Justin Dietrich, Ryan Scott

11/08/2022

CST - 310

Prof. Citro

Project 7

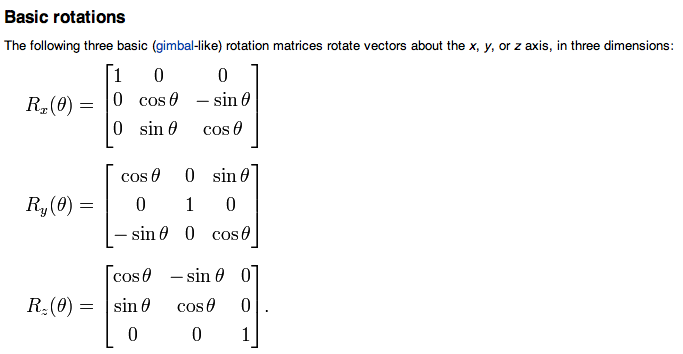
Putting It All Together

**Mathematical Concepts**

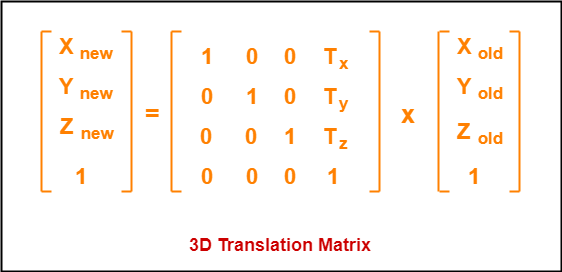
This program utilizes the translation and rotation of shapes, which are defined by matrices. Using functions such as glRotate and glTranslate, the program performs matrix calculations on our behalf that assist in the translation and rotation of shapes.

Below are the formulas that are used in these functions:

**Rotation:**



**Translation:**



**Programming Implementation**

The Triangle program utilizes a global angle and position variable. Each frame, the triangles are rotated and translated in accordance with these variables. To stop a rotation, we use a boolean indicating whether or not the angle variable has changed. Upon pressing the key, these values are incremented.

To accommodate the Cube program, we modified the draw() function in Cube to include some additional parameters. As with the previous program, we use booleans to pause rotation and cube movement according to a global position variable and a global velocity variable. When the cubes reach a certain position, the global velocity variable is flipped.

**Detailed Description of the Method for Creating a Mesh**

The Cube program has a cube mesh that was provided for us by Professor Citro. However, we modified it to take position parameters; this allows the cube to be drawn at any location we specify.

**Description of the method for reading the two codes and processing it**

As we ran the Triangle program, we took note of where the three triangles had been drawn. Under that code, we went ahead and appended both the transformations and rotations into the program to adjust the triangles.

In order to make the Cube program work, we implemented a similar approach from the Triangle program. We looked at where the cubes are drawn in order to add subsequent transformations, as well as two bouncy cubes of varied colors to make the scene complete.

**Technique**

During rotation of the triangles, we made sure to translate the triangles before the rotation occurs in order to maintain the windmill-like effect. This is important because if we failed to do this, the triangles would rotate around the camera origin instead of rotating around the camera position, causing undesirable visual consequences.

**Hotkeys**

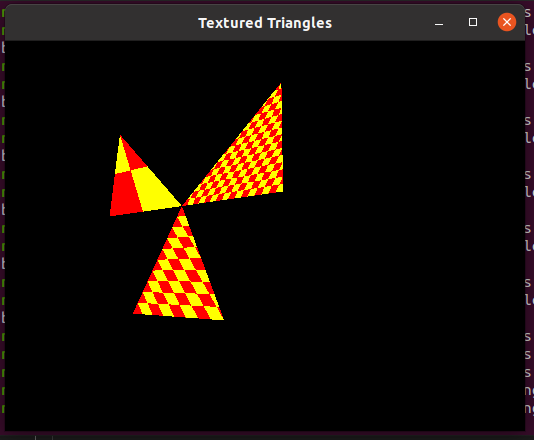
Triangle Hotkeys

| **Command** | stop spin | start spin | move up | move down | zoom in | zoom out |
| --- | --- | --- | --- | --- | --- | --- |
| **Hotkey** | P | C | U | D | + | - |

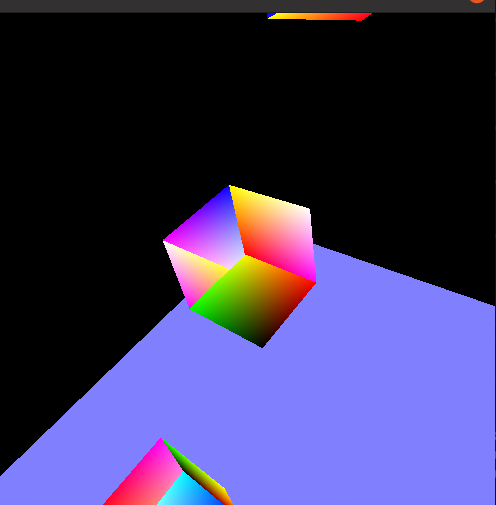
Cube Hotkeys

| **Command** | rotate | stop | continue | move up | move down | zoom in | zoom out |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Hotkey** | R | S | C | U | D | + | - |

**Screenshots**



Spinning triangles are caught on CCTV going *way* above the speed limit!



A rotating cube is photobombed by two opposing bouncing cubes!

**Sources:**

<https://learnopengl.com/Lighting/Basic-Lighting>

<https://www.youtube.com/watch?v=yQvYvv9hy8c>

Citro Padlet