Faculty of Science

Course:	CSCI 3010U: Modelling and Simulation
Component:	Assignment #2
Topic:	Car Following Simulation and 3D Plotting in WebGL

Collaboration Policy

These assignments contribute significantly toward your mark. They are designed to prepare you for future exams and your project. It is in your own best interests to work on this assignment alone. However, if you are stuck on part of the assignment, collaboration is permitted to some degree. The official policy of the instructor is that it is acceptable to discuss general strategies with your fellow students, but when it comes to actually writing code you should do it entirely alone.

Any student who turns in work copied (in whole or in part) from another student will result in both (or all) students being investigated for a violation of academic integrity, which will result in failure of the assignment, a permanent record of the incident on your academic record, and possible further consequences. You'll also find the next examination to be far too difficult, since you've skipped some of my planned learning objectives.

Overview - Part 1

For the first part of the assignment, you will simulate two cars driving along the x-axis. Modelling a car following another car is useful for analyzing traffic patterns at intersections, and after highway slowdowns. One car's acceleration will be controlled by preset values, entered by the user, and the other car will follow a model developed by General Motors.

Part 1 - Instructions

To perform this simulation, you will model two vehicles position, velocity, and acceleration and apply the following formula:

$$a_{following}(t + \Delta t) = \frac{\alpha}{x_{front}(t) - x_{following}(t)}\left[v_{front}(t) - v_{following}(t)\right]$$

Equation 1 – The acceleration of a car following another car

Term	Description
$a_{following}(t + \Delta t)$	The acceleration of the following car at the current time step
$x_{front}(t)$	The position of the lead car at the previous time step
$x_{following}(t)$	The position of the following car at the previous time step
$v_{front}(t)$	The velocity of the lead car at the previous time step
$v_{following}(t)$	The velocity of the following car at the previous time step
α	Sensitivity constant

Table 1 – Definition of terms in the car following formula (Equation 1)

Write a simulation in Open Source Physics that uses this formula to run a simple simulation of a car following another car. The visualization will be the relative distance between the two vehicles over time.

The modelling of the lead car requires further explanation. The lead car will have an initial acceleration which will last for a certain number of time steps, at which point the lead car will use a new acceleration value. Sample output of this simulation, along with example values for each of these input values are given in the sample output, in figure 1.

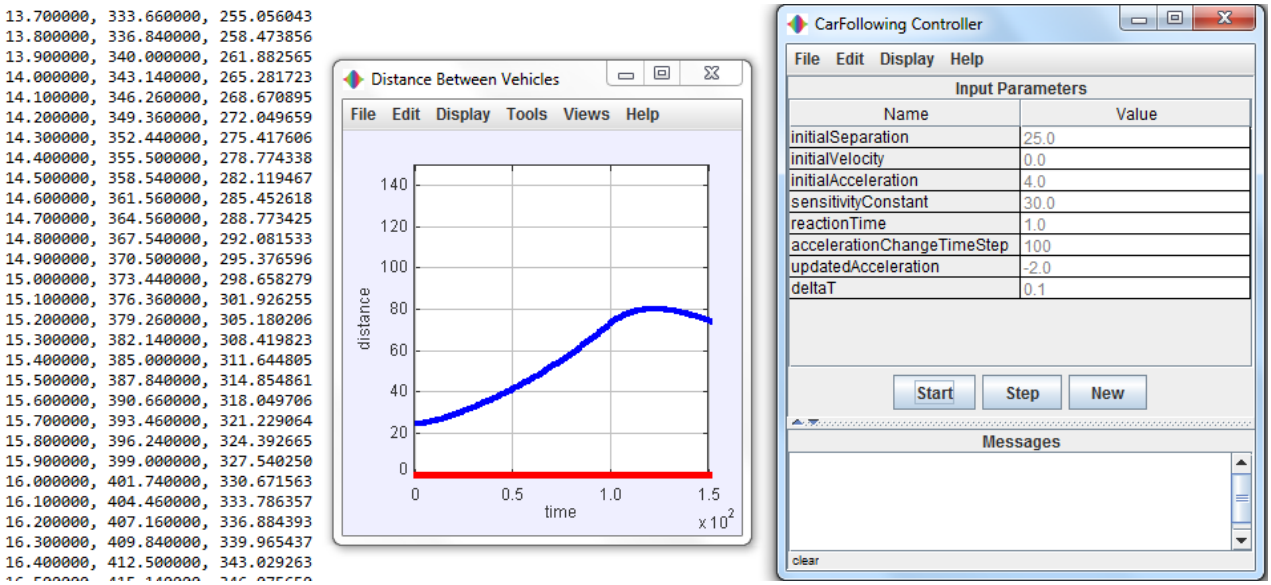

Figure 1 – Sample input values and the resulting visualization

Part 2 - Overview

For this part of the assignment, you will create a simple 3D plot tool in WebGL. The tool will have a simple user interface, which merely allows the user to change the X and Y axes, as well as the size of the triangles that will represent each point, and a checkbox to turn off/on rotation (which lets you examine your plot from different angles, plus it just looks cool).

Part 2 - Instructions

For this part of this lab, you will modify the provided HTML code (which also includes some of the resulting JavaScript) to generate a X,Y grid, calculate the Z value at each grid location according to the formula below, and create a plot which uses WebGL to draw a single triangle at each (x,y,z) location.

$$z = \cos(\sqrt{x^2 + y^2})$$

Equation 2 – The equation for calculating z values

***Note**: It is recommended that you put this z calculation into its own function, to make for easy replacement with other functions during testing.*

Your program will re-generate this grid each time the parameters collected in the HTML file change. These change events already trigger the reset() function. Write the code necessary to generate the X,Y grid, and the Z values, and generate a list of vertices for the triangles from those values. This triangle data will be fed to WebGL to draw the scene.

The triangles at each point are adjustable, via the provided HTML control *pointScale*. Multiply the X and Y values of your triangle by this scale factor when generating your vertices. This base triangle data (given in table 1) will be multiplied by the pointScale, then added to the (x,y,z) for the point being created. The triangles vertices be based off of the following values:

Vertex	X	Y	Z
1	-0.5	-0.5	0.0
2	0.5	-0.5	0.0
3	0.0	0.5	0.0

Table 2 – The base triangle coordinates

The *rotate* component also will affect your plot by turning on/off constant rotation in each of the three axes. When checked, you will increase the rotation angle in each direction by a constant amount (e.g. 0.02 radians). When unchecked, the rotation angle will remain unchanged. Each time the plot is rendered, you will apply a corresponding transformation matrix to your vertices, through the vertex shader. Earlier examples from the lectures do the same thing, albeit in a simpler way.

The colours of each point will depend on the Z value of that point. In the vertex shader, set the colour of each vertex based on the following formulae:

```
R = (z / 2.0) + 0.5
G = 0.0
B = (z/-2.0) + 0.5
A = 1.0
```

Equations 3 – The equations for the colour components of a vertex

You should definitely use the simple-matrix.js file provided. It contains matrix operations that you may find useful. You can use the `generateTransformMatrix()` function to generate a transform matrix,

which you can pass to the vertex shader as a uniform value. One of the animation in-class examples did precisely the same thing, so you could look there for inspiration if you can't figure it out.

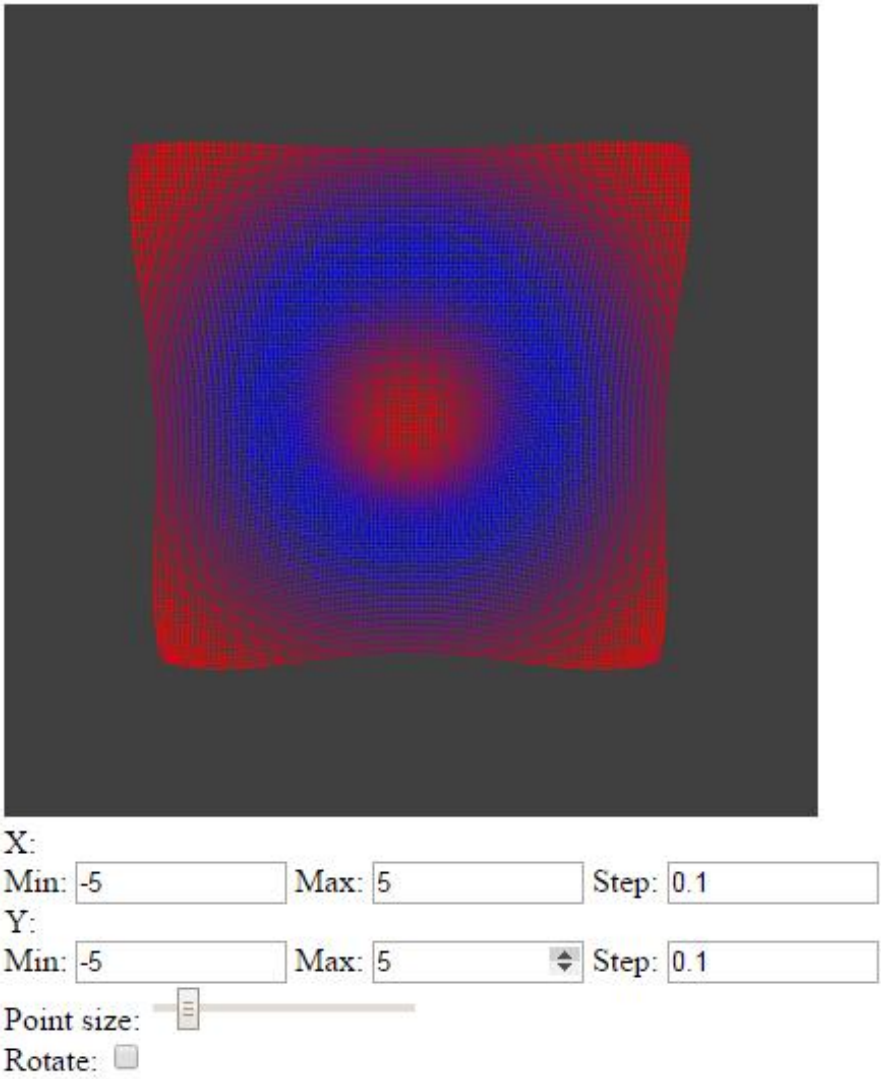

Figure 2 – Some sample input values and the corresponding plot output

How to Submit

Submit a ZIP file (containing all parts of the assignment) to the corresponding drop box on Blackboard. Name your file using the naming convention Lastname_FirstName_100200300_Asmt2.zip (e.g. Fortier_Randy_100539147_Asmt2.zip), and include a comment section at the top of each source code file that contains your full name, your student ID, and a brief description of the assignment.