# CSC3003S: Capstone Project 3
# PlainsWalker

Daanyaal du Toit (DTTDAA001)
and
Brett Sharwood (SHRBRE003)

# Abstract

Animated films are the result of a large amount of time, money and effort. Each frame needs to be precisely rendered and elements need to be carefully tracked between screens to ensure continuity in the animation. It is thus extremely difficult and expensive to animate the movement of very large crowds of animals in a realistic and visually pleasing fashion. The aim of the *PlainsWalker* programme is to facilitate this animation by simulating the herding of animals and exporting the known positions and directions to an animation programme.

# Contents

# 1 Programme Specification

## 1.1 Project Requirements

This program models the interaction herd animals and the world around them. It uses the Reynolds[1] flocking model to calculate realistic inter-herd behaviour. The constructed simulation is then exported to the SoftImage animation package to be rendered in 3D.

## 1.2 Extra Features

The herd animals must deal with additional environmental factors such as way-point following and slope, predator and obstacle avoidance.

[1]Reynolds, Craig W. (1987) Flocks, Herds, and Schools: A Distributed Behavioral Model. *Computer Graphics*, 12(4), 25-34

# 2 User Manual

## 2.1 Prerequisites

### 2.1.1 Java

For the *PlainsWalker* programme to run, Java SE 6 must be installed on the computer. Instructions on downloading and installing Java can be found on Oracle's website.

### 2.1.2 Python

In order for the simulation to be imported into SoftImage, python 2.7 must be installed. Instructions on downloading and installing python can be found on the python website.

### 2.1.3 SoftImage

To render a 3D version of the simulation, AutoDesk SoftImage is needed. Instructions of downloading and installing SoftImage can be found on the SoftImage website. Bear in mind that SoftImage is proprietary and, after a 30-day free trial, a product key must be bought to continue using the software.

### 2.1.4 Gear

Gear is an add-on for SoftImage, which is needed for the exported simulation. You can download Gear here, and the zip download contains instructions on linking the required modules to python and installing the add-on to SoftImage.

## 2.2 Installing *PlainsWalker*

*PlainsWalker* is available as an executable JAR file. It also comes with Windows and Linux batch files.

### 2.2.1 Running From Command Line/Terminal

Windows:

```
C:\> cd <path to PlainsWalker.jar>
C:\<path to PlainsWalker.jar>\> java -jar PlainsWalker.jar
```

Linux:

```
~$ cd <path to PlainsWalker.jar>
<path to PlainsWalker.jar>$ java -jar PlainsWalker.jar
```

### 2.2.2 Using the Batch File

A batch file is essentially a saved script which performs the above task. The benefits are that it is much simpler to use, particularly for users who are not familiar with command line interface, and it precludes the need to type out the same commands continually.

The batch file is configured to run the jar from within the same directory; double clicking on the batch file should run the programme. If the batch file is moved - for instance to the desktop for easy access - the path must be changed. To do this, simply right-click on the file and open it in the text-editor of your choice. Then add the line

```
cd <path to PlainsWalker.jar>
```

at the top of the file.

## 2.3 Requirements for a Simulation

To run a simulation, you need a starting *heightmap* file. This specifies the landscape on which the simulation will take place and should be in the following format:

```
[width] [length] [list of height values]
```

*width* and *length* should be integers which define the size of the 2D array, and list of height values should be $width * length$ space-separated doubles. These numbers should all be on one line.

## 2.4 Starting a Simulation

Once the *PlainsWalker* programme is open, the first step is to create a new simulation, or load an existing one.

### 2.4.1 New Simulation

To start a simulation, go to "File > New Simulation". A dialogue will appear; navigate to where the desired heightmap is saved and select "Open".

### 2.4.2 Saving a Simulation

To save an existing simulation, go to "File > Save Simulation". A dialogue will appear; navigate to the desired location for saving, choose a filename to save as and select "Save".

### 2.4.3 Loading a Saved Simulation

To load a previously saved simulation, go to "File > Load Simulation". A dialogue will appear; navigate to where the previous simulation is saved and select "Open".

## 2.5 Adding Herd Animals, Predators and Waypoints

### 2.5.1 Modes

To start placing objects on the landscape, select one of the buttons on the toolbar.

| Button Name | Meaning |
|---|---|
| H | Herd Animal |
| W | Waypoint |
| P | Predator |
| O | Obstruction |

This will put you in the specified mode and any clicks on the landscape will place an element of that type.

### 2.5.2 Groups

Once in a mode, press a number (0 - 9) to select which *group* to assign the next placed element to. For instance, "2"+click in **H** mode will assign a herd animal to herd 2.

### 2.5.3 Placing

Placing an object is as simple as clicking on the landscape. However, do this *carefully* as the functionality of editing and deleting placed objects has not yet been added.

## 2.6 Running the Simulation

To start the simulation, press "Play". The simulation can be paused at any point, or "stopped", which will skip to the end. Restarting the simulation has not yet been implemented.

## 2.7 Exporting to SoftImage

To export to SoftImage, click the "Export" button after the simulation has ended. A dialogue will appear; choose a location and title for the animation file. Then, run "export.py" in the *scripts* directory and give it the location and name of the animation file. This will generate a piece of code called "import.py" in the *scripts* directory. In *SoftImage*, click on the script icon (roughly the middle of the screen, right at the bottom) and open "import.py". Running the script should result in the animation correctly configuring.[1]

[1] Actual zebra modelling turns out to be rather difficult and as such has not yet been implemented. For now, it is appropriate for you to see a selection of herding spheres.

# 3 Conceptual Overview of *PlainsWalker*

# 4 Programme Design

– USE CASE DIAGRAMS

- – CLASS DIAGRAMS
- – DATA STRUCTURES USED
- – SIGNIFICANT METHODS IN EACH CLASS
- – SPECIAL TECHNIQUES?

# 5 Validation and Verification

Things about Testing and How We Did It.

# 6 Discussion of the Results

## 6.1 Areas for Improvement

The *PlainsWalker* simulation meets most of its goals – visibly, much of the required functionality is, at least in part, available. However, there is a lot that requires improvement and many ways that we did not achieve our goals.

### 6.1.1 Efficiency

Our simulation slows down drastically for large numbers of herd members, which makes it frustrating to use and in some cases may cause it to crash as it exceeds the memory space assigned to the JVM. Despite our efforts to streamline the programme, it is still very slow and could probably be severely improved by some clever techniques that we didn't think of or manage to research in time.

### 6.1.2 Functionality

There are several core functions that we didn't manage to include. These largely revolve around the editing of placed objects, such as moving them between assigned groups, deleting them and changing their starting positions.

### 6.1.3 Usability

The usability is the section we most feel we could have improved on, having had several ideas that we didn't manage to finish in time.
Specifically, we wanted to improve the user interface and add a significant amount of visual feedback to recognise the user's action. We had several ideas, including changing the cursor image and window background colour depending on the insertion mode (e.g. herd insertion vs. predator insertion).

### 6.1.4 Linking

Although python scripts and an export script were written, we would have preferred to have created a nicer and more user-friendly link between *PlainsWalker*

and *SoftImage*.

# 7 Conclusion