CS 174A — Introduction to Computer Graphics: Assignment 2

Let your imagination and creativity fly! Due: End of Friday Nov 4th

Weight: 15 % *Maximum points:* 37

Note: The class will screen all the games and animations (by opt-in) and hold a vote on the best ones! The top ones will receive a substantial bonus in the course.

Collaboration: **None**. If you discuss this assignment with others you should submit their names along with the assignment material.
Start working on this assignment early. You will not have time to do a satisfactory job at the last minute.

Write a program that displays an animated scene. Your scene should include a combination of hierarchical objects that move around. Required elements:

- [4 points] Make use of hierarchical objects with at least two levels (e.g., a human arm)
- [4 points] Demonstrate the camera tracking a moving object by overwriting the camera matrix using *lookAt()*.
- [6 points] Design polygonal objects of your own to supplement the existing ones. To specify these shapes you must provide novel positions, normals, and texture coordinates to the graphics card. Inspect the method called init_ buffers() in class "Shape" to see which WebGL calls accomplish that.
  You may extend Shape and re-use its methods to get these points, or do the WebGL calls yourself -- your choice. Put the code for it at the bottom of Custom_Shapes.js*, which you will receive during discussion.
  *Extending classes is not naturally part of javascript but nevertheless we will provide a tutorial called Custom_Shapes.js to show you some easier examples that do so, besides the difficult Patch example that powered your Assignment 1 shapes.
- [2 points] Show discontinuous edges for at least one of your custom polygonal objects. Light it with the provided Phong reflection model to create flat shading. Explain in your README where to look for this.
- [2 points] Assign reasonable texture coordinates to, and texture, one of your custom polygonal objects... either procedurally (which involves designing custom shaders) or by mapping an image.
- [2 points] Real-time speed. Make sure that your animation runs at the same speed regardless of the machine your program runs on; i.e., one simulated second corresponds roughly to one real second. The variable animation_time inside the graphicsState object is a good measure of the passage of real seconds.
- [2 points] Display the frame rate of your program on the graphics window, by growing the debug_screen_strings array that gets passed to your Animation each frame.
- [2 points] Make and submit a movie of your animation (length 90 sec or less) using your favorite screen recording application (e.g., camstudio/quicktime). If your program is interactive, submit a video of it being used. You can add subtitles. Make sure you encode your movie to be below the CCLE limit of 100MB, and observe the 90s limit.
- [4 points] Creativity (story, aesthetic style, etc).
- [4 points] Complexity. Are the underlying mechanics that make it work impressive?
- [5 points] Overall quality: Fluidity of object and camera motion, attention to detail in scene construction and texturing, etc.

Special instructions:

- Your video must be only of what your program executable can output given certain user inputs. The video should not be edited.
- Note that creativity and quality amount to 9 points. You will not get a perfect score if your scene is complex, but not creative.
- You must use only the provided template code for making graphics calls; but, you can modify it as you see fit.
- You must do the assignment from scratch. Using outside help or any piece of code from any previous offerings of the course will be considered plagiarism.
- You can see examples of animations made for previous offerings of this course at: http://web.cs.ucla.edu/~dt/courses/CS174A/animations/assignment2-best-16s/

Submission guidelines:

- Submit your movie with the name *<uid_yes/no>.ext (e.g., 802870392_yes.mp4)*, where *<uid>* denotes your 9 digit bruin ID, *ext* can be any common video format, and the *<yes or no>* indicates whether or not we should include it in the competition & viewing.
- Submit all the files related to your project in a single archive named *<uid_yes/no>.zip (e.g., 802870392_yes.zip)*. Please keep the name "index.html", as well as the same folder structure as the template. Here the *<yes or no>* indicates whether or not we should host your work on the page above as an example for future quarters' students.
- Include in the top level of your *<uid>.zip* archive a *README.TXT* file that summarizes your animation, identifies the hierarchical, polygonal, and texture mapped objects, and explains anything else that might be helpful to know in grading and understanding your project. Make it presentable -- if you opt-in to letting your work be displayed, it will accompany your work.