Exp.No: 05 Vishalini.R

Date: 11-09-25 231701062

**3D Transformations on Basic Objects**

**Aim:**

To write a program that performs 3D transformations (Translation, Scaling, Rotation) on basic 3D objects (Cube, Pyramid) and displays the results.

**Procedure:**

1. Start the program.
2. Define vertices of the 3D object (Cube).
3. Represent transformations as matrices:

* Translation**:** T = [ 1 0 0 tx

0 1 0 ty  
 0 0 1 tz  
 0 0 0 1 ]

* Scaling**:** S = [ sx 0 0 0   
   0 sy 0 0  
   0 0 sz 0  
   0 0 0 1]
* Rotation (X/Y/Z): Use standard 3D rotation matrices.

1. Multiply object vertices by transformation matrix (homogeneous coordinates).
2. Plot original object and transformed object using 3D projection.
3. End the program.

**Code:**

import numpy as np

import matplotlib.pyplot as plt

from mpl\_toolkits.mplot3d.art3d import Poly3DCollection

cube\_vertices = np.array([

[0, 0, 0, 1],

[1, 0, 0, 1],

[1, 1, 0, 1],

[0, 1, 0, 1],

[0, 0, 1, 1],

[1, 0, 1, 1],

[1, 1, 1, 1],

[0, 1, 1, 1]

])

faces = [[0,1,2,3], [4,5,6,7], [0,1,5,4], [2,3,7,6], [0,3,7,4], [1,2,6,5]]

def translation\_matrix(tx, ty, tz):

return np.array([[1,0,0,tx],[0,1,0,ty],[0,0,1,tz],[0,0,0,1]])

def scaling\_matrix(sx, sy, sz):

return np.array([[sx,0,0,0],[0,sy,0,0],[0,0,sz,0],[0,0,0,1]])

def rotation\_matrix\_z(theta):

rad = np.radians(theta)

return np.array([

[np.cos(rad), -np.sin(rad), 0, 0],

[np.sin(rad), np.cos(rad), 0, 0],

[0, 0, 1, 0],

[0, 0, 0, 1]

])

T = translation\_matrix(1, 1, 0)

S = scaling\_matrix(1.5, 1.5, 1.5)

R = rotation\_matrix\_z(45)

transformed\_vertices = cube\_vertices @ S.T @ R.T @ T.T # order: scale -> rotate -> translate

def plot\_cube(ax, vertices, color='skyblue', alpha=0.4):

for face in faces:

square = [vertices[i][:3] for i in face]

ax.add\_collection3d(Poly3DCollection([square], color=color, alpha=alpha, edgecolor='k'))

fig = plt.figure()

ax = fig.add\_subplot(111, projection='3d')

plot\_cube(ax, cube\_vertices, color='blue', alpha=0.2)

plot\_cube(ax, transformed\_vertices, color='orange', alpha=0.6)

ax.set\_xlabel('X')

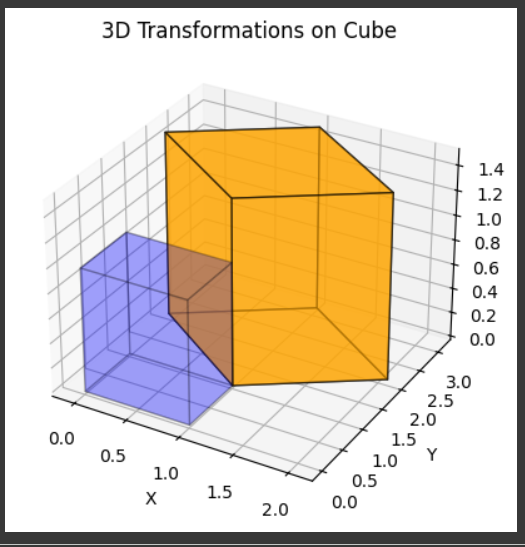
ax.set\_ylabel('Y')

ax.set\_zlabel('Z')

ax.set\_title('3D Transformations on Cube')

plt.show()

**Output:**

****

**Result:**

The program successfully performs 3D transformations (translation, scaling, rotation) on the cube.  
The transformed cube is displayed along with the original cube in a 3D view.