COSC363 Report

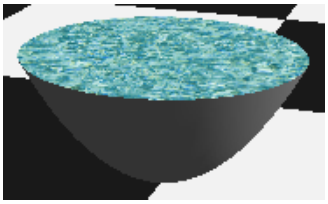
Jack van Heugten Breurkes — 23859472

Control functions

- Arrow keys: up and down to move camera forward and back, left and right to rotate the camera.
- Space bar: trigger jump animation
- ‘x’ key: trigger paper plane crash

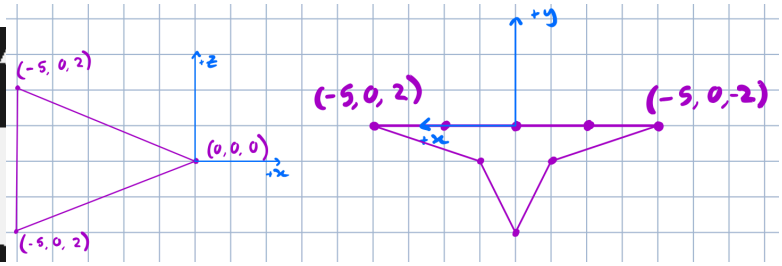
Model 1: Parabolic Pool

The parabolic pool model was created using surface levels drawn using GL_TRIANGLE_STRIP. The shape of the pool is a paraboloid and is calculated using an equation instead of storing the points in arrays.



Model 2: Paper airplane

The paper airplane model was created using a GL_TRIANGLE_FAN for both the side faces and for the back face. The model is animated to fly in circles around the museum using a rotation about the centre of the museum. The plane also moves up and down in a sinusoidal manner as it flies. Pressing the ‘x’ key on the keyboard will cause the plane to crash, activating the particle system to show fire effects.

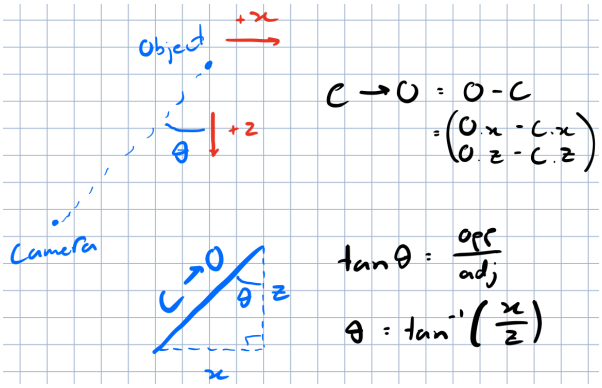


Model 3: Teapot, table and glasses

The teapot, table and glasses exhibit was created using glut built-ins. The table is made up of cubes that have been scaled to form wooden plank shapes. The glasses are made up of cones with stretched cubes for the chutes. The teapot has been animated to tip tea into each of the glasses continuously via a repeating sequence of pitch and yaw rotations.

Extra feature 1: Billboarding

Cylindrical billboarding (billboarding involving rotation on the x and z axes only) has been implemented by taking an object's position in world coordinates and using it to calculate a vector from the camera to the object. This vector is then used to calculate the angle by which to rotate the object so that it is facing the camera.



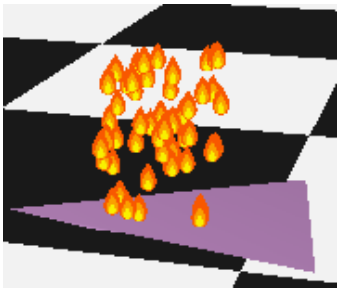
The position of the object in world coordinates has been calculated using the GL_MODELVIEW_MATRIX and the camera position[1]. Using the submatrices M and V of the modelview matrix, the position of an object in world coordinates can be found using the equation objPosWC = camera position + M^T * V where M^T is the transpose of submatrix M.

M			V
a0	a4	a8	a12
a1	a5	a9	a13
a2	a6	a10	a14
a3	a7	a11	a15

Cylindrical billboarding has been used both for the trees displayed around the museum and for the flame effects seen in the particle system.

Extra feature 2: Particle system

The particle system can be seen by causing the plane to crash by pressing 'x' on the keyboard. Particles are generated with a random rise speed, start height, x position and z position. Particles rise based on their speed and return to a height of zero once they reach the max particle height.



Extra feature 3: Physics-based jump animation

A jump animation has been implemented for the camera which can be activated using the space bar. The player's vertical velocity is tracked and used to update the player's y position. After updating the player's vertical height, the vertical velocity is updated using the following equation:

$$\Delta vertialVelocity = -9.8 * \Delta time$$

Where delta time is the time since the last display update in seconds.

Extra feature 4: Skybox

A skybox has been included in the scene. The skybox is created by rendering a large box centred around the camera and displaying textures on each surface. It has been ensured that this box always falls within the camera view frustum.

Extra feature 5: Mathematically generated surface

The bowl shape of the "Parabolic Pool" exhibit is generated mathematically using a series of points falling on the surface of a paraboloid. The points are generated for a certain pair of y values, and then the surfaces of the paraboloid falling between those y values are generated by rotating points about the y axis to create a surface using GL_TRIANGLE_STRIP. The equation for the line that is used as the basis for the surface is as follows:

$$x = \sqrt{y} * 2$$

or

$$y = \frac{x^2}{4}$$

References

1. Formula used to calculate an object's position in world coordinates adapted from section 8 of http://www.lighthouse3d.com/opengl/billboarding/billboardingtut.pdf
2. All textures taken from www.textures.com