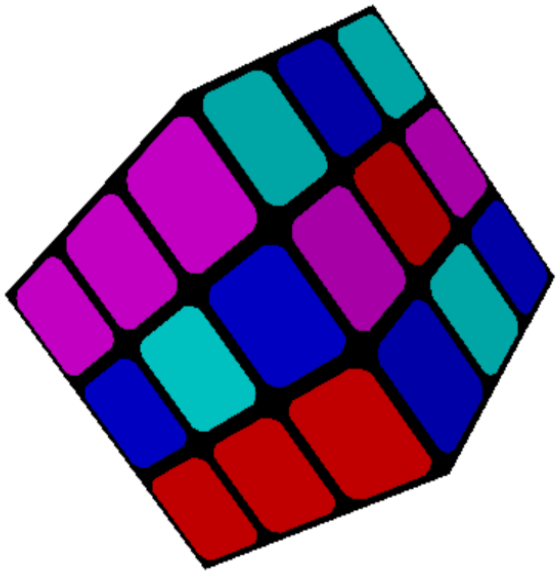
Assignment 3 – Rubik's Cube



In this assignment you will render a dynamic Rubik's cube.

Part 1 - building the cube

1. Use the `AddShape(Cube, -1, TRIANGLES) to` render a cube. See what happen when you translate and rotate the cube.
2. For building the Rubik's cube you will need to duplicate the cube 26 or 27 times and use translate to move each cube to its initial position. Rotate the cube to see if you did it properly.
3. Change key_callback function in order to rotate the Rubik's cube around x and y axis of the scene using the arrows keys.
4. Add mouse callbacks for rotating and translating the whole cube.
5. Change the Fragment Shader so each side of the cube will get its own unique color.

Part 2 – data structures

1. Understand Movable data structure. This data structure will save the transformations you did for each object.
2. Build data structure for single cube and give each cube an index. You will need to know which of the cube you need to rotate each time you rotate a wall of the Rubik's cube.

Part 3 – rotations in a row (the difficult part)

1. Add to callback_key function:
    a. 'R' press state for right wall rotation (90 degrees clockwise).

b. 'L' press state for right wall rotation (90 degrees clockwise).

c. 'U' press state for up wall rotation (90 degrees clockwise).

d. 'D' press state for down wall rotation (90 degrees clockwise).

e. 'B' press state for back wall rotation (90 degrees clockwise).

f. 'F' press state for front wall rotation (90 degrees clockwise).

g. ' ' press state for flipping rotation direction (from clockwise to counter clockwise or vise versa).

h. 'Z' press state: dividing rotation angle by 2;

i. 'A' press state: multiply rotation angle by 2 (until maximum of 180);

2. Mouse callback:

   a. Moving while holding left button will rotate the cube: Left to right movement will rotate the cube around global Y axis and up to down movement will rotate it around global X axis.

   b. Moving while holding right button will move the camera up, down, left or right.

   c. Scroll up and down will move the camera along Z axis (back and forward).

3. Try to implement one wall rotation in the Rubik's cube.

4. Plan where each cube supposes to be after every rotation and how you can follow each cube (hint: use index array).

5. Try to implement few rotations in a row according to your plan.

6. 5 points bonus for implementing 2x2, 4x4, 5x5 Rubik's cube.

7. 10 points bonus for implementation of random mixer and efficient solver. You must be able to display the mixer and the solver action on the screen and write mixer and solver steps sequence to files mixer.txt and solver.txt.

8. **You may check the executable file to compare to your results.**

Part 4 – Submission

1. If you do one of the bonuses or both add readme file with explanation.

2. Zip your project (or files you add and change) and rename it to <id1>_<id2>.zip