Platform for Investigating Predator/Prey Chase Scenarios

Cade Breedlove (working with Dr. Nick Long)
Stephen F. Austin State University

Abstract

In this project, we worked to develop a platform for investigating different pursuit strategies involving a variety of predator-pray situations. We took the approach of making the simulation turn based which would allow for using either input from human users or algorithmic strategies. Other variations we sought to look at include:

- Having multiple predator/prey
- Varying the relative speeds and turning characteristics of the predator/prey
- Adding obstacles of varying sizes

This platform would allow for exploration of multiple strategy types to be explored to examine how sensitive a "winning" strategy in terms of the characteristics of the pursuit.

Introduction

This project focuses on creating computer software that can track the paths taken by a predator and a prey with different speeds and turning characteristics. A turn based version of a sample chase is shown in Figure 1. Because there are so many decisions to be made in a chase scenario, strategies are usually evaluated over many runs to examine the efficacy of different approaches. This kind of simulation has a wide range of applications in areas like missile defense, asymmetric warfare, or collision scenarios with autonomous vehicles.

We limited our initial work to examine the basic dynamics of a chase scenario with one predator and one prey with a random initial configuration, but built all elements of the software to be adaptable for adjusting the number of predators or prey, the relative speed of the predators and prey, and the turning characteristics. Generally speaking, the predator will have a higher top speed (which will allow for a greater distance traveled in each turn), where as the prey will be able to turn much faster. The simulation is a competition of speed versus agility.

Because we use a turn based strategy for our simulations, two essential limiting elements for both the predator and prey are the maximum distance that can be traveled in a single step and the minimum turning radius of each participant, both of which can be checked efficiently with computers.

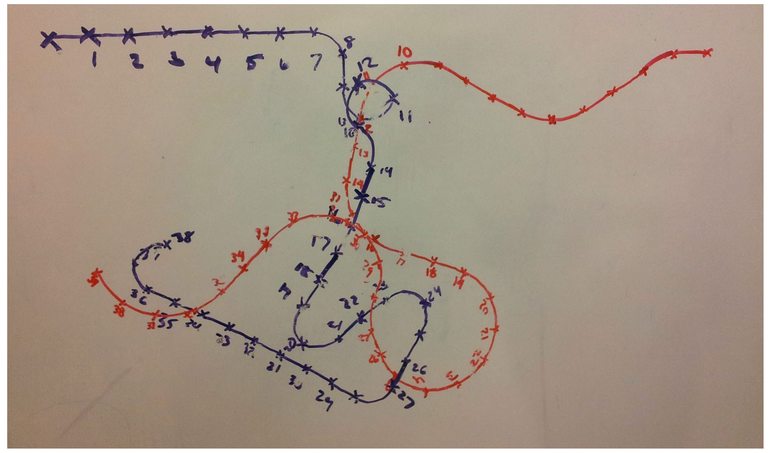

Figure 1. Pursuit Diagram

Development Tools and Tasks

Unity is the main development tool that was used to create the software for our chase scenarios. Unity is a powerful development tool that allows user to create 2D and 3D games, simulations, and interactive experiences and build for many different types of computer hardware. Unity is an industry standard platform with an internationally diverse set of users in game design and software development. Additionally, Unity has many built in tools, like user input tools or algorithmic movement strategies, that can greatly reduce the work needed to build a like ours.

Scripting was utilized throughout the creation of the project. Scripting is a form of coding that provided instructions that are read and executed by another program. This can be seen when the player script is activated by my game manager script to get stored information about the player.

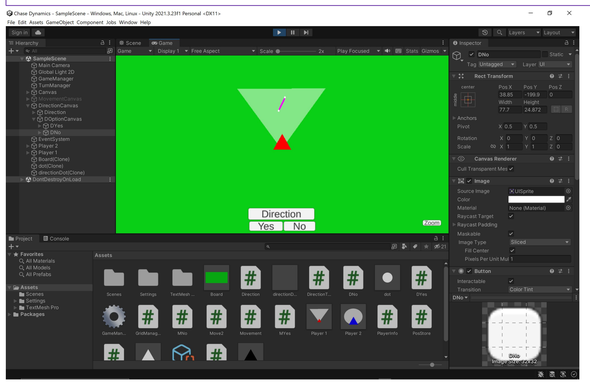

Figure 2. Unity Game Engine

Discussion

Within Unity, I utilized game objects to create the characters that take in user input or utilize algorithmic strategies to take turns. These turns are indicated by a position selection as well as a representation of direction for both the predator and prey. I utilized other game object types to visually represent where the user can select the next location and direction for each step. Data is collected through the use of a collider that can detect a mouse click and be able to pull the x and y positions and convert them if necessary for the appropriate input. There is also a board game object that defines the bounds of our play field where the pursuit strategies are happening. A UI canvas was utilized to create buttons, this canvas is an object that can easily adjust with the changing of the camera size and is utilized to display some information as well as display the buttons used to progress through a users turn. Scripts are also utilized to move information that is needed to make buttons work as well as moving the users to their next selected spot.

Figure 3 shows a picture of a Bezier Curve which we will use to fill in the path taken based on starting and ending positions and direction of travel. Bezier curves will also allow us to easily check the distance traveled and the minimum turning radius for each step. This also gives us a smooth path that we can use to illustrate the paths of the predator and prey.

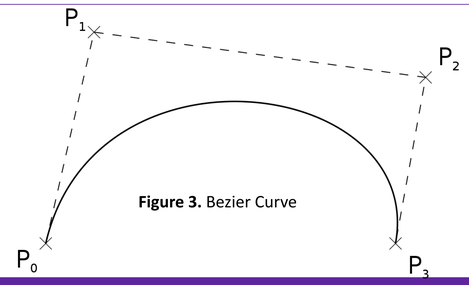

Figure 3. Bezier Curve

Future Work

In the future, we hope to implement algorithmic strategies for predators and prey. These algorithmic strategies will have the ability to have users playing against algorithms well as playing algorithms against each other. Also, the implementation of obstacles with random placement and size will offer more interesting variations. Along with the implementation of the obstacles, we will need to define whether a location is not traversable, and classify how objects hinders the user input in some way.

Contact

Cade Breedlove
Steven F. Austin University
Computer Science
breedlovcm@jacks.sfasu.edu
817-586-9139

References

Unity Game Engine Version 2021.3.23f1
Unity.com