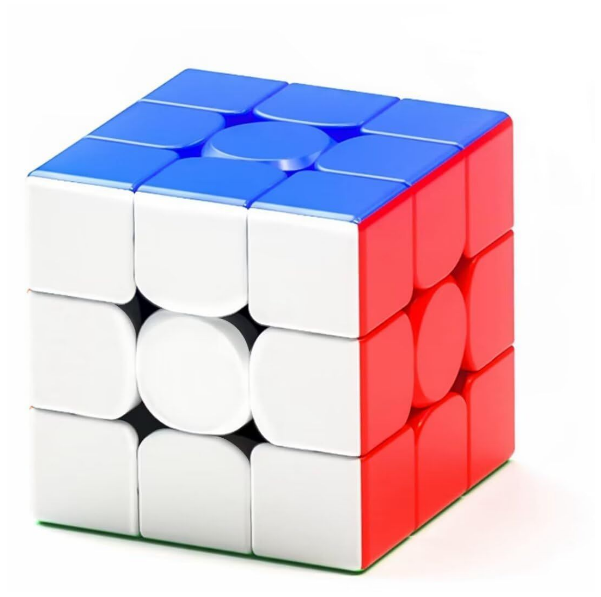
Rubik's Cube Animation

CS4204 Final Project
Jiyuan Zhang, Taijia Liang

3x3 Rubik’s Cube Rotation Animation



Function

1. Rendering pipeline
    a. Transform - set_axis_rotation()
    b. Mesh
    c. Camera
    d. Renderer - light issue

Function

2. Animation manager
    a. Location
    b. Angle
    c. Frame rate
3. Save and combine frames

Cube manager

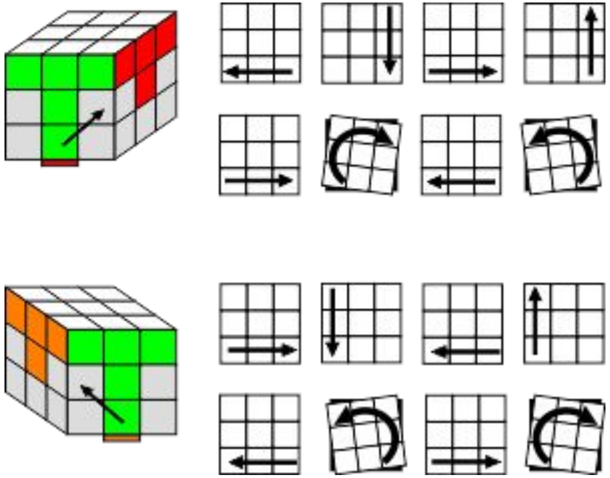
create_piece_mesh(stl_path, position, colors) and setup_cube_pieces(self, stl_path): Little cube mesh object-position, color.

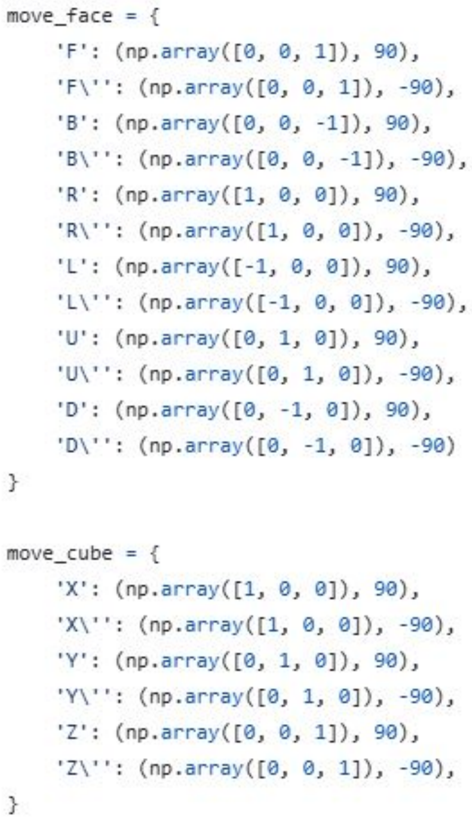
rotate_cube(self, axis, angle): rotate 9 cube objects with the axis.

Cube manager

Rubik cube formula:

- Front
- Back
- Right
- Left
- Up
- Down



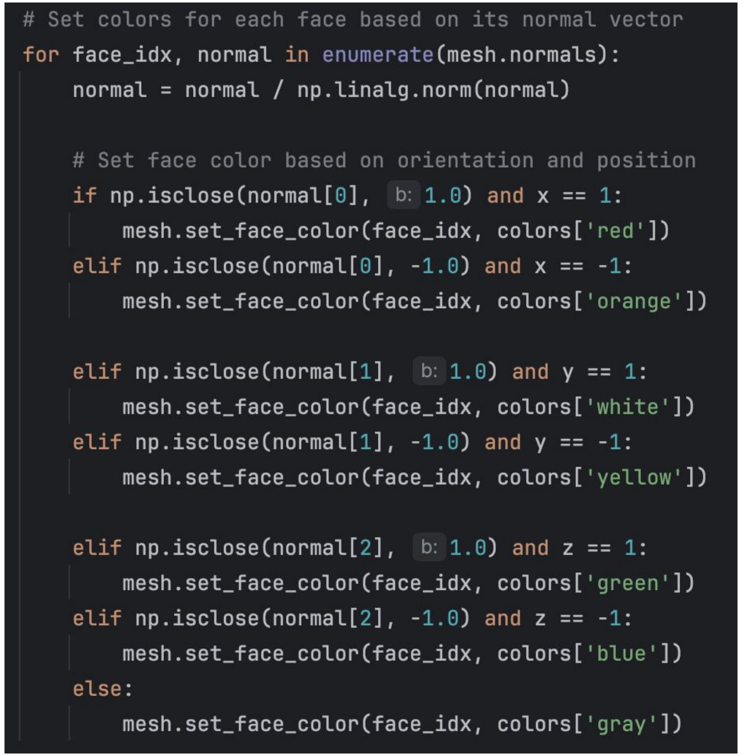


```
move_face = {
    'F': (np.array([0, 0, 1]), 90),
    'F\'': (np.array([0, 0, 1]), -90),
    'B': (np.array([0, 0, -1]), 90),
    'B\'': (np.array([0, 0, -1]), -90),
    'R': (np.array([1, 0, 0]), 90),
    'R\'': (np.array([1, 0, 0]), -90),
    'L': (np.array([-1, 0, 0]), 90),
    'L\'': (np.array([-1, 0, 0]), -90),
    'U': (np.array([0, 1, 0]), 90),
    'U\'': (np.array([0, 1, 0]), -90),
    'D': (np.array([0, -1, 0]), 90),
    'D\'': (np.array([0, -1, 0]), -90)
}

move_cube = {
    'X': (np.array([1, 0, 0]), 90),
    'X\'': (np.array([1, 0, 0]), -90),
    'Y': (np.array([0, 1, 0]), 90),
    'Y\'': (np.array([0, 1, 0]), -90),
    'Z': (np.array([0, 0, 1]), 90),
    'Z\'': (np.array([0, 0, 1]), -90),
}
```

Mesh

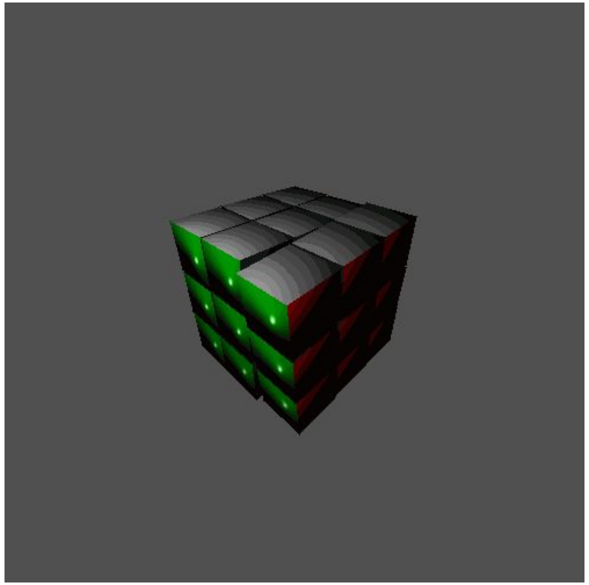
Add color attribute inside the Mesh: face_color

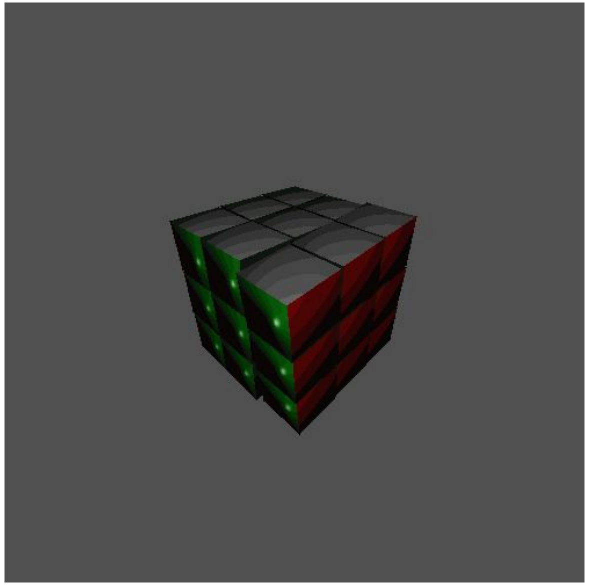
Set the color based on face's normal vector

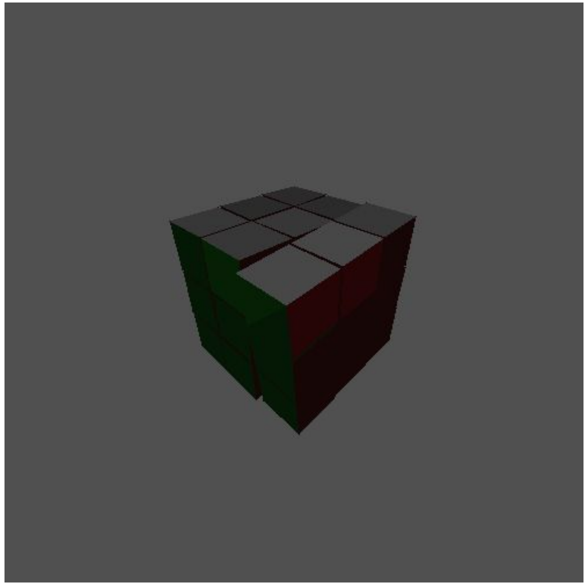


```
# Set colors for each face based on its normal vector
for face_idx, normal in enumerate(mesh.normals):
    normal = normal / np.linalg.norm(normal)

    # Set face color based on orientation and position
    if np.isclose(normal[0], b: 1.0) and x == 1:
        mesh.set_face_color(face_idx, colors['red'])
    elif np.isclose(normal[0], -1.0) and x == -1:
        mesh.set_face_color(face_idx, colors['orange'])

    elif np.isclose(normal[1], b: 1.0) and y == 1:
        mesh.set_face_color(face_idx, colors['white'])
    elif np.isclose(normal[1], -1.0) and y == -1:
        mesh.set_face_color(face_idx, colors['yellow'])

    elif np.isclose(normal[2], b: 1.0) and z == 1:
        mesh.set_face_color(face_idx, colors['green'])
    elif np.isclose(normal[2], -1.0) and z == -1:
        mesh.set_face_color(face_idx, colors['blue'])
    else:
        mesh.set_face_color(face_idx, colors['gray'])
```

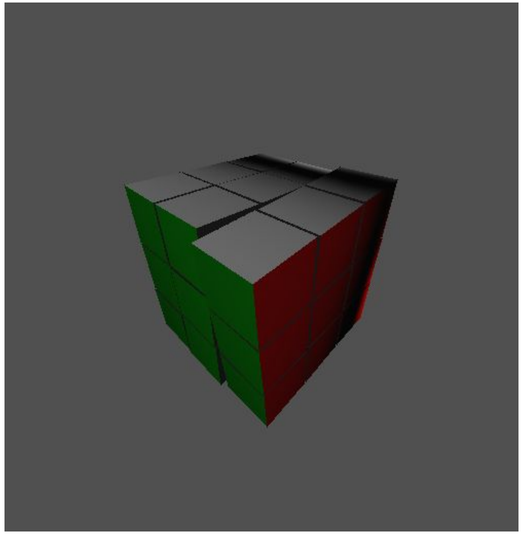
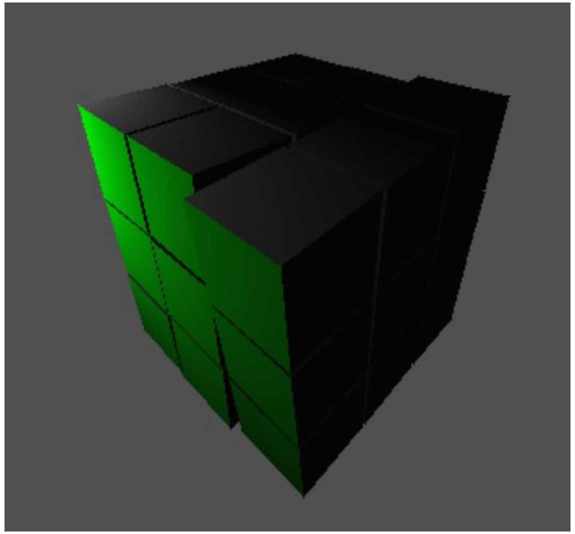
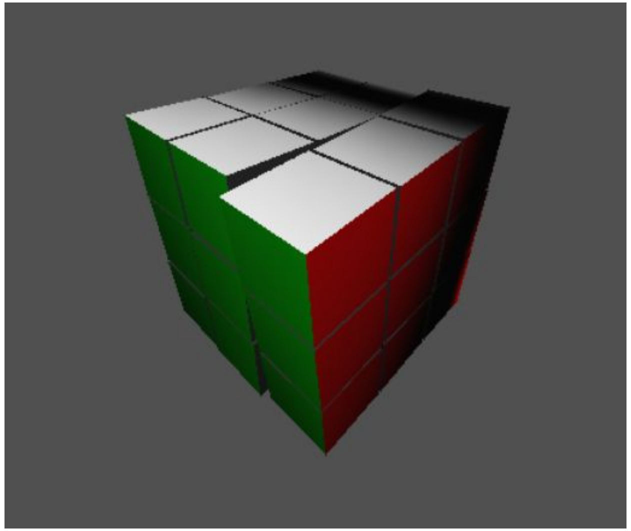
Lighting issue



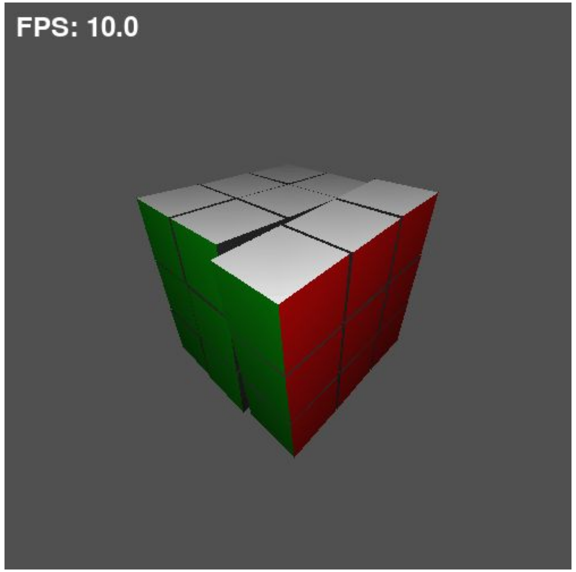


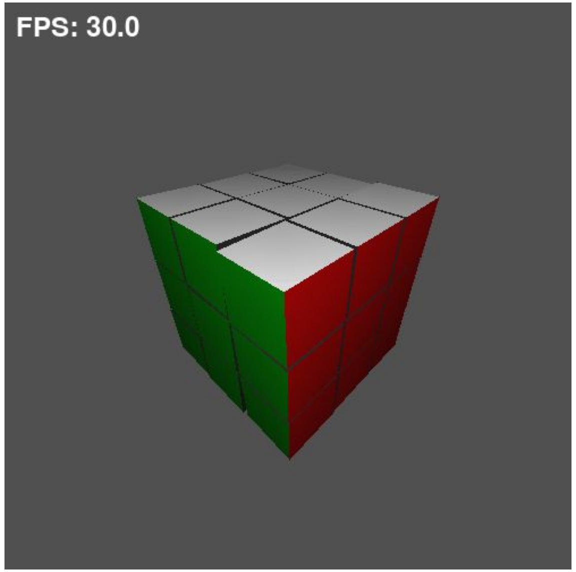


Lighting issue

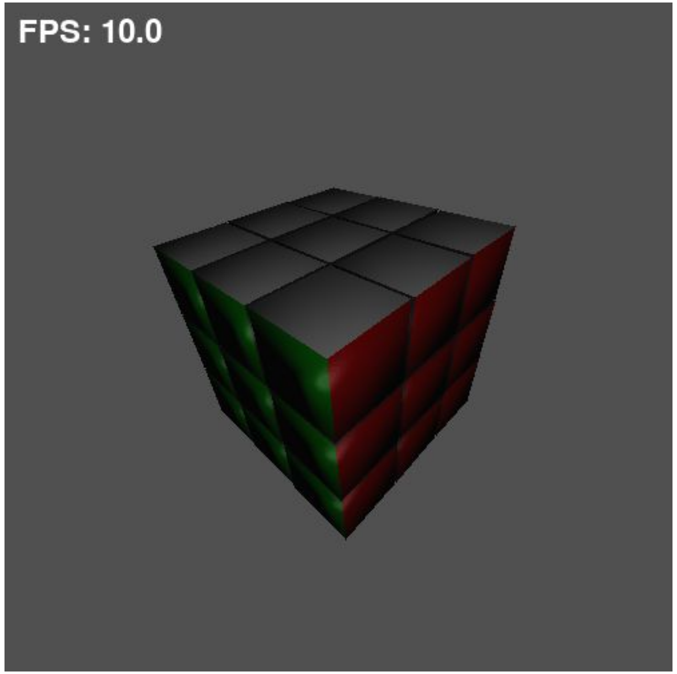




Flat Shading





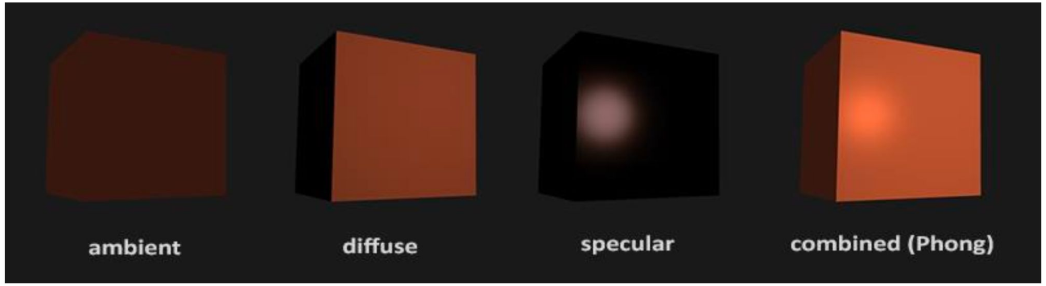
Phong shading



Updated Phong shading

Face normal —> Diffuse lighting computation

Vertex normal —> Specular highlight



$$I_f = \sum A + \sum D + \sum S$$