CSC 251: Graphics - Spring 2014:

Assignment 1: Arachnophobia

Drawing in 2D with OpenGL

Intermediate: Saturday, January 11, 11pm; **Final:** Monday, January 20, 11pm.

1 Goal

The goal of this assignment is to get yourselves familiarized with OpenGL as well as to brush up your linear algebra basics.

2 The World

The world is 2D and contains a rectangular base at the near (lower) end, where you operate from. A set of black, red and green spiders are moving towards you from random positions at the far (top) end. You have a laser cannon that can shoot a short laser ray in the direction pointed to by the cannon (see Figure 1). You can move the cannon left or right to any position on the base as long as the space you are moving to is clear. The cannon and baskets do not interfere (they may occupy the same space). The game should display the final score when it gets over.

If a spider lands on your base, it occupies that position on the base and you will not be able to move to that position or over it. If a spider lands on you (the cannon), the game is over. You also have a red and a green basket at the base that can be used capture red and green spiders. Your goal is to capture as many red and green spiders as possible and shoot down as many black spiders as possible.

You should be able to control the baskets and the cannon by using the mouse and the keyboard. Details of the control are given below. Specifications of the world should be read in when you start your program.

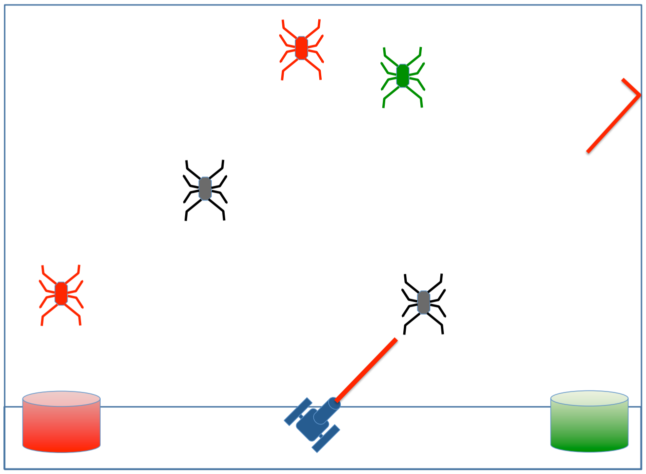

Figure 1: Sample *Spider World* with five spiders and the cannon shooting a black spider. The picture given above is indicative and your models of objects may look different.

2.1 Objects in the World

The following are the properties of each element in the world.

1. The World: The world is rectangular and is specified by its size: $width, height$. There is a base rectangle that occupies the width of the world and with a length (or height) that is 10 percent of the world height. The world does not change once it is created, while all other object listed below may be move.

2. Spiders: There are three kinds of spiders; red, green and black colored. The spiders randomly appear at the top end of the world and move towards the base at a constant speed. Any spider that comes in contact

with a laser ray will be destroyed (disappear) immediately. If a spider reaches the base and it encounters a basket, it enters the basket and disappears. If it reaches the base, the user loses 5 points and it blocks that space. The speed of the spider varies between $minSpeed$ and $maxSpeed$ and is decided randomly when it is created. Once created, the spider moves at this fixed speed. You may also slowly increase the min and max speeds at time progresses.

3. Baskets: There are two baskets at the base: red and green. They are at the left and right ends of the base when the game starts. You can select a red or green basket by pressing r and g buttons respectively and move a selected basket one position to left or right by pressing left or right arrow key. If a red basket captures a red spider or if a green basket captures a green spider, you get one point. If a basket captures a spider of any other color, you lose one point.

4. Cannon: The cannon starts at the middle of the base and it can be selected by the b button. It may be moved left or right by the arrow keys as in the case of the baskets. It may also be turned left or right by the up and down arrow keys. The space bar is used to shoot a laser ray in the direction pointed to by the cannon. If a spider lands on the cannon, the game is over.

5. Laser beam: When you press the space bar, a ray of light of the length, 10 percent of the world height emanates out of the cannon. It moved forward at a high speed, reflects if it hits the sides of the world, and exits out if it reaches the far end of the world. If a laser shot destroys a black spider, you get one point and that laser ray is terminated. Once you shoot a laser ray, the canon takes one second to recharge, and then only you can shoot again.

2.2 Specifying the Objects

When the program starts it should be able to take a file containing the descriptions of the world. It will contain: 1) The dimensions of the world, 2) Difficulty level, which controls the number and speed of spiders and 3) Initial positions of baskets and cannon on the base.

3 Controls

You should be able to select any movable (basket or cannon) object by pressing the r,g and b buttons. Highlight the selected object (in a slightly different color or any other form of highlight). One should be able to move the selected object using left or right arrow keys and rotate the cannon it left or right about its mid-point using up and $down$ keys. You can shoot a laser beam (provided the cannon is fully recharged) by pressing the space bar.

Escape or q button is used to quit the interface. Use the p-button to pause or un-pause the game.

You should also be able to select any movable object by clicking on or near it, and move it left or right by dragging. Clicking the right button on the cannon and dragging will cause the cannon to rotate.

4 Optional

Display a running scoreboard at the top right corner of the world. Another element that might add more interest and difficulty is the presence of moving obstacles that can deflect laser beams. Including audio could make the game more interesting, and so will improving quality of the object model. You may also add any additional item that you feel would make the game more interesting. Make sure that you explicitly list out the additional factors that you have included, their behavior, and how to control it, in your game in the *Readme* file that you upload.

5 Useful Hints

1. While designing the game, make sure that you have the following components, and design them independently: i) A world representation: Create a structure/class that stores all the details of the objects in the world and their current state. ii) An event handler: The event handler is invoked whenever a user interaction (key press, mouse click, drag, etc.) happens or when a time tick happens, the handler should take the appropriate action and modify the world state accordingly. iii) A draw function: For each event or time tick the draw function is invoked, which will draw all the objects in the world using

OpenGL calls. You can further improve the world representation and event handling by adding interesting object properties that follow intuitive physical laws.

2. For the draw functionality, write a few basic functions for the following and use them in drawing/animating your world. The first is a $drawLine$ function to draw a line segment of specified end points, thickness and color. Use this function to define a $drawWorld$ function that would draw the elements according the current state of the world. The drawWorld function could be called every time an object in the world moves.

3. The main control logic (or game engine) will look at the current state (time, collisions, etc.) and update each element in the world. Create a collision detection function that checks for collision between any two objects. This may be used by the game engine to find any impact and take corresponding action.

4. Start with a simple world and complete the game. You may enhance the objects/motion, once you are done with your V1.0. While creating the objects and the game engine, think of how to make each parameter that you set to be flexible.

5. The object classes may contain methods to draw themselves.

6 Submission

The submission will be in two stages. In the first stage (intermediate), you are expected to create the world with simple representations for the objects (colored boxes). The spider boxes should be randomly generated and should move towards the base. You should be able to select and translate the baskets and cannon with the keyboard, and fire a laser beam with the space key.

For the intermediate submission, the following things are not required: i) Mouse control, ii) Rotation of cannon iii) Collision detection and deflection of laser from the sides, iv) Detailed models for objects such as spiders, cannon and basket. These are required only for the final submission. Please note that what is mentioned here is minimum requirement, and you are encouraged to do more by the intermediate deadline.

Your submissions should include your source code, a makefile and a compiled executable. You need to include a *Readme* file that describes any additional information that is needed in compiling/executing you code. Do not use any non-standard libraries. In addition to these, include a file named help.txt or help.pdf (no word or other proprietary formats) in the submission that gives a one page description of the game and how to play it in case you implement any additional features that is not mentioned above.

Details of how to submit and any modification to the above submission details will be posted by the TAs or faculty towards the submission deadline.

7 Grading

You will be graded based on the functionality, correctness, quality of interface and cleanliness of coding. Implementing the basic requirements correctly will fetch you 90% of your grade. Remaining 10% will be given based on the improvements that you do over the basic game. In addition, submissions that are found to be exceptional by the graders, will be showcased, and will be awarded extra credits up to 10%.

The intermediate submission will carry 30% of the grade, and the final will carry the remaining part.