Spike Outcome Report

Number: 04

Spike Title: Agent Marksmanship

Personal: Peter Argent (7649991)

Goals:

Create an agent targeting simulation with:
• an attacking agent (can be stationary),
• a moving target agent (can simply move between two-way points), and
• a selection of weapons that can fire projectiles with different properties.

Can 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:

- Visual Studio 2017 with Python 3 installed
- Pyglet Documentation here: http://pyglet.readthedocs.io/en/pyglet-1.3-maintenance/
- Help from peers.
- Python 3 Documentation http://docs.python.org/

Tasks undertaken:

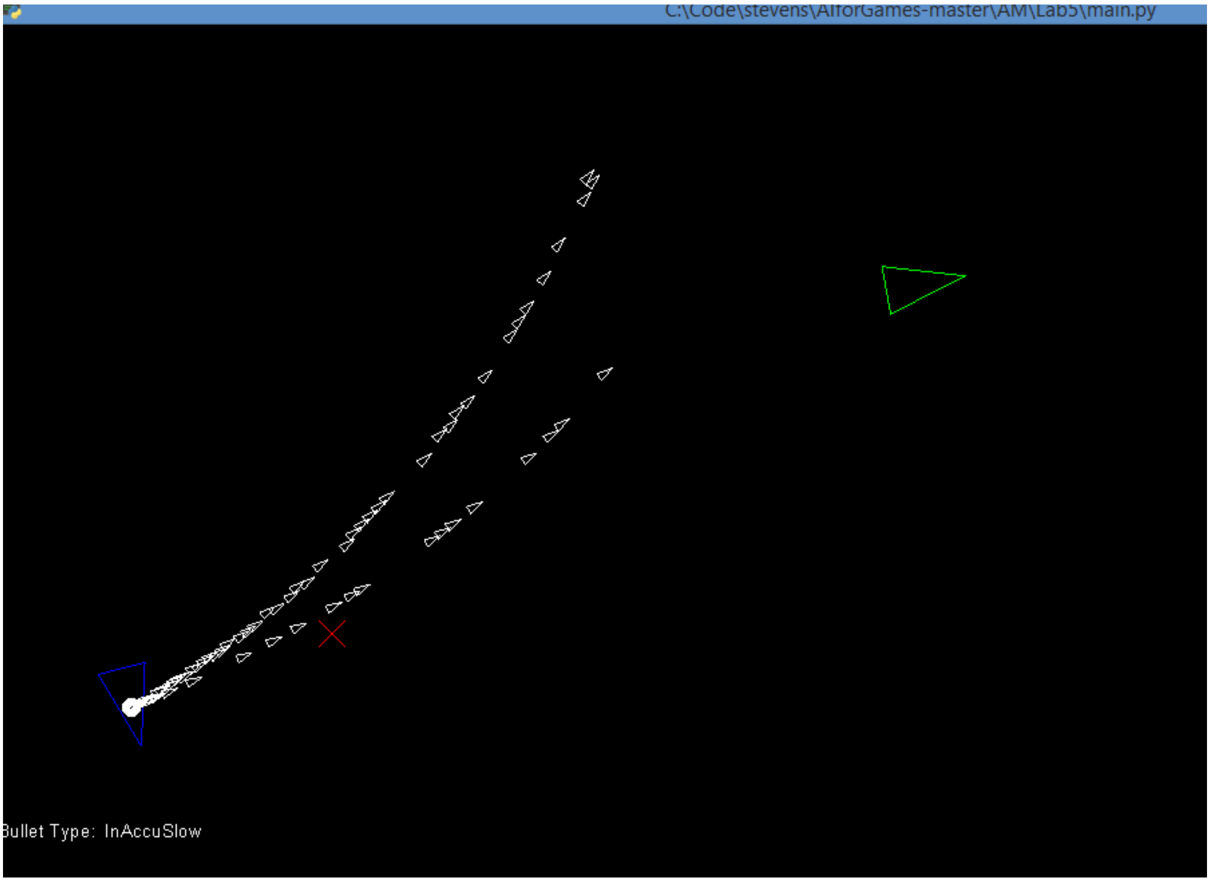
Using the world code from the tactical analysis/movement behaviour spikes we

- Add an attacking agent which has the following (agent.py)
    - X axis as a locked position
    - Y as a random value between the height of the window and zero.
        - This is for variance between how the agent can different predict the targets location from different locations
    - No other behaviour
- The target agent (Target.py):
    - Can be still
    - Or Travel between two random points
- Added a bullet (weapon.py)
    - Took code from pursuit to allow for the bullets movement
    - There is a check for accuracy to stationary target
    - Add Bullet types for Inaccurate-Accurate axis and Slow-Fast axis
        - InAccuFast, AccuFast, AccuSlow, and InAccuSlow
    - For each shot taken:
        - Randomly generates numbers between 0-10 and if the value was lower than the value of the bullet type it was an accurate shot, if higher than it was an inaccurate shot.
        - If Inaccurate added an extra 100 to each the x and y axis

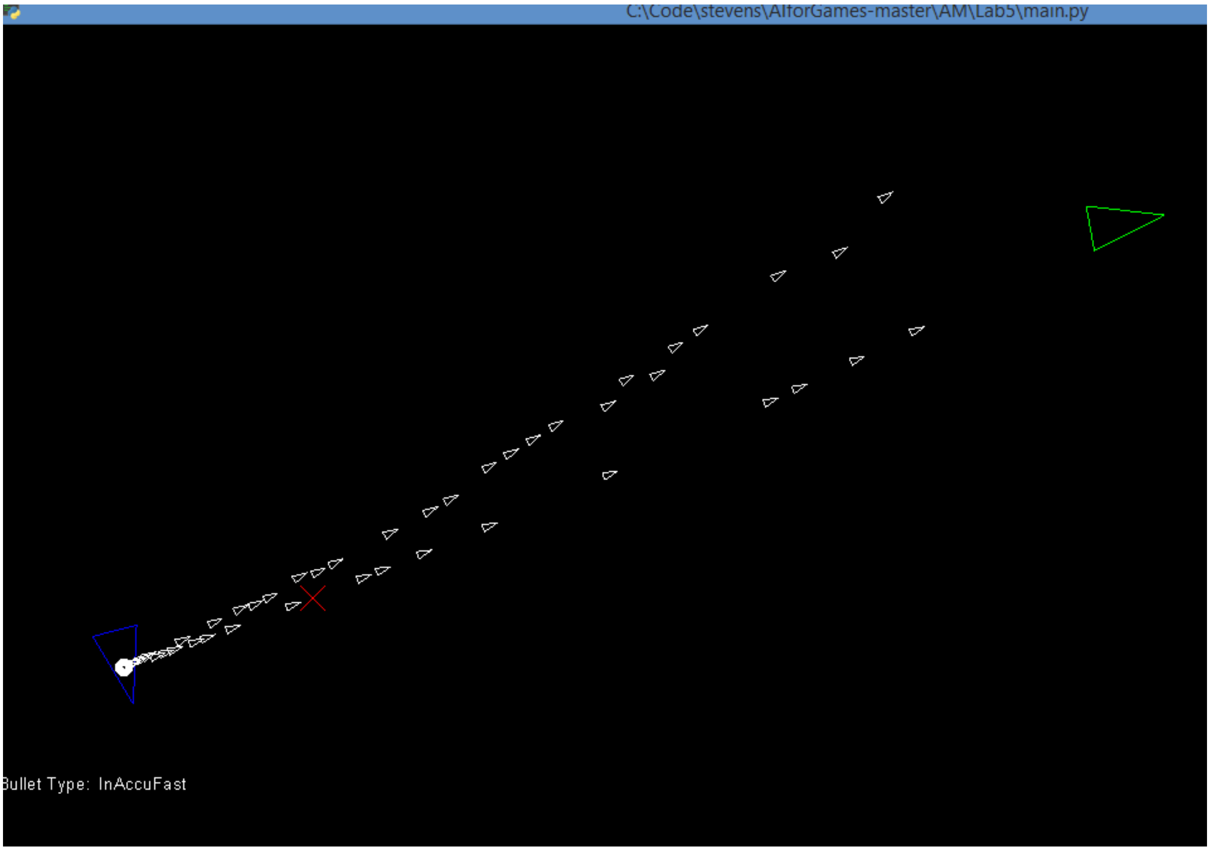
    - For When the target is moving:
        - The pursuit code is edited so that instead of a LookAhead variable we use the *targets pos* plus *vel* minus the *bullets vel*
        - Additional accuracy check copied and pasted into correct place
        - Added bullet types movement based on a moving target
        - For each shot taken:
            - Randomly generates numbers between 0-10 and if the value was lower than the value of the bullet type it was an accurate shot, if higher than it was an inaccurate shot.
            - If Inaccurate added an extra 50 to each the x and y axis
- Added code for when the target is hit:
    - Targets colour changes to red
    - Target slows down
    - Unable to fire for 100 ticks

What we found out:

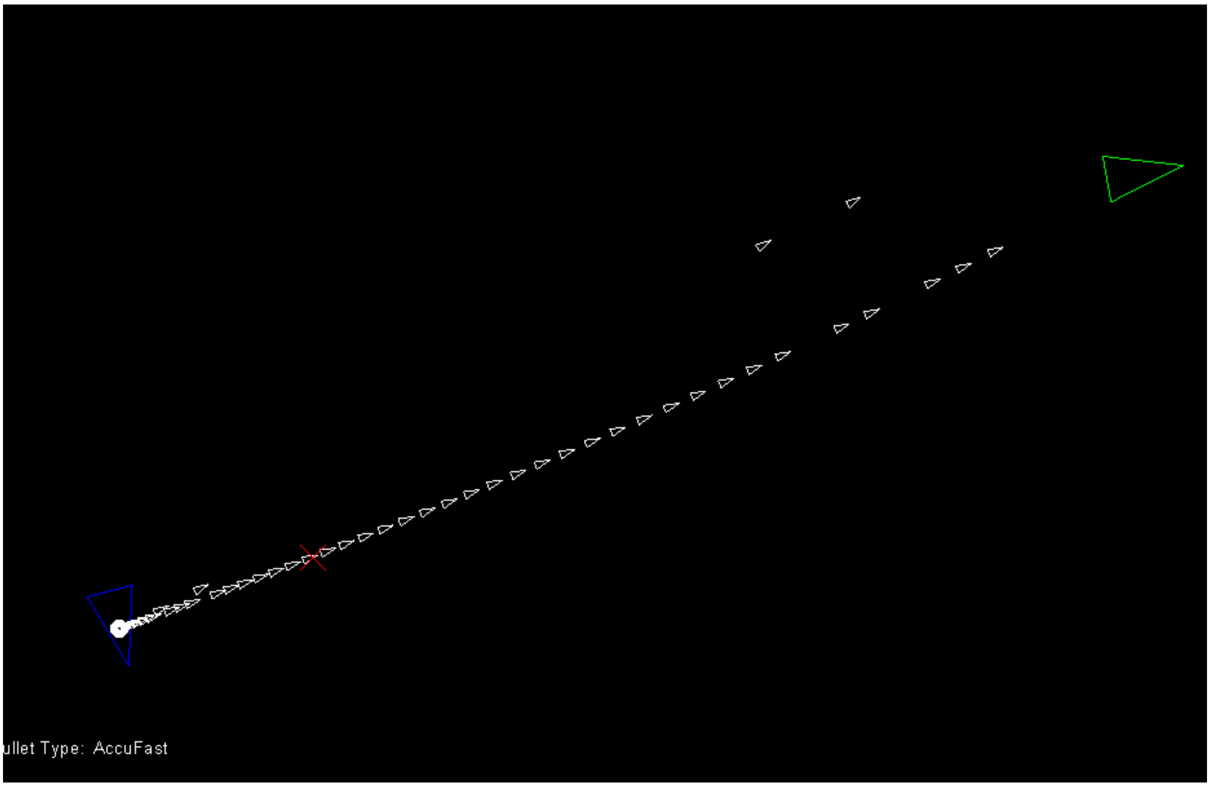
The Following pictures show the agent auto firing at a moving target. From them we see the spread and follow of each bullet type



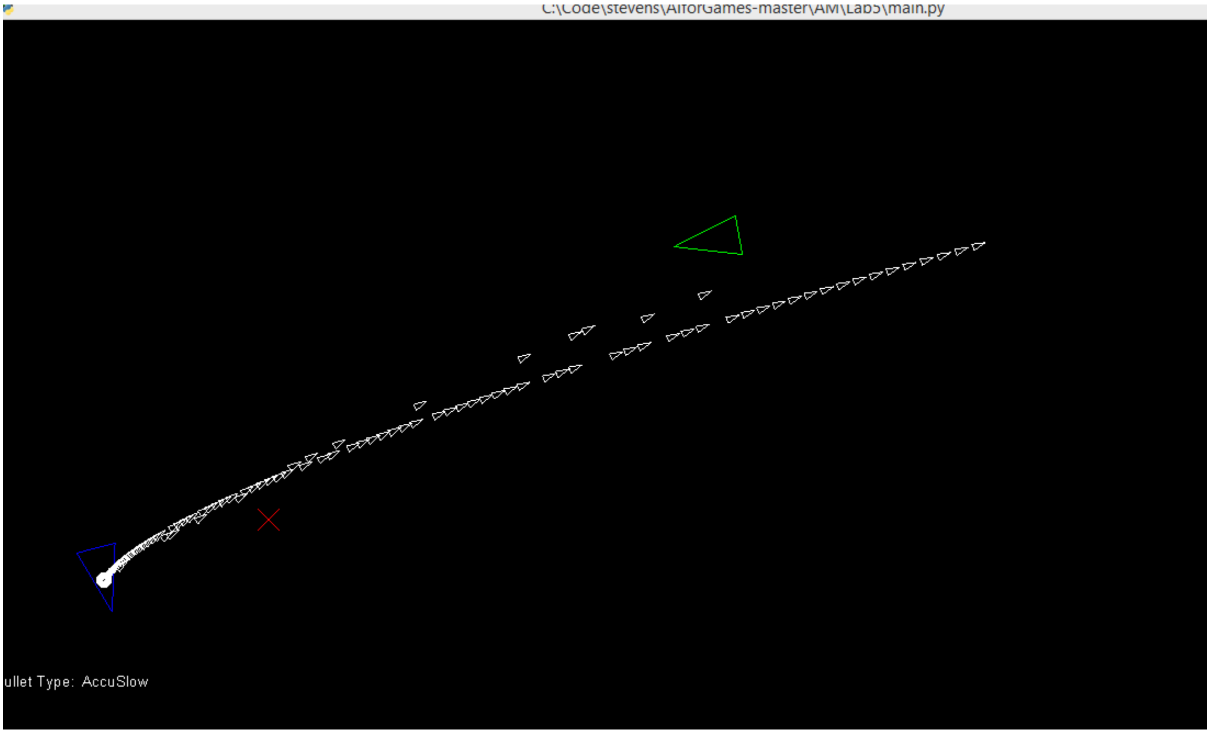
Bullet Type: InAccuSlow



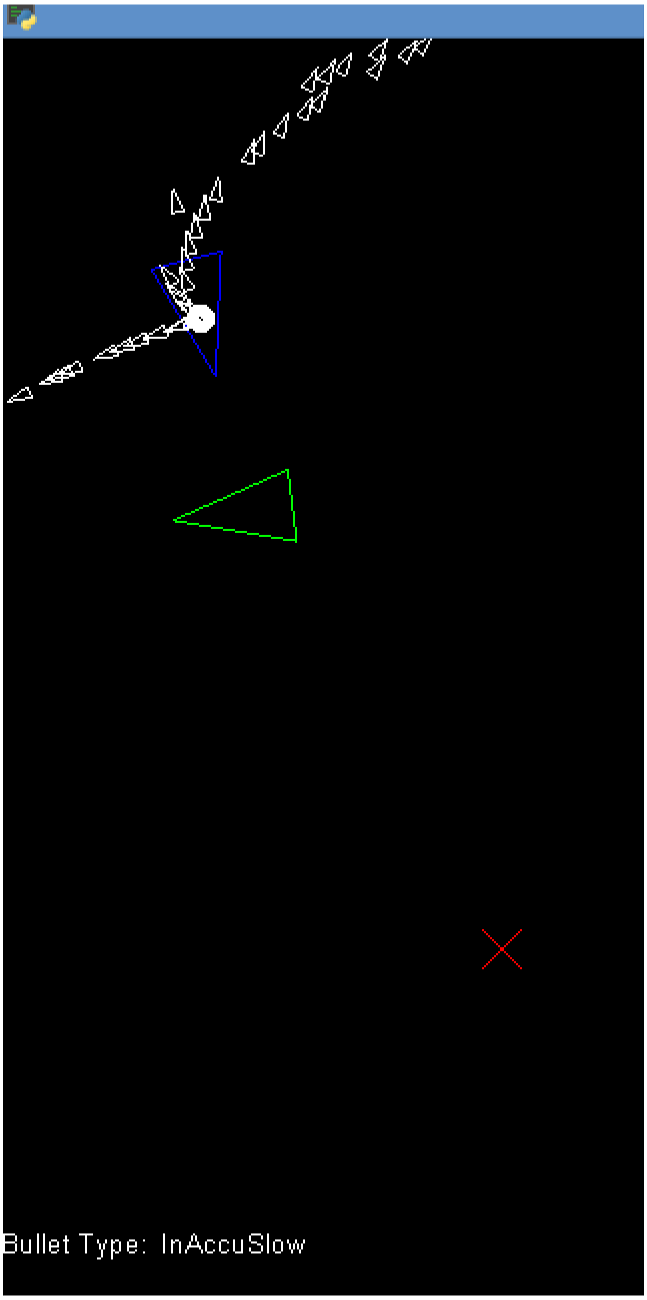
```
Bullet Type: InAccuFast
```



```
ullet Type: AccuFast
```



We can defiantly tell where the shoots fall on the inaccurate-accurate scale and the fast-slow scale. The bullets are working as intended, for the most part


Bullet Type: InAccuSlow

In this picture we can see the trouble the agent has when it is close to the edge with an inaccurate shot type.

Open issues/risks [Optional]:

The major open issue that can be seen in the spike is the issue of bullets behaving weirdly when an inaccurate gun is used close to the corners of the world.

Notes:

World Keys:
Fire Mode:
Q – InAccuSlow (Slow, Inaccurate)
W – InAccuFast (Fast, Inaccurate)
E – AccuFast (Fast, Accurate)
R – AccuSlow Main (Slow, Accurate)
Control Keys:
T – Make the target move between two points
Y – Freeze Target in place

S – Have the Agent Shoot at the target
D – Move Shooting Agent to a random y position in the world
G – Toggles Auto Fire
H – Remove Bullets