Lab Manual 5- Affine Transformations: Translation, Rotation, Scaling, and Shearing

Introduction

Affine transformations are fundamental operations in image processing and computer vision. These transformations include translation, rotation, scaling, and shearing, which allow for the manipulation of images while preserving collinearity and parallelism.

1. Translation

Definition:

Translation shifts an image in the x and/or y direction without changing its shape or size.

Manual Calculation Example:

Given a point **P(x, y)** and a translation vector **(Tx, Ty)**, the new point **P'** is calculated as:

$x' = x + Tx \; y' = y + Ty$

Example:

- Let **P(2,3)**
- Translation vector **(4,5)**
- New coordinates: **P'(6,8)**

Application:

- Object tracking in videos
- Augmented reality for positioning elements

2. Rotation

Definition:

Rotation turns an image about a fixed point (typically the origin) by a specified angle θ.

Manual Calculation Example:

Given a point **P(x, y)**, the new coordinates after rotation by angle θ are:

$x' = x\cos\theta - y\sin\theta \; y' = x\sin\theta + y\cos\theta$

Example:

- **P(3,4)** rotated by **30°**

- Using $\cos 30° = 0.866, \sin 30° = 0.5$:
- **P'(0.598, 4.964)**

Application:

- Image alignment in computer vision
- Medical image registration

3. Scaling

Definition:

Scaling changes the size of an image by multiplying the coordinates by a scale factor Sx, Sy.

Manual Calculation Example:

Given a point **P(x, y)** and scaling factors **Sx, Sy**:

$x' = x \cdot Sx \; y' = y \cdot Sy$

Example:

- **P(2,3)** with **Sx=2, Sy=3**
- New coordinates: **P'(4,9)**

Application:

- Zooming in/out on images
- Resizing in computer graphics

4. Shearing

Definition:

Shearing distorts an image along one or both axes, changing its shape.

Manual Calculation Example:

For shear factors **Shx, Shy**:

$x' = x + Shx \cdot y \; y' = y + Shy \cdot x$

Example:

- **P(2,3)** with **Shx=1, Shy=0.5**
- **P'(5.5,4)**

Application:

- Slant correction in OCR (Optical Character Recognition)
- 3D perspective transformations

Python Implementation

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

def apply_transformation(image, matrix):
rows, cols = image.shape[:2]
transformed_image = cv2.warpAffine(image, matrix, (cols, rows))
return transformed_image

image = cv2.imread('input.jpg', 0)

# Translation
Tx, Ty = 50, 30
translation_matrix = np.float32([[1, 0, Tx], [0, 1, Ty]])
translated_image = apply_transformation(image, translation_matrix)

# Rotation
angle = 30
rotation_matrix = cv2.getRotationMatrix2D((cols/2, rows/2),
angle, 1)
rotated_image = apply_transformation(image, rotation_matrix)

# Scaling
Sx, Sy = 1.5, 1.5
scaling_matrix = np.float32([[Sx, 0, 0], [0, Sy, 0]])
scaled_image = apply_transformation(image, scaling_matrix)

# Shearing
Shx, Shy = 0.2, 0.3
shear_matrix = np.float32([[1, Shx, 0], [Shy, 1, 0]])
sheared_image = apply_transformation(image, shear_matrix)

# Display results
fig, axes = plt.subplots(1, 4, figsize=(15, 5))
axes[0].imshow(translated_image, cmap='gray'); axes[0].set_title('Translated')
axes[1].imshow(rotated_image, cmap='gray'); axes[1].set_title('Rotated')
axes[2].imshow(scaled_image, cmap='gray'); axes[2].set_title('Scaled')
axes[3].imshow(sheared_image, cmap='gray'); axes[3].set_title('Sheared')
plt.show()
```

Lab Tasks (2 Hours)

Task 1: Apply translation to an image and analyze the effect of different translation values. **Task 2:** Rotate an image by 30°, 60°, and 90° and compare

the results. **Task 3:** Apply scaling with different scale factors and note the impact. **Task 4:** Apply shearing with different shear factors and observe how the image distorts. **Task 5:** Implement all transformations in Python and save the resulting images.

This lab provides a comprehensive understanding of affine transformations and their applications in real-world image processing tasks.