CE810 - Game Design 2

Lab - Searchable Design Spaces

Joseph Walton-Rivers & Piers Williams

Monday, 14 May 2018

University of Essex

Intro

Design Spaces

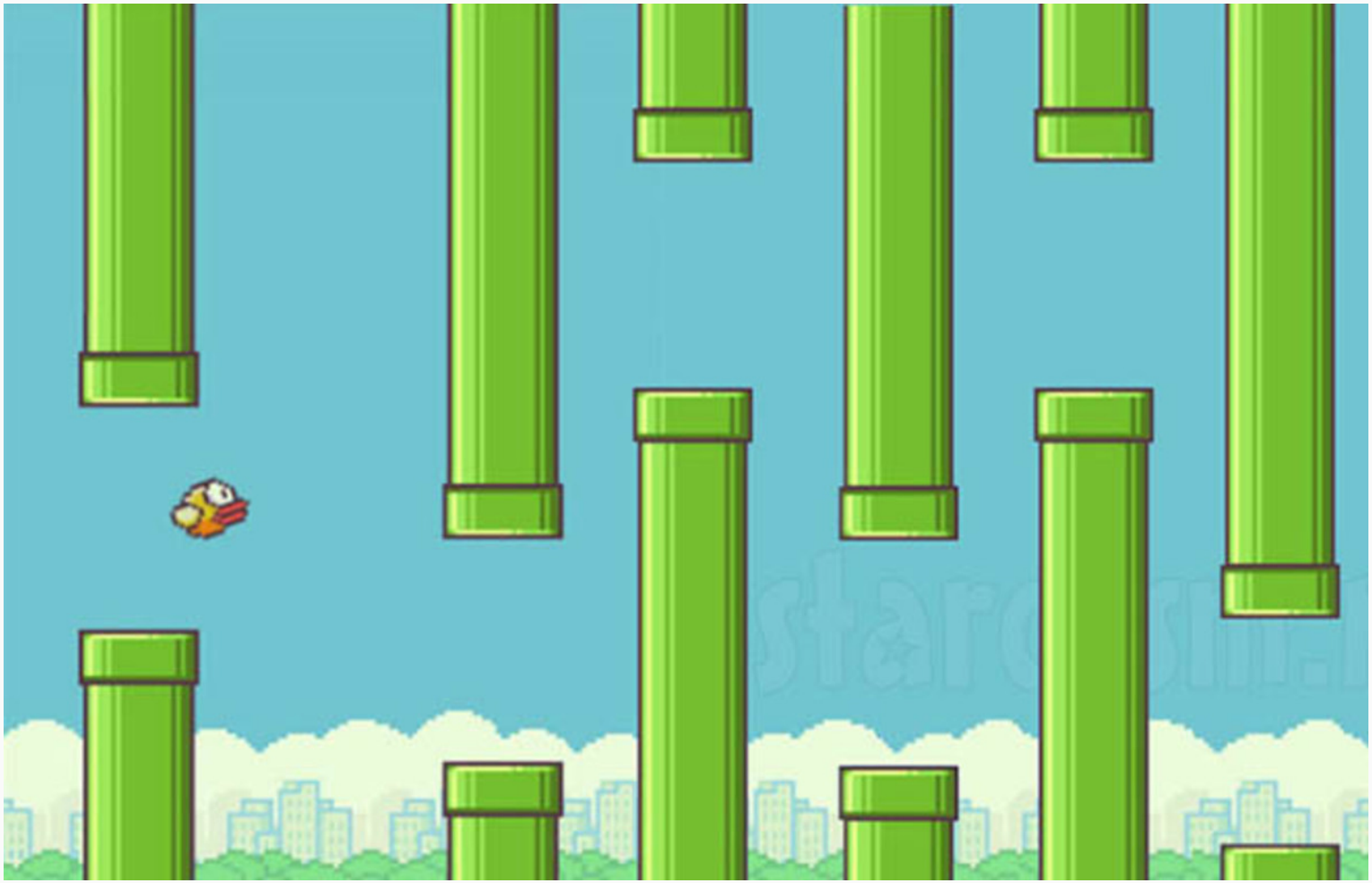
- In this morning's session we talked about Game Parameters
- These are properties which help to define the game
- These are often **dependant** on one another.

Flappy Bird

Exercise: Game Parameters

Question

What game parameters are there for *Flappy Bird*?



Answer: Game Parameters

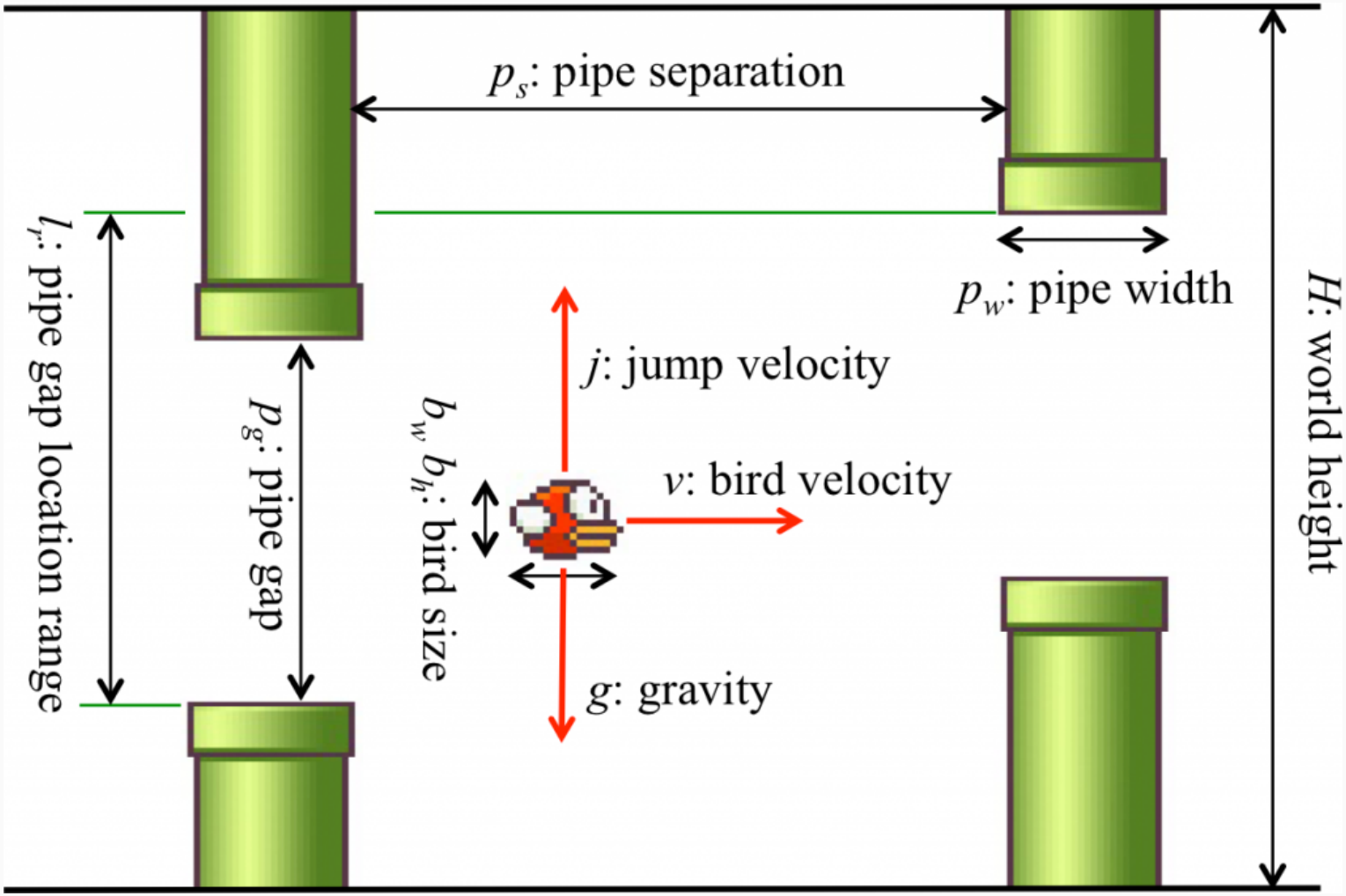


Figure 1: paramters by Isaksen et al @ NYU

Flappy Birds

Activity

Go to the *Flappy Bird* demo and change the sliders.

How does changing the parameters affect the gameplay?

`http://game.engineering.nyu.edu/projects/exploring-game-space/` [1]

How does this relate to us?

Isaksen et al basically did the following:

1. Select parameters
2. Repeat *N* times
    2.1 Generate games
    2.2 Evaluate games
    2.3 Record results
3. Analyse results
4. Output result

This is how we're going to think about tuning our own game parameters.

The Role of AI

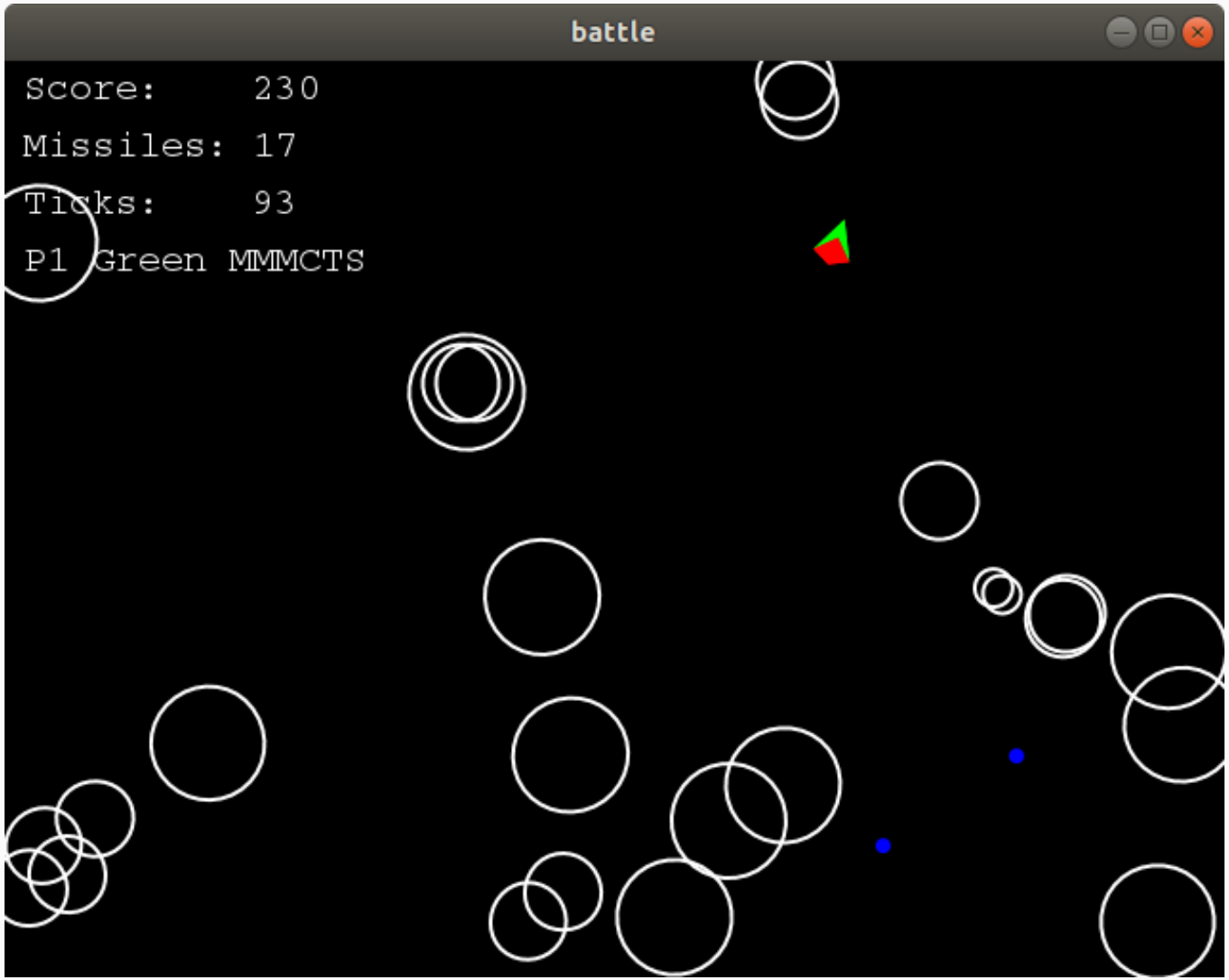
Problem Doing evaluations is time consuming

Solution Make AIs that play them

Asteroids

Asteroids

Lets look at a more complicated example



Parameters

Question

What parameters could we change for *Asteroids*?

- Game Length
- Number of bullets
- Speed of bullets
- Number of asteroids
- Number of asteroid children
- Ship speed
- Ship turn rate
- Bullet cost
- Can bullets hit the ship

Metrics

Question

What can we measure about *Asteroids*?

- Rankings
- Score difference
- Time to win (game ticks)
- Distance travelled

Metrics

Exercise

Create a version of *Asteroids* that **disadvantages** the rotate and shoot player over the other agents.

Asteroids Codebase

Overview

- Relatively simple
- Easy to change parameters
- Can customise it further if you particularly want something else
- Genetic algorithm included to assist you
    - Only thing it needs … a better Fitness Function

Running a Game

```
SimpleBattle battle = new SimpleBattle(true, params);
BattleController p1 = new SingleMCTSPlayer(new Random());
battle.playGame(
  p1,
  new MultiRecorder(scoreRecorder, bulletRecorder)
);
```

0

Parameters?

```
int[] params = new int[N_PARAMS];
Arrays.fill(params, -1);
params[N_MISSILES] = 200;
```

The set of parameters currently supported is in that handy list from slide 9

0

Searching the space

- Searching design spaces can require a lot of computational power.
- We can do better
- Genetic Algorithms can aid us here
- We have implemented a basic searcher using a GA Library
- It’ll do for now
- Head to class “com.fossgalaxy.games.asteroids.battle.jenetics.Jenetics”

Jenetics: Parameters

```
Arrays.fill(USING, false);
USING[N_MISSILES] = true;
USING[BULLET_TIME_TO_LIVE] = true;
USING[SHIP_MAX_SPEED] = true;
USING[SHIP_STEER_RATE] = true;
USING[BULLET_KILL_SHIP] = true;
```

0

Jenetics: Limits

```
int[][] limits = {
  {10, 500}, // N_MISSILES
  {20, 100}, // BULLETT_TIME_TO_LIVE
  {1, 10},
  {5, 50},
  {0, 1}
};
```

0

Jenetics: Chromosomes

```
// Convert limits to the chromosomes
List<Chromosome<IntegerGene» genes = Arrays
  .stream(limits)
  .map(x -> IntegerChromosome.of(x[0], x[1], 1))
  .collect(Collectors.toList());

// Chromosomes to genotype
Factory<Genotype<IntegerGene» genotype =
↪    Genotype.of(genes);
```

0

Jenetics: Fitness

```
int[] params = getParamsFromGenotype(genotype);
AIExperiment experiment = new AIExperiment(5,
 ↪    controllerFunctions, params);
Map<String, List<Integer>> scores = experiment.run();
Map<String, Integer> avg = new HashMap<>();
for(Map.Entry<String, List<Integer>> entry : scores.entrySet()){
avg.put(
entry.getKey(),
entry.getValue().stream().mapToInt(Integer::new).sum() / 5);
}
return avg.get("PiersMCTS") - avg.get("RotateAndShoot");
```

0

Jenetics: Engine

```
ExecutorService exec = Executors.newFixedThreadPool(3);
final Engine<IntegerGene, Double> engine = Engine
  .builder(Jenetics::fitness, genotype)
  .populationSize(50)
  .executor(exec)
  .optimize(Optimize.MAXIMUM)
  .build();
```

0

Jenetics: Running it

```
final Genotype<IntegerGene> result = engine.stream()
  .limit(limit.byExecutionTime(Duration.ofMinutes(120)))
  .limit(300)
  .peek( x-> {
    System.out.println("Generation: " + x.getGeneration());
    System.out.println("Best Fitness: " + x.getBestFitness());
    }
  )
  .collect(EvolutionResult.toBestGenotype());

System.out.println(result);
```

0

References i

A. Isaksen, D. Gopstein, and A. Nealen.
Exploring game space using survival analysis.
In *FDG*, 2015.

D. Perez, S. Lucas, and J. Liu.
Lecture slides for ce810.
2015-2017.