**Project 8 Part 3 Rotating a platonic solid with OpenCV (perspective)**

Name: Vivian Feng Period: 5 Date: 5/17/2023

Did you name your file l083.cpp (Lower case L, then 083)? Yes

Does your file compile & run on terminals? Yes

Did you use a rotation matrix? Yes

Did you start from the coordinates I provided for the cube?? Yes

Describe here in words all the transformations you applied to vertices, for each describe how you implemented it in your code (by multiplying with a matrix, what was the matrix, or by adding a matrix, what was that matrix… be specific):

For the cube, each frame was generated by:

* Scale by 150: multiply by matrix
* Rotate some amount in along the x axis by angle x:
* Rotate some amount along the y-axis by angle y:
* The transformation matrix was obtained by multiplying these matrices together, and then the transformation matrix was applied to the coordinates by multiplying the transformation matrices with the coordinate matrix

The same transformations apply to the tetrahedron, except the scaling factor was 250

Describe in words the rotation you did:

* I performed a rotation about the x axis and the y axis. Each frame increments the angle of rotation about the x axis by one degree, and the angle of rotation about the y axis by one degree. The angle ranges from 0 to 360 for x, and 0 to 360 degrees for y.

Did you use homogenous coordinates? Yes

(that allows you to combine all transformations into one matrix)

Did you combine all those transformations into one single matrix? No

If you used only one transformation matrix, what was it?

What functions/methods from OpenCV did you use?

* Mat class to represent matrices and store video frames
* \* Matrix multiplication operation in OpenCV
* at<double>(I,j) method in Mat class – used to access an element in the matrix
* circle() – used to draw the point in the image for each frame
* line() – used to connect edges
* cols – Mat class attribute used to access the number of columns in a matrix
* VideoWriter – class that creates object used to write to video file
* VideoWriter::fourcc() – class method used to obtain codec number
* open() – method of VideoWriter used to open video file buffer
* << - overloaded operator used to write frame to VideoWriter object
* release() – method for closing VideoWriter object

What functions/methods from OpenCV did you experiment with but ended not using?

* All methods I experimented with, I ended up using

Did you do a perspective rendering? Yes

**Default values used (in case no parameters are given)**

What is the position of the eye you used? (50.5, 651, 349)

What is the plane of the screen you projected on? Plane is defined by *a = (10, 301, 23),* and n = (1,9,1)

Did you name your video rotation.avi? Yes

What functions/methods from OpenCV did you use?

* at<double>(i,j) method of Mat object— access element in a matrix
* -= operator for element in matrix
* matrix.rows — get the number of rows in a matrix (Mat object)
* matrix.cols — get the number of columns in a matrix (Mat object)
* matrix.dot(Mat other) – takes dot product of matrix, is a method of the Mat object matrix
* solve() – used to find the coordinates (u,v) for w1 and w2

What functions/methods from OpenCV did you experiment with but ended not using?

* matrix.convertTo(), which I initially used before finding out that OpenCV overloads \* for scalar multiplication

Obs.: feel free to rotate any platonic solid, around any line, and you may put the position of the screen/viewing window in any place as long as the rotating platonic solid can be seen reasonably.