

    target_distance = PointToVector2D(target_offset).length()

    target_direction = PointToVector2D(target_offset).normalise()

    relative_velocity = target_direction * self.speed

    relative_speed = relative_velocity.dot(target_direction)

    if (relative_speed <= 0.0):
        intercept_time = 1.0
    else:
        intercept_time = target_distance / relative_speed

    intercept_location = target.pos + target.vel * intercept_time

    intercept_point = intercept_location - self.pos
    if self.miss():
        print("miss")
        intercept_point += Point2D(100,100)

    aim = PointToVector2D(intercept_point).normalise()
    vel = aim * self.speed

    return vel
```

The second part, checking for a collision, we implemented in a fairly rudimentary manner, since we know that the bullet is always coming from the front face of the target, we restrict the collision detection based on this assumption, and do something like this:

```
def check_for_collision(self):
    rightmost_point = Point2D(self.pos.x + self.radius,self.pos.y)

    if rightmost_point:
        if self.world.target.point_inside(rightmost_point):
            print('hit')
            self.world.bullet = None
            self.world.target.hit()
```

6. To demonstrate how this reacts with different weapon physics, we added in a few kinds of different weapon types (in line with the spike goals) to demonstrate slow and fast projectiles as well as accurate and inaccurate options.

Key Bindings:

- Shoot: Space
- Change Weapons
    - Rifle: 1
    - Rocket Launcher: 2
    - Pistol: 3
    - Grenade: 4

What we found out:

We found out how to hit a moving target with a projectile, specifically:

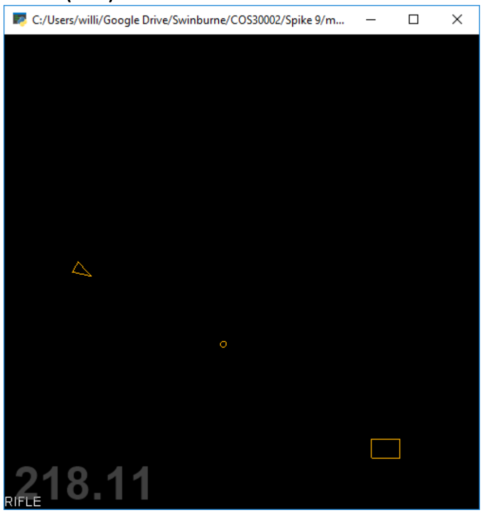
- How to predict the final location of an agent
- How to determine a "hit" through collision detection

The biggest learning curve for this task was learning how to calculate the final position for the agent, in order to give the bullet the correct trajectory. In order to overcome this, we used a number of online physics resources to learn how to calculate this. But even after doing this we were still having some problems as the model we were using to calculate this did not take into account the object starting at rest correctly and we were always missing. To counter this, we removed the acceleration component from the bullet physics so it was fired at top speed already.

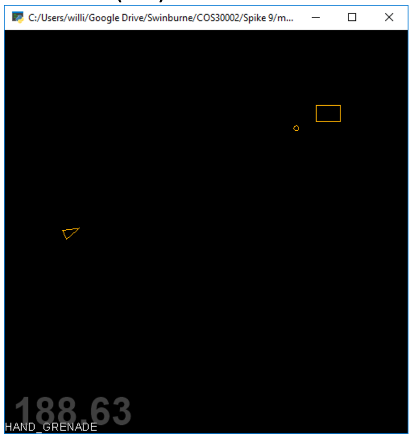
In the end, we were quite successful with this spike, correctly modelling the four different kinds of guns. Some improvements that could be made would be to adjust the calculations so that the acceleration of the bullet is also considered. Another good addition would be to have multiple targets and have the agent (shooter) use some sort of tactical analysis to determine which one they should "attack" first.

Screenshots

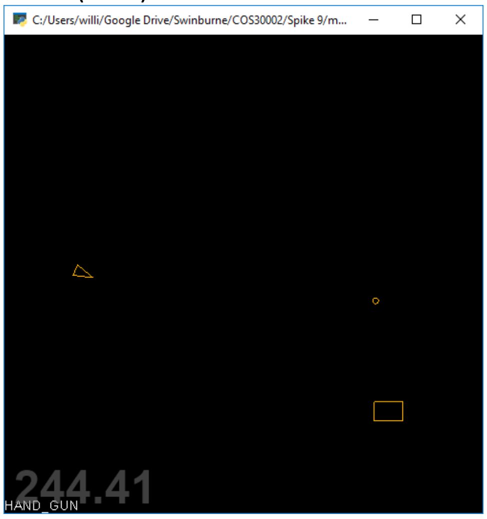
Rifle (Hit)



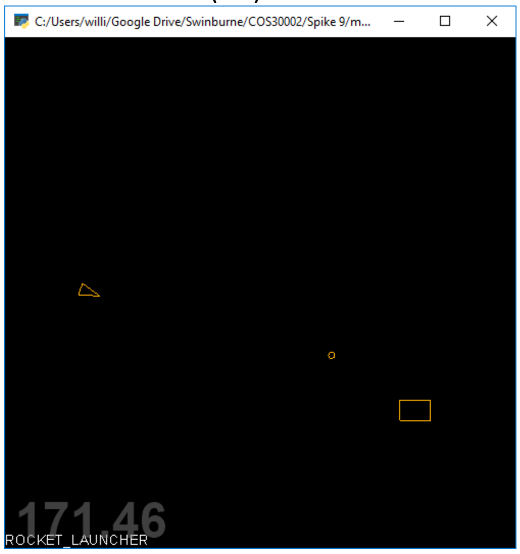
Grenade (Hit)



Pistol (Miss)



Rocket Launcher (Hit)


C:/Users/willi/Google Drive/Swinburne/COS30002/Spike 9/m...
171.46
ROCKET_LAUNCHER