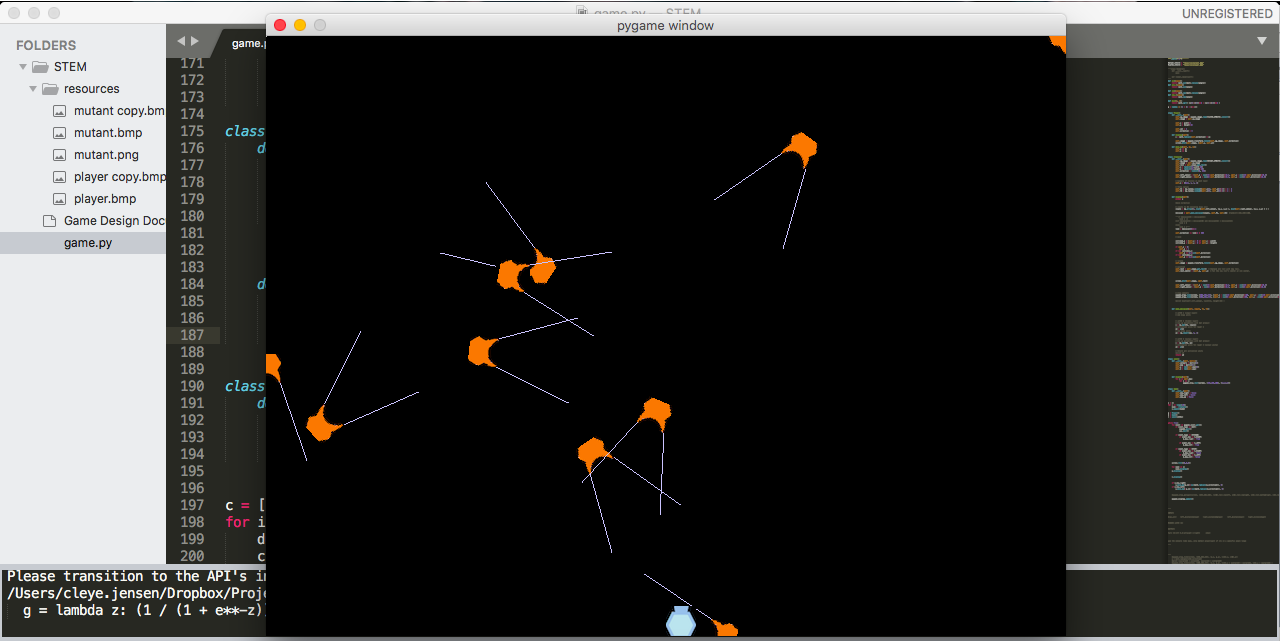
Game Design Document

I didn’t know what my game was going to be until June. Planning for the game began early 2018 when the theme was released. The aspects that stood out the most from Transformation were the mutation and evolution. I had dabbled in machine learning and neural networks before and was inspired by genetic algorithm simulators on YouTube to create something similar. At first I had no clue what to make, early brainstorms developed ideas like evolving from a fish to a human, training a car how to move through mazes and even a plant growing game. ~~In spite of the continual emphasis on planning, planning, planning, at the time of writing this game document in early April I still don’t know what my game will be~~



**Game Title**

The game is named after the game’s protagonist Dr. Darwin, similar to how Donkey Kong is named after the game’s protagonist. Given the premise of the game, it is obvious the title is a reference to Dr. Charles Darwin.

**Game Description**

Evil scientist Dr. Darwin is on a mission to evolve the perfect killing creature. Survive every generation by shooting and deviating the creatures. How many generations can you survive before the creatures outsmart you?

**Audience**

Dr. Darwin is intended for all ages but primarily adolescents, teenagers. Anyone who enjoys critical thinking games and/or has an interest in evolution simulators

**Characters/Roles**

The protagonist is an evil scientist who moves about in a flying saucer. He is named Dr Darwin after Charles Darwin who invented the theory of evolution. His objective is to breed a unique creature who can effectively target the player while also dodging incoming bullets shot by the player

The player is a . They can move about, shooting at creatures. The player has a health bar which decreases when they shoot a bullet or are attacked by creatures.

Dr Darwin breeds creatures who have the ability to detect the player, bullets, walls and vary its speed and direction based on its detections.

**Environment**

The game

**Theme**

I tried to emulate a type of minimalist retrofuturism style, one with simple, bright colours, non-complex shapes.

retrofuturism, tron,

**Objective/Goals**

The game

**Perspective**

The game

**Controls**

The game

**Reference points/Originality**

The game

**Platform**

when I tested on windows I found the game was much much faster

**Development Environment**

The game

**System Requirements**

The game

**Resourcing/Capability**

The game

**Style**

The game

**Process**

The game

**Timeline/Deadline**

The game

**Responsibility**

The game

**Submission Guidelines**

The game

**Problems**

There were many problems I had to deal with

The first major problem I had was learning how to rotate the characters. In pygame, the size of the rectangle container of an image was changed when the image rotated making the character’s movement very erratic. This took a week to figure out

As game development went on, ideas that were too hard to implement were scrapped and new ideas were taken on board

Firstly I planned what inputs and outputs the creature’s neural network should have

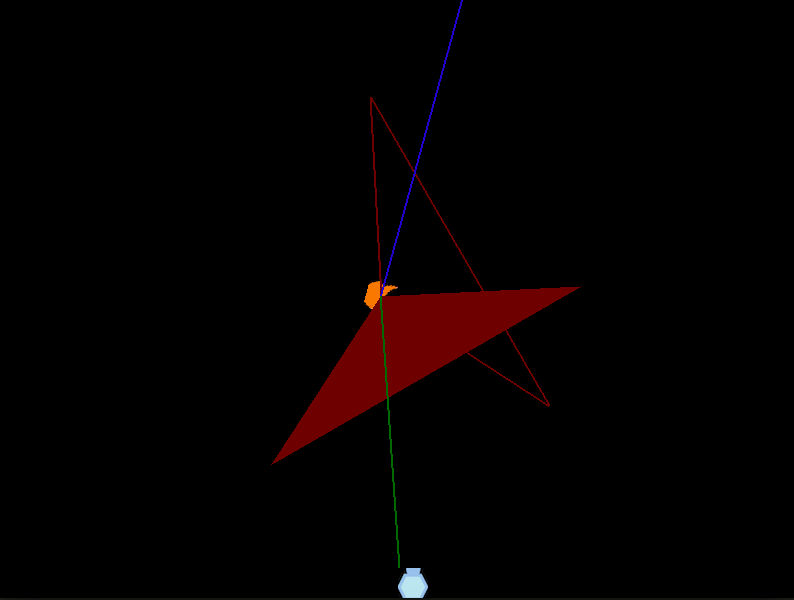
INPUTS bias\_unit left\_distance2player right\_distance2player left\_distance2wall right\_distance2wall

HIDDEN LAYER (4)

OUTPUTS turn\_left turn\_right shoot

At first I planned to make the creatures detect players by how far away the left side and the right side of the creature were from them. However if both eyes were 600pixels away from the creature, it doesn’t tell much

Instead I would use the sensors like eyes, only detect player/wall if its in a specific angle range



Another issue was the turn output had 2 different values: turn\_left and turn\_right, but this looked jittery and unnatural

The alternative was to create 0-1 turning values for the direction

The evolutionary algorithm would work by evaluating all the creatures fitnesses. Their fitness would then be taken into account when half the population is sampled weighted on their fitness (i.e. so while the fittest creature had the highest likelihood of breeding it wasn’t guaranteed to be chosed) this was to ensure diversity in the creatures genes. The chosen half of the population was breeded by mutating their genes, this was done by slightly varying the parameters in the creature’s neural network

Fitness was based on rewards and punishments, all were chosen to breed creatures that could chase and attack the player and dodge bullets.

Rewards are:

damage done on player

average distance to player

time with player in sight

--ability to sense bullets and dodge them--- too hard

Punishments are:

being stationary

running into walls

being hit by bullets

The creatures in each generation have 20s to show their fitness.

Hopefully too many creatures with computationally expensive neural networks won’t slow down the game, so that limited the population in each generation

to do:

music

storyline

gui

Other ideas that never made it:

The creature shoots back

Levels

some creatures drop powerups when they die

4 waves (a wave is 16 specimens of a species) come for each generation

species:

twirlers, sharpshooters

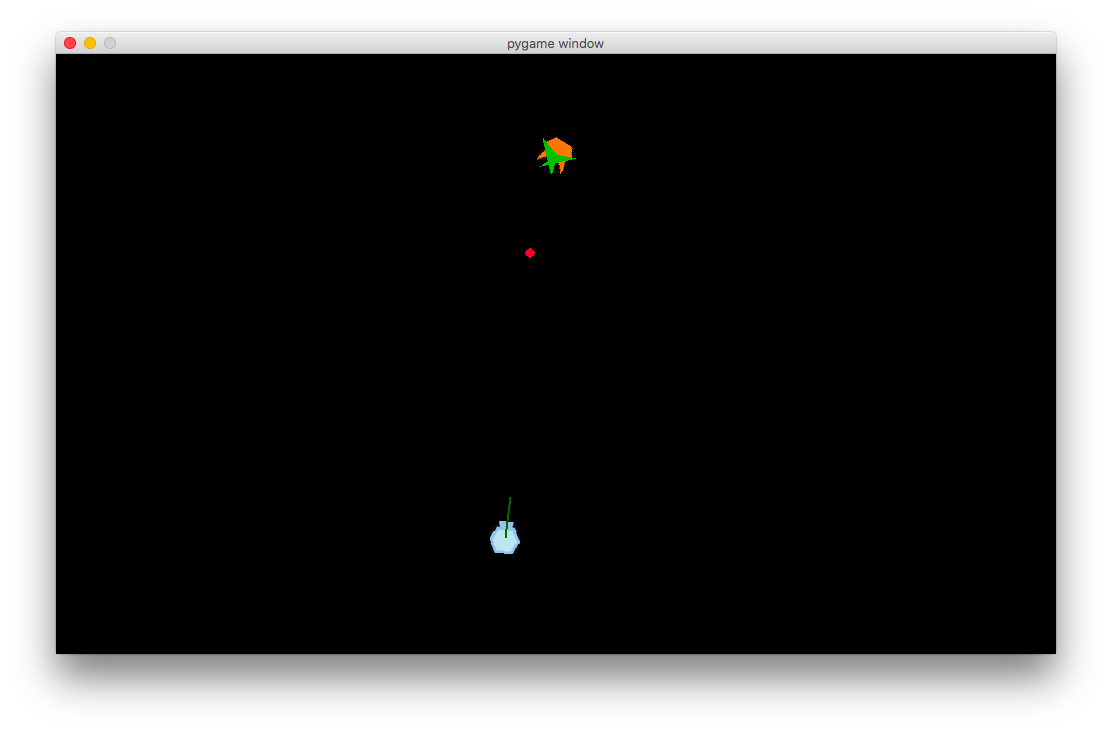
survive as many generations as possible

better start gui

borrowed button class off old program

problem: bullets were sensed even when moving away from them

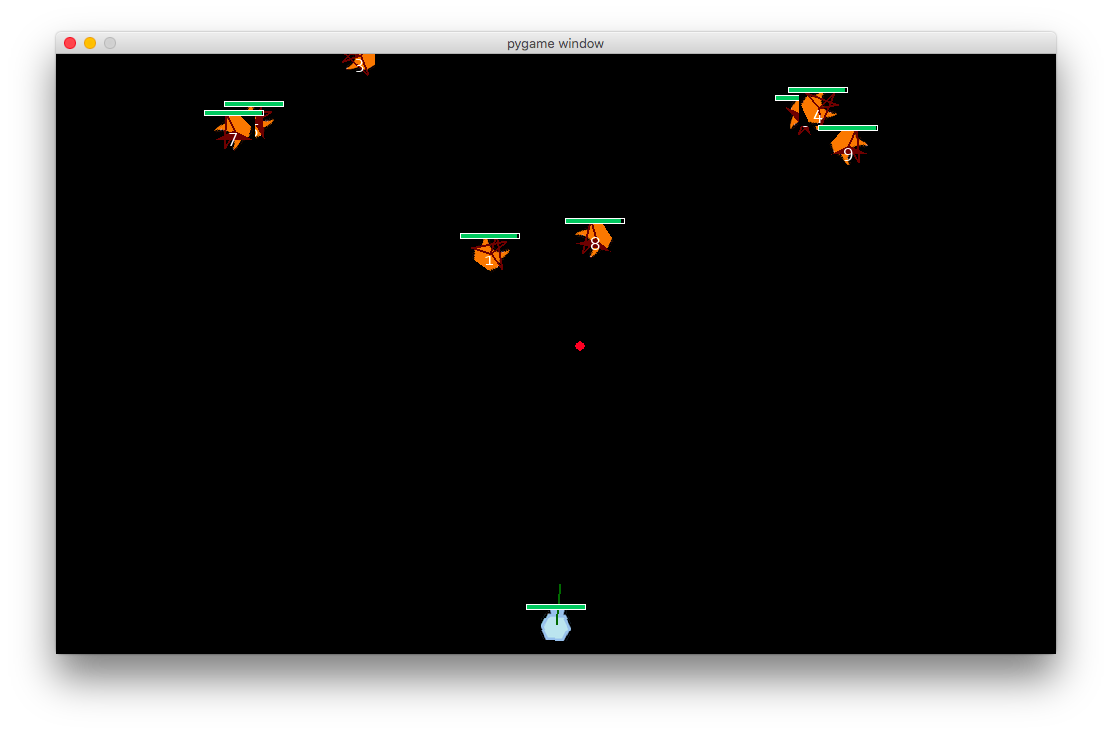
added angles to sensor, add periphery value



added distance to player, bullet sensing, wall sensing as parameters

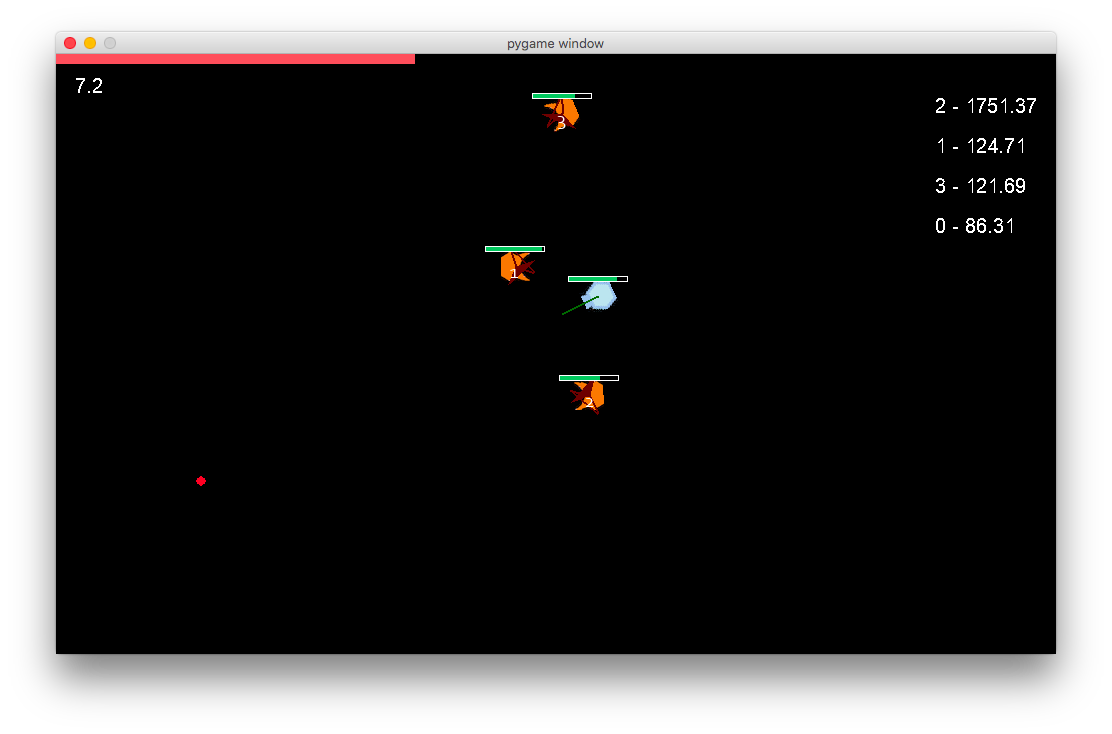
wall proximity function

added speed output



I’ve had the idea that the creature has limited energy or life and this is expended the faster it moves, and also shooting it depletes its life by a random amount

if that’s the case I need an overarching class for movable objects which player and creatures inherit from



techniques:

try lure them into walls

move around them to get beind their back

fitness starts at 0:

reward:

damage done on player

average distance to player

time with player in sight

--ability to sense bullets and dodge them--- too hard

punish:

being stationary

running into walls

hitting bullets

Creature

- update

- display

- calculate\_fitness ->

- make\_decision -> output units

interesting notes:

some creatures were attracted to the bullets,

half the creatures were really smart, the other half fairly dumb

what inputs and outputs should the nn have

INPUTS(6) bias\_unit, left\_sensor\_detect, right\_sensor\_detect, distance2player, left\_sensor\_detect\_bullet, right\_sensor\_detect\_bullet, wall\_proximity

HIDDEN LAYER (4)

OUTPUTS(2) turn (0-left 0.5-straight 1-right) velocity

optimum creature:

if not detecting the player, rotate

if detecting player with both eyes, move forward

if close to player, move faster

if detect bullet in one eye, move away and fast

if detect bullet in both eyes, move away and fast

if close to wall, rotate

I saved music for last because I didn’t want to hear whatever song I put in played over and over again while I’m testing the game

approaching

I had no idea how to do the music then I used noteflight and audacity to record a piano sample

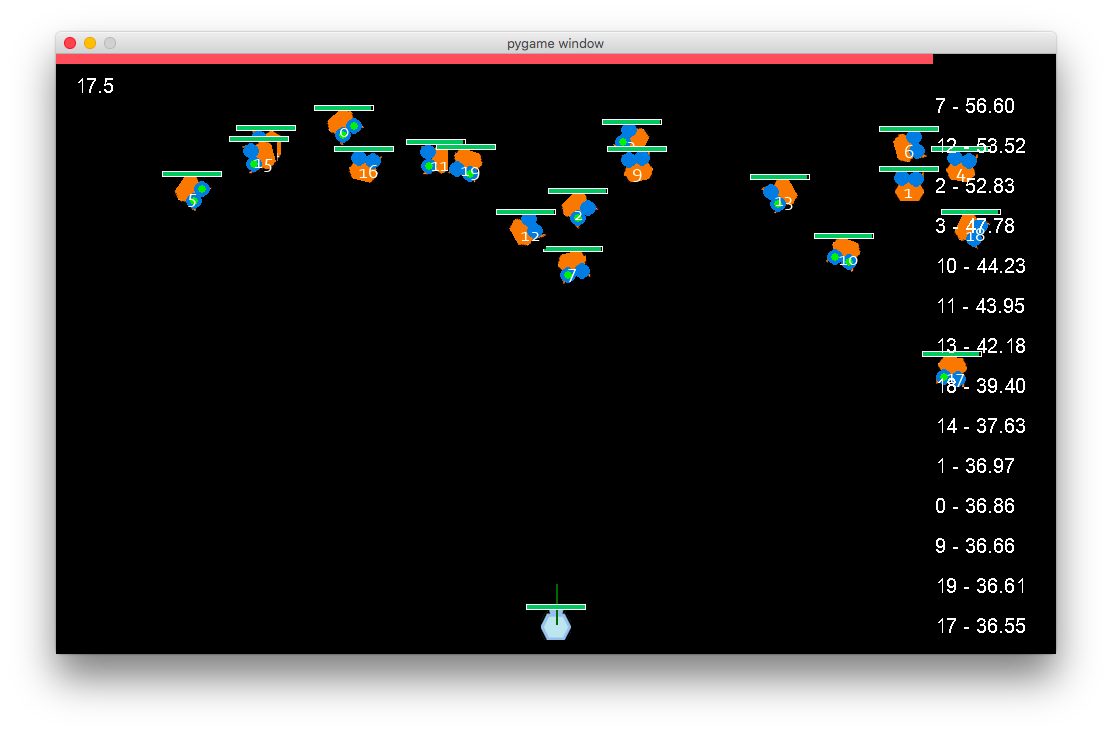
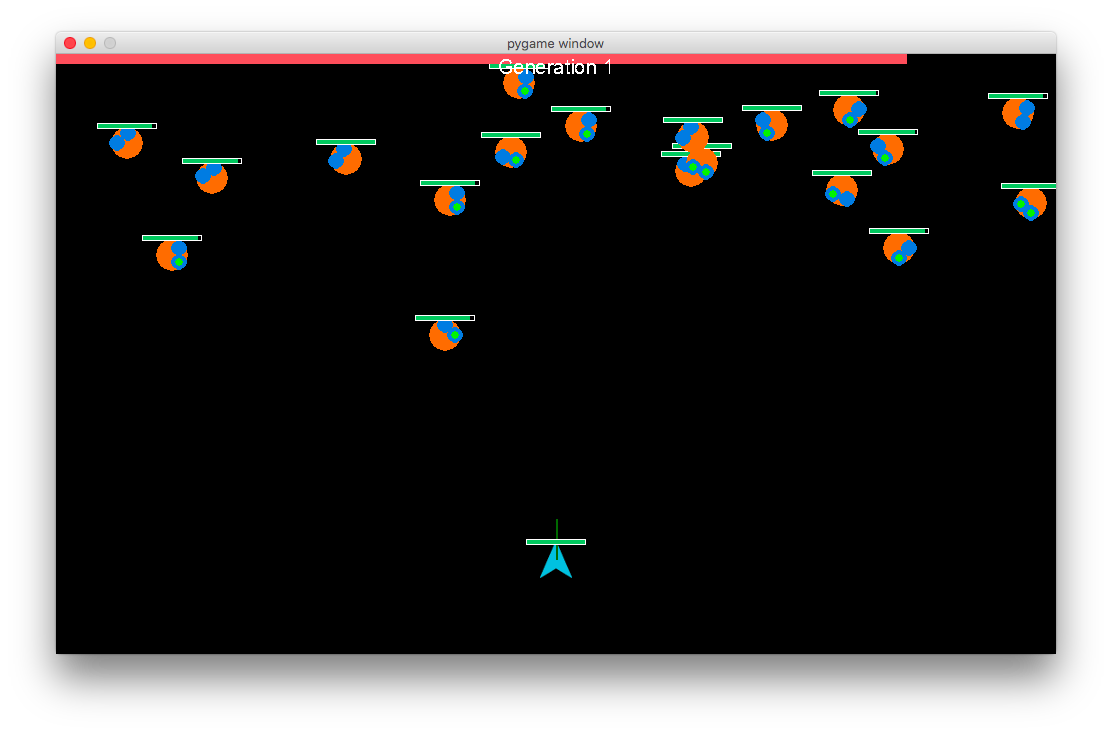
todo:

music

storyline

graphics

I experimented with the layers of the neural network and realised it was way too complex for the intended goal of following the player, I removed the hidden layer so that there were only one set of parameters thus making mutation easier.



to add difficulty increase speed and rotational velocity with generation

testing:

faster not smarter

player/creatures too close to wall, creatures kamikaze into wall

start creatures in centre

mutability don’t mutate fit creatures

win: defeat all in time limit

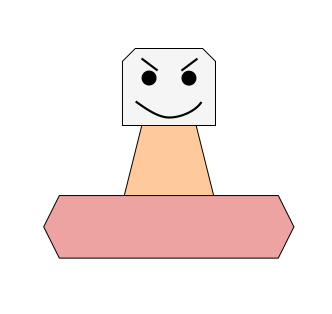
player wins back health if its bullet kills a creature

storyline:

Evil Dr Darwin is trying to breed to ultimate killing creature

Defend yourself by trying to kill them with bullets

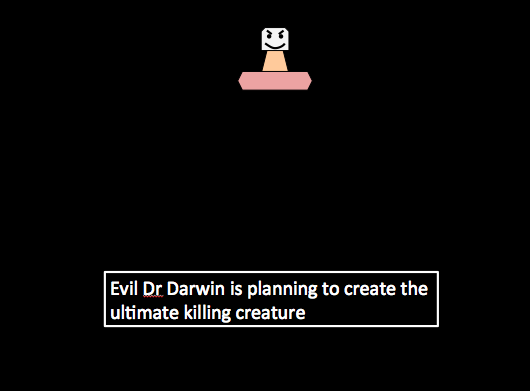
Run them into walls



had to implement scene class to control scenes

add animations like creature dying: creatures shrinks then sort of “pops” like Mario

intro to look like this



change creatures so they make sense