CS460 Computer Graphics

Assignment 1: Using Texture in OpenGL Drawing. Total points 30
Deadline: 12 Nov, Friday

Description:

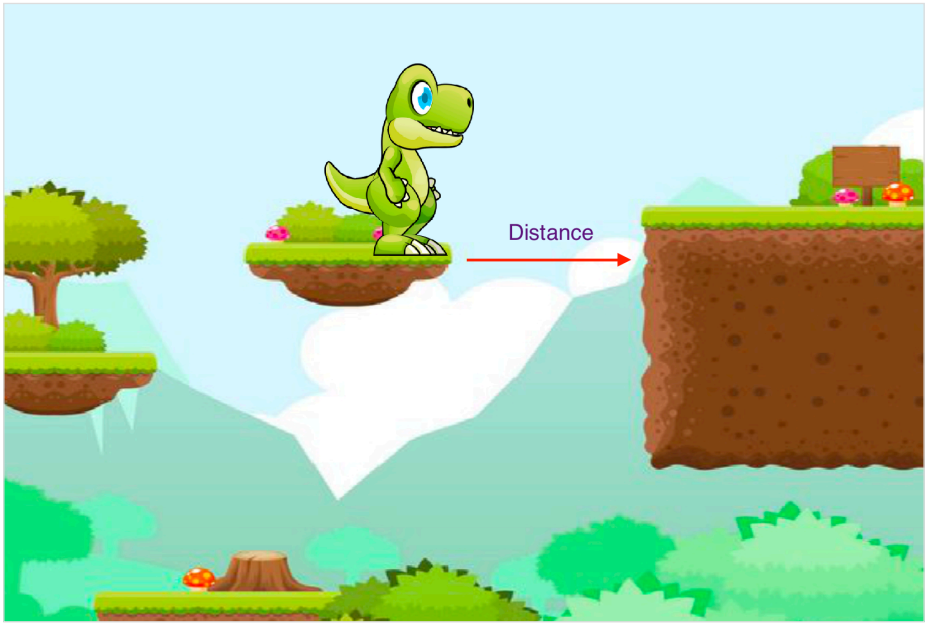
The purpose of this assignment is to use texture on various basic geometric shapes to create a realistic virtual environment. You can use the supplied texture resources or use your own (make sure the dimension of the texture is a power of 2) to create an appealing environment.

Instructions: Drawing a Textured World

The world will be drawn by using textured quads/triangles/circles, etc. The students can combine a number of primitive drawing objects to create complex animated drawing object and place it in the environment. The students should store the texture name, geometry dimension, geometry transformation information in a file. In the initialization section, the program is going to read the geometry/texture/transformation data for a set of drawing objects and add them in the ArrayList variable for rendering purposes. In the render method, there should be no initialization.

A skeleton project, texture resources, sprite texture resources have been included with this assignment. The students can extend the given skeleton project in their assignment implementation. The student are free to choose the dimension of the drawing. However, the drawing canvas must be bigger than 768 pixels width and 768 pixels height and less than 1920 pixels width and 1080 pixels height.

In addition to having static texture geometry objects, the students should attempt to create a number of dynamic animated textured drawing objects [Five animated textured drawing objects would be optimal, including the keyboard controlled SpriteKey object). One obvious choice would be use a GSpriteKey object. One of the animated sprites must be controllable by using keyboard commands in the world.



By using keyboard event control, the GSpriteKey object should journey from point A to point B as shown in the first figure. When the GSpriteKey object has arrived at point B, the program can display a congratulatory message. Further, the program should position the GSpriteKey object at point A so that the process can restart. During the journey, the user can carefully choose Walk, Run, and Jump animation type to move from point A to point B.

When the GSpriteKey object is moving from one object to another drawing object placed in the world collision detection algorithm should be used. As displayed in the previous figure, GSpriteKey object should be able to employ jump animation type to move from one ledge to another ledge drawing objects in the gaming environment. If the user is unable to maneuver the GSpriteKey object properly, the GSpriteKey object will fall below the ledge or in some cases fall on the water. In such cases, the GSpriteKey object should be placed back at the point A location so that the user can restart the journey.

Each drawing element must be element assuming it is drawn at the origin. Afterwards, the students can use any OpenGL translation functions to animate the drawing objects. When creating the scene, use the provided classes instead of starting from scratch. The students can use the provided set of classes from the previous assignment to begin working on their assignment.

In this assignment, each student is given option to create their own texture world and animated textured drawing objects. In this scene, students must add 4 new textured drawing objects. Each textured drawing objects can be composed of a number of primitive drawing objects (Quad, Circle, Oval, Triangle, etc). Please note, these drawing objects should be animated in the scene in a meaningful way. Examples, of the drawing objects that the students can consider can be, plane, helicopter, car, windmill, kite, UFO, rocket, robot, butterfly, ambulance, school-bus, firetruck, etc.

Req Type	Marking
R.x	Assignment gets 0 if any critical submission requirements (shown in red) are not met.
A.x	You lose 2 marks for each good practice requirements (shown in amber) not met.
G.x	You earn 2 marks for each design requirements (green) satisfied and well implemented; 1 mark if it's partly met or met but not well implemented; and 0 if it's not met.

Assignment Restrictions:

Technology Restrictions: You need to use the JOGL library to complete this assignment. You need to submit the java files and the texture files in a zip archive. You are free to use any IDE (Eclipse, etc) to complete the assignment.

The students must submit three screenshots or a short video (10 seconds) in order to demonstrate the animation in the drawing scene of their projects.

Marking: This assignment is based on 4 design requirements numbered **G.1**...**G.4** for a total of 30 points. Points are awarded, or deducted, based on itemized requirements as follows:

Submission and Good Programming Practice Requirements

The following requirements pertain to all your assignments regardless of what your application is supposed to do (i.e. regardless of the design requirements). These requirements are to ensure that your code is usable, readable, and maintainable.

R.0 UNIQUENESS REQUIREMENT: The solution and code you submit MUST be unique. That is, it cannot be a copy of, or be too similar to, someone else's code, or other code found elsewhere. You are, however, free to use any code posted on our course website as part of our assignment solution. [Assigment mark =0 if this requirement is not met.]

R.1 CODE SUBMISSION ORGANIZATION AND COMPILATION: You should submit all the code files, texture files, and a data file (withe location, transformation, etc property for the geometry objects) necessary to compile and run your app. The professor will execute your java files by setting up an eclipse project. If you compress your submission on Blackboard, you must use only .zip format (not .rar or .jar or others). However, students are allowed to write code on Windows, Linux, or MacOS.

The code should be generic enough to be OS agnostic. Make sure that there is no compiler errors in the submitted project. [Assigment mark =0 if this requirement is not met.]

R.2 README FILE: Your submission MUST include a README.txt file telling the professor how to setup and run your app. The professor should not have to look into your code to figure out how to create the project by using the submitted Java and texture files. Your README.txt MUST contain the following:

- Your name, student number and email address and if you are working with a partner then their name, student number and email address as well.
- Issues: List any issues that you want the instructor to be aware of when they are evaluating your submission. In particular, tell us what requirements you did not implement or that you know are not working correctly in the submitted code. Here you are giving us your own assessment of your submitted project. [Assigment mark =0 if this requirement is not met.]
- Version: JDK version, OS you tested on your code on.

A.3 VARIABLE AND FUNCTION NAMES: All of your variables and functions should have meaningful names that reflect their purpose. Don't follow the convention common in math courses where they say things like: "let x be the number of customers and let y be the number of products." Instead, call your variables **numberOfCustomers** or **numberOfProducts**. Your program should not have any variables called "x" unless there is a good reason for them to be called "x". (One exception: It's OK to call simple for-loop counters i, j and k etc. when the context is clear and VERY localized.) [Minus 5 marks from assignment if this requirement is not met.]

A.4 COMMENTS: Comments in your code must coincide with what the code actually does. It is a very common bug to modify code and forget to modify the comments and so you end up with comments that say one thing and code that actually does another. By the way, try not to over-comment your code but instead choose good variable names and function names that make the code more self-commenting. Don't be afraid to create local variables so that the variable name provides more clarity. [Minus 5 marks from assignment if this requirement is not met.]

A.5 CITATION REQUIREMENT: If you use code from other sources you should cite the source in comments that appear with the code. If the source is an Internet website then put the URL in the comments. You may use bits of code from outside sources but this may not form the complete solution you are handing in. You DON'T have to cite demo code we provide on the course web site or with tutorials and assignments, however that code should not be used for things you post publicly (like on GitHub). [Minus 5 marks from assignment if this requirement is not met.]

VERY IMPORTANT: Any sample code fragments provided may have bugs (although none are put intentionally). You must be prepared to find errors in the requirements and sample code. Please report errors so they can be fixed and an assignment revision posted.

Here are the core requirements of this assignment.

G.1: TEXTURED DRAWING: All the drawing objects should be modified so that they are drawn by using texture and with respect to the origin. We can then use transformation (including translation, rotation, and scaling) in order to position the drawing in the drawing canvas. The initialization of the drawing objects must be separated from the rendering algorithm.[6 points]

G.2: FIVE NEW DRAWING ELEMENTS: Students must create 5 textured animated drawing objects. You can reuse all the previously created drawing objects, however, they must be textured properly. Each drawing objects can be composed of a number of primitive drawing objects and may involve sufficient complexity. These textured drawing objects should be animated in the scene in a meaningful way. All animations must be carried out by using transformation (translate, scale, rotate, etc). Examples, of the new textured drawing objects that the students can consider can be, plane, helicopter, car, windmill, kite, UFO, rocket, robot, butterfly, ambulance, school-bus, firetruck, etc. In the assignment submission, three pictures or a video should be attached to demonstrate the animation of the drawing objects in the scene. [10 points]

G.3: SPRITE CONTROL: When the GSpriteKey object is moving from one object to another drawing object placed in the world collision detection algorithm should be used. As displayed in the previous figure, GSpriteKey object should be able to employ jump animation type to move from one ledge to another ledge drawing objects in the gaming environment. If the user is unable to maneuver the GSpriteKey object properly, the GSpriteKey object will fall below the ledge or in some cases fall on the water. In such cases, the GSpriteKey object should be placed back at the point A location so that the user can restart the journey. [9 points]

G.4: LOADING DATA FROM FILE: In the project we create the drawing objects to draw a scene. At the initialization we use a number of data to initialize a drawing object.

In the bonus challenge, the students should create a text file and store these data in the file. The project should then read these data to initialize the drawing objects. A programmer/instructor should be able to change the data defined in the text file to set a different location for the drawing objects or different color for the drawing objects. [5 points]

In order to accomplish this task, we need to define certain format and data requirement for the drawing objects. For example, we can consider the following format to store the drawing object data in the text file:

#GL_HOME
10 20 0 // location
home.png // texture file name

#GL_TREE
5 20 0 // location
1.0 1.2 1.0 // scaling
tree.png //texture file name

While reading this text file, the program will find the number symbol followed by the drawing object type name. So, in the first case, the program is going to create a home object. For the first object, we are going to read the location and the color of the the triangle, quad, and windows. We need to specify comment for each values, however, the program is going to discard the comment part and skip to the next line to read the next entry from the file.

For each drawing objects, we may need different types of values and that is fine as long as we use comments to explain them. We also need to use these values to initialize the drawing object in the program. After creating each drawing objects, we can add them to the ArrayList and the ArrayList object will in turn display the drawing objects in the render function.

Please let me know if you have any questions. Thank you.