**Project 8 Part 2 Rotating a platonic solid with OpenCV (perspective)**

Name: Aditya Vasantharao Period: 3 Date: 5/4/22

Did you name your file l082.cpp (Lower case L, then 082)? 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):

I applied 3 rotation transformations (one on each axis), the perspective projection transformation, and then a scale and translation transformation.

First, for each coordinate, I multiplied each resulting coordinate by the following matrix:

1 0 0 0

0 cos(theta) -sin(theta) 0

0 sin(theta) cos(theta) 0

0 0 0 1

Which rotates the coordinates about the x axis by angle theta.

After this, for each coordinate, I multiplied each resulting coordinate by the following matrix:

cos(theta) 0 sin(theta) 0

0 1 0 0

-sin(theta) 0 cos(theta) 0

0 0 0 1

Which rotates the coordinates about the y axis by angle theta.

After this, for each coordinate, I multiplied each resulting coordinate by the following matrix:

cos(theta) -sin(theta) 0 0

sin(theta) cos(theta) 0 0

0 0 1 0

0 0 0 1

Which rotates the coordinates about the z axis by angle theta.

The thetas for each of the 3 rotation transformations are iterated from 0 to 2pi in increments of pi/150, creating 300 frames.

After this, I applied the perspective projection transformation as such:

For each original unscaled point (x,y,z)—so each point is small and close in magnitude to 1—the final, transformed point = (u,v,1), where

u = y \* t

v = z \* t,

where t = (plane - eye)/(x - eye).

After this, I multiplied each coordinate by the following matrix to scale it:

x 0 0 0

0 y 0 0

0 0 z 0

0 0 0 1

Where x, y, z were all equal to 50, and these were the factors to scale the coordinates by.

After this, for each coordinate, I multiplied each resulting coordinate by the following matrix to translate it:

1 0 0 x

0 1 0 y

0 0 1 z

0 0 0 1

Where x=400, y=300, and z = 0. This translates the figure to the middle of the 800x600 canvas.

Describe in words the rotation you did:

I rotated about all three axes: x, y, and z. The way I rotated it using homogenous matrices is outlined above.

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, VideoWriter, Mat::begin<double>(), VideoWriter::open(), Size, Point, circle(), line()

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

None

Did you do a perspective rendering? Yes

What is the position of the eye you used? (12, 0, 0)

What is the plane of the screen you projected on? x = 5

Did you name your video rotation.avi? Yes

What functions/methods from OpenCV did you use?

Mat, VideoWriter, Mat::begin<double>(), VideoWriter::open(), Size, Point

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

None

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.