# Lab-2: Image Transformation Codes

2. To perform Image Transformations using OpenCV in Python

Image Transformations in computer vision refer to techniques used to modify or manipulate an image in terms of its geometry, orientation, size, or pixel values without altering its essential content.

These transformations are crucial for tasks like image preprocessing, data augmentation, image registration, or computer vision model training.

🔄 Types of Image Transformations:

1. Geometric Transformations

Translation – Moves the image in x/y direction.

Rotation – Rotates the image by a certain angle.

Scaling – Resizes the image (zoom in/out).

Shearing – Tilts the image.

Flipping – Mirror the image horizontally or vertically.

import cv2

import numpy as np

📷 2. Read the Image

img = cv2.imread('your\_image.jpg')

cv2.imshow('Original Image', img)

cv2.waitKey(0)

cv2.destroyAllWindows()

🔀 3. Translation (Moving the image)

# Move image by 100px right and 50px down

M = np.float32([[1, 0, 100], [0, 1, 50]])

translated\_img = cv2.warpAffine(img, M, (cols, rows))

cv2.imshow('Translated Image', translated\_img)

cv2.waitKey(0)

cv2.destroyAllWindows()

🔄 4. Rotation

# Rotate by 45 degrees around the center

M = cv2.getRotationMatrix2D(center, 45, 1.0)

rotated\_img = cv2.warpAffine(img, M, (cols, rows))

cv2.imshow('Rotated Image', rotated\_img)

cv2.waitKey(0)

cv2.destroyAllWindows()

🔍 5. Resizing

resized\_img = cv2.resize(img, (300, 300), interpolation=cv2.INTER\_LINEAR)

cv2.imshow('Resized Image', resized\_img)

cv2.waitKey(0)

cv2.destroyAllWindows()

↔️ 6. Flipping

# Flip vertically (0), horizontally (1), both (-1)

flipped\_img = cv2.flip(img, 1)

cv2.imshow('Flipped Image', flipped\_img)

cv2.waitKey(0)

cv2.destroyAllWindows()

🧭 7. Affine Transformation

pts1 = np.float32([[50, 50], [200, 50], [50, 200]])

pts2 = np.float32([[10, 100], [200, 50], [100, 250]])

M = cv2.getAffineTransform(pts1, pts2)

affine\_img = cv2.warpAffine(img, M, (cols, rows))

cv2.imshow('Affine Transform', affine\_img)

cv2.waitKey(0)

cv2.destroyAllWindows()

🔳 8. Perspective Transformation

pts1 = np.float32([[56,65], [368,52], [28,387], [389,390]])

pts2 = np.float32([[0,0], [300,0], [0,300], [300,300]])

M = cv2.getPerspectiveTransform(pts1, pts2)

perspective\_img = cv2.warpPerspective(img, M, (300, 300))

cv2.imshow('Perspective Transform', perspective\_img)

cv2.waitKey(0)

cv2.destroyAllWindows()

Always call cv2.waitKey(0) and cv2.destroyAllWindows() after displaying an image.

Replace 'your\_image.jpg' with the actual path to your image.

Translation (Moving Image) in OpenCV

Translation means shifting an image along the X and/or Y axis. In computer vision, translation is commonly used for image augmentation or shifting focus within an image.

We use a transformation matrix to move the image:

import cv2

import numpy as np

# Load the image

image = cv2.imread('your\_image.jpg')

# Define translation values

# Create the translation matrix

M = np.float32([[1, 0, tx],[0, 1, ty]])

# Apply the translation

translated\_image = cv2.warpAffine(image, M, (image.shape[1], image.shape[0]))

# Display the result

cv2.imshow('Original Image', image)

cv2.imshow('Translated Image', translated\_image)

cv2.waitKey(0)

cv2.destroyAllWindows()

cv2.imread() – Reads an image.

np.float32() – Defines the matrix with 32-bit float values.

cv2.warpAffine() – Applies the affine transformation.

cv2.imshow() – Displays the image.

2. 🔄 Rotation in OpenCV (Python) with Code

In OpenCV, rotating an image involves using an affine transformation matrix. This is commonly done using cv2.getRotationMatrix2D() and cv2.warpAffine().

✅ Code to Rotate an Image

import cv2

import numpy as np

# Load the image

image = cv2.imread('your\_image.jpg')

# Get the image dimensions

(h, w) = image.shape[:2]

# Define the center of the image (around which rotation happens)

# Define the angle of rotation (positive = counterclockwise)

# Define the scale of the image

# Get the rotation matrix

rotation\_matrix = cv2.getRotationMatrix2D(center, angle, scale)

# Perform the rotation using warpAffine

rotated\_image = cv2.warpAffine(image, rotation\_matrix, (w, h))

# Display the rotated image

cv2.imshow('Rotated Image', rotated\_image)

cv2.waitKey(0)

cv2.destroyAllWindows()

cv2.getRotationMatrix2D(center, angle, scale):

cv2.warpAffine(image, matrix, (w, h)):

Let me know if you want this in a Jupyter notebook, with sample images, or rotation via a GUI slider.

3. Image Resizing-

3. In OpenCV, resizing an image refers to changing its dimensions (width and height) using the cv2.resize() function.

resized\_image = cv2.resize(src, dsize, interpolation)

src: Original image.

interpolation: (Optional) Interpolation method like cv2.INTER\_LINEAR, cv2.INTER\_AREA, etc.

cv2.INTER\_LINEAR: Default, good for upscaling.

cv2.INTER\_AREA: Preferred for downscaling.

cv2.INTER\_NEAREST: Fastest, lower quality.

import cv2

# Load the original image

image = cv2.imread('example.jpg')

# Resize to a specific width and height

resized = cv2.resize(image, (300, 200)) # width=300, height=200

# Resize by scale (e.g., half the size)

resized\_half = cv2.resize(image, (0, 0), fx=0.5, fy=0.5)

# Show images

cv2.imshow('Original', image)

cv2.imshow('Resized (300x200)', resized)

cv2.imshow('Resized Half', resized\_half)

cv2.waitKey(0)

cv2.destroyAllWindows()

Always use cv2.waitKey() and cv2.destroyAllWindows() to display and then close image windows properly.