**Lab Report: Advanced Computer Vision - Lab 1**

**Student Name:** Anmol Pandey  
**Student ID:** E22CSEU1069  
**Course:** Advanced Computer Vision  
**Lab Number:** 1

## ****1. Objective****

The objective of this lab session was to understand and implement fundamental geometric transformations in computer vision. Specifically, we performed translation, scaling, and rotation on a 2D object using transformation matrices. This exercise enhances our understanding of how objects can be manipulated in an image processing pipeline.

## ****2. Introduction****

Geometric transformations play a crucial role in computer vision applications such as image registration, object tracking, and 3D modeling. In this lab, we focused on 2D transformations and visualized their effects on a rectangular object. The transformations applied include:

* **Translation**: Moving the object to a different position.
* **Scaling**: Resizing the object.
* **Rotation**: Rotating the object around the origin.

## ****3. Implementation Details****

### ****3.1 Libraries Used****

The following Python libraries were used:

* numpy: For matrix operations and numerical computations.
* matplotlib.pyplot: For visualization.
* cv2: For image processing (though not extensively used in this notebook).

### ****3.2 Code Structure****

The implementation consists of the following key functions:

#### ****3.2.1 Function to Plot Objects****

import matplotlib.pyplot as plt

def plot\_object(vertices, color='blue', label=None):

vertices = np.append(vertices, [vertices[0]], axis=0) # Close the shape

plt.plot(vertices[:, 0], vertices[:, 1], color=color, label=label)

**Explanation:**

* The function takes a set of vertices representing an object.
* It appends the first vertex at the end to close the shape.
* The shape is then plotted using Matplotlib.

#### ****3.2.2 Function to Apply Transformations****

def transform\_object(vertices, transformation\_matrix):

ones = np.ones((vertices.shape[0], 1))

homogeneous\_vertices = np.hstack([vertices, ones]) # Convert to homogeneous coordinates

transformed\_vertices = homogeneous\_vertices @ transformation\_matrix.T

return transformed\_vertices[:, :2]

**Explanation:**

* Converts input coordinates into homogeneous coordinates.
* Applies the transformation matrix using matrix multiplication.
* Returns the transformed coordinates in Cartesian form.

### ****3.3 Defining the Original Object****

rectangle = np.array([[0, 0], [2, 0], [2, 1], [0, 1]])

This defines a simple rectangular object with four corner points.

### ****3.4 Transformation Matrices****

The following transformation matrices were used:

#### ****3.4.1 Translation Matrix****

translation\_matrix = np.array([[1, 0, 2],

[0, 1, 3],

[0, 0, 1]])

**Effect:** Moves the rectangle 2 units right and 3 units up.

#### ****3.4.2 Scaling Matrix****

scaling\_matrix = np.array([[1.5, 0, 0],

[0, 2, 0],

[0, 0, 1]])

**Effect:** Stretches the rectangle by 1.5 times in the x-direction and 2 times in the y-direction.

#### ****3.4.3 Rotation Matrix****

rotation\_matrix = np.array([[np.cos(np.pi/4), -np.sin(np.pi/4), 0],

[np.sin(np.pi/4), np.cos(np.pi/4), 0],

[0, 0, 1]])

**Effect:** Rotates the rectangle 45 degrees counterclockwise around the origin.

## ****4. Results and Observations****

The notebook executes the transformations and visualizes the results. The output consists of multiple plots showing:

1. **Original Rectangle**: A simple rectangular shape.
2. **Translated Rectangle**: The shape shifted to a new position.
3. **Scaled Rectangle**: The shape resized while maintaining proportions.
4. **Rotated Rectangle**: The shape rotated by 45 degrees.

**Observations:**

* The translation transformation correctly shifts the object without altering its shape.
* Scaling increases the dimensions but does not affect the orientation.
* Rotation alters both position and orientation relative to the origin.

## ****5. Conclusion****

This lab provided hands-on experience in applying geometric transformations to 2D objects using transformation matrices. The exercise reinforced the importance of homogeneous coordinates in computer vision applications. Understanding these fundamental transformations is essential for more advanced topics like image warping and 3D transformations.

## ****6. Future Scope****

* Extend this approach to apply transformations on real images using OpenCV.
* Implement affine and perspective transformations.
* Explore transformations in 3D space for a better understanding of computer vision in real-world applications.

**End of Report**