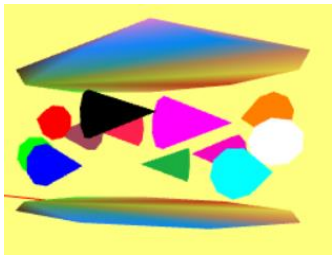
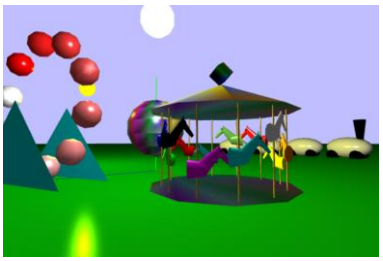
CS 425 – Computer Graphics I – Spring 2023
Homework Assignment 2 – Second Draft
An Amusement Park in Three Dimensions

Due: **Wednesday March 1st, by 11:00 a.m. via Blackboard.**






Overall Assignment:

Model an amusement park in 3D using WebGL, subject to a few requirements. This assignment is designed to cover (at least) the following skills:

- Calculating vertex coordinates for simple primitives in 3D.
- Drawing the same object multiple times in different positions, orientations, and with different scaling factors, from the same set of vertices.
- Moving objects over time, including rotations not centered at the origin and compound motions. (E.g. the wheels on a car turn relative to the car, while the car as a whole, including the wheels, moves through the world. Or robotic arms such as a "spider" ride.)
- Adjusting the viewpoint interactively, to observe the scene from different angles.

Details

- The simulation must include some form of merry go round, not centered at the origin, which turns as a unit and with "horses" that move up and down relative to the merry go round as it turns.
    - This feature must employ compound motion. The movement of the horses must be only relative to the merry-go-round itself, i.e. to place horses on the edges of the merry-go-round and to make them go up and down. The rotation (and placement) of the merry-go-round must also move the horses with it.
- High resolution objects are NOT required for this simulation. Adequate results can be achieved with a simple cube, cone, cylinder, and possibly sphere if they are scaled appropriately. See below regarding required classes.

Required Classes

- TruncatedCone
    - The truncated cone should sit upright, with its base at the origin and centered symmetrically about the positive Y axis. It should have a height of 1.0 and a

diameter at the base of 1.0. The diameter at the top should be "ratio" times the diameter at the base. Ratio = 0.0 yields a true cone; ratio = 1.0 yields a cylinder. Ratio may not be negative.

    - The constructor for this class must calculate the vertex positions needed for the truncated cone and push them to a GPU data buffer. Necessary arguments to the constructor include the ratio of top diameter to bottom diameter and the number of sectors to use around the truncated cone. Other arguments such as desired color may also be passed to the constructor as needed.

- MerryGoRound
    - The MerryGoRound class constructor will create a number of "horse" objects and place them as needed about the perimeter of the merry go round. For the base assignment these may be simple truncated cones. There should also be some additional structure, such as a top and a bottom.
- Horse (Optional Enhancement)
    - The Horse class, if written, would construct a more realistic looking merry go round horse than a simple cone, comprised of a combination of truncated cones and at least one other basic primitive.
- A "main" Javascript file will also be needed, to create one or more instances of the MerryGoRound class and to place it/them where desired in the world. A ground plane and a set of axes at the origin will be helpful for orientation purposes.

User Interaction

- The program must provide for user interaction through the use of keyboard commands.
    - W, A, S, D to move forwards, rotate left, move backwards, and rotate right respectively.
    - These rotations should be about the camera Y axis. The forwards and backwards motions should adjust the X, Z coordinates of the camera along the projection of the camera viewpoint in the XZ plane, while maintaining a constant Y viewpoint height. Pitch and roll should not be affected by these operations.
    - You may want to incorporate additional commands to adjust pitch, zoom, etc.
    - 'R' to reset the camera viewpoint to the original position.
    - 'Q' to quit the program.
    - 'H', 'h', '?' or any undefined key should bring up a help list of available commands.
    - Unless there is a reason to do otherwise, upper and lower case should function identically. I.e. 'D' and 'd' should normally perform the same task.
        - One thought is to have the shifted versions of A and D slide the viewpoint right or left, while the unshifted a and d rotate the view as described above.

Variations and Other Issues:

- The basic assignment does not require the implementation of lighting models, which means that solid color models may not have a good 3D appearance. This is acceptable but try to pick shapes and colors that work as well as possible given the circumstances. (The second and third images shown above are from a later assignment incorporating lighting and shading. In the meantime, coloring vertices of an object with varying colors helps to show the shapes.)

Web Page Issues:

- The web page that you turn is serves as the documentation for your code. It must display at a minimum your name and NetID (e.g. aStudent42, not your UIN number.) It should also describe what the program does and how it works and document any special features to note of your program. Follow any additional instructions the TA provides regarding required documentation. (All files must include name and NetID as comments at the beginning of files.)

Optional Enhancements:

It is the policy of this course that students may add extra enhancements above and beyond the basic assignment. It is not necessary to do these enhancements to get full credit, and the enhancements can never raise a score above 100%, but they can compensate for deficiencies elsewhere in the program. Here are a few ideas:

- Additional camera controls, e.g. for pitch, roll, or to "slide" the viewpoint side to side.
    - The yaw described above should rotate about an axis parallel to the world Y axis, even if the current camera viewpoint is tilted downward or sideways.
- Implement walking mode (camera Y coordinate fixed at 5' above the ground) versus flying mode (Y kept above the ground but otherwise unrestricted.)
- Use of the mouse to control camera movement.
- Create classes for more complex geometry, such as a soccer ball, surface of revolution, or extrusion along a path.
- Implement lighting models in the shaders, to enhance the 3D appearance. (This will become a requirement for the next assignment.)
    - Demonstrate how different lighting and shading models affect the appearance of objects.
- Apply more complex shading to the objects, such as texture mapping or a faux marble.
- Other enhancements as you may consider. Check with the TA if you have any questions. (Note that enhancements done in previous assignments will not earn credit if repeated.)

What to Hand In:

- Hand in a zip file containing a minimum of 4 files: An HTML file, a JavaScript file containing the main application, and JavaScript files for each class created.. Any additional classes created should each have their own JavaScript files. Additional files for the shaders are allowed if the shader programs are complex, but most people will likely include the shaders in the HTML files.
- Do not include the supporting script files from ../Common. You can assume those will be available in the ../Common directory when your program is run.