TheAILearner

Mastering Artificial Intelligence

Affine Transformation

In this blog, we will discuss what is affine transformation and how to perform this transformation using OpenCV-Python. So, let's get started.

What is an Affine Transformation?

An affine transformation is any transformation that preserves collinearity, parallelism as well as the ratio of distances between the points (e.g. midpoint of a line remains the midpoint after transformation). It doesn't necessarily preserve distances and angles.

Thus all the geometric transformations we discussed so far such as translation, rotation, scaling, etc are all affine transformations as all the above properties are preserved in these transformations. To understand in simple terms, one can think of the affine transformation as a composition of rotation, translation, scaling, and shear.

In general, the affine transformation can be expressed in the form of a linear transformation followed by a vector addition as shown below

$$\underbrace{\begin{bmatrix} x' \\ y' \end{bmatrix}}_{\text{Transformed points}} = A \underbrace{\begin{bmatrix} x \\ y \end{bmatrix}}_{\text{Input points}} + B$$

Here,

$$A = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} \quad ; \quad B = \begin{bmatrix} b_{00} \\ b_{10} \end{bmatrix}$$

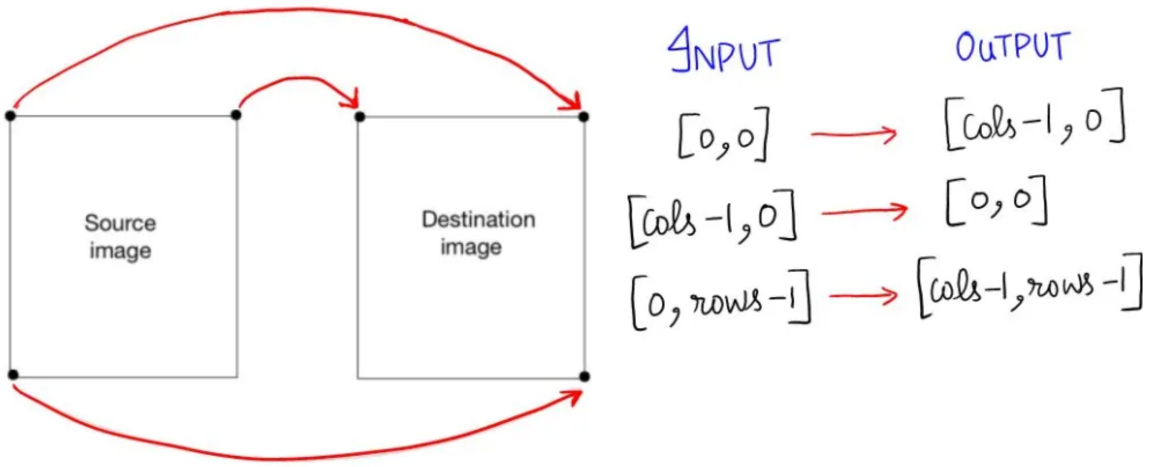
Combining A and B we can write,

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a_{00} & a_{01} & b_{00} \\ a_{10} & a_{11} & b_{10} \end{bmatrix}}_{\text{Transformation Matrix (M)}} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a_{00}x + a_{01}y + b_{00} \\ a_{10}x + a_{11}y + b_{10} \end{bmatrix}$$

Since the transformation matrix (M) is defined by 6 (2×3 matrix as shown above) constants, thus to find this matrix we first select 3 points in the input image and map these 3 points to the desired locations in the unknown output image according to the use-case as shown below (This way we will have 6 equations and 6 unknowns and that can be easily solved).

For instance, if you want to take the mirror image, you can define the 3 points as (you may choose any 3).



Once the transformation matrix is calculated, then we apply the affine transformation to the entire input image to get the final transformed image. Let's see how to do this using OpenCV-Python.

OpenCV

OpenCV provides a function cv2.getAffineTransform() that takes as input the three pairs of corresponding points and outputs the transformation matrix. The basic syntax is shown below.

```
transform_mat = cv2.getAffineTransform(src, dst)

# src: coordinates in the source image
# dst: coordinates in the output image
```

Once the transformation matrix (M) is calculated, pass it to the cv2.warpAffine() function that applies an affine transformation to an image. The syntax of this function is given below.

```
dst = cv.warpAffine(src, M, dsize[, dst[, flags[, borderMode[, borderValue]]]] )

# src: input image
# M: Transformation matrix
# dsize: size of the output image
# flags: interpolation method to be used
```

Now, let's take the above example of a mirror image and see how to apply affine transformation using OpenCV-Python. Below are the steps.

- Read the image
- Define the 3 pairs of corresponding points (See image above)
- Calculate the transformation matrix using cv2.getAffineTransform()
- Apply the affine transformation using cv2.warpAffine()

```
import cv2
import numpy as np

# Read the image
img = cv2.imread('D:/downloads/opencv_logo.PNG')
rows, cols = img.shape[:2]

# Define the 3 pairs of corresponding points
input_pts = np.float32([[0,0], [cols-1,0], [0,rows-1]])
```

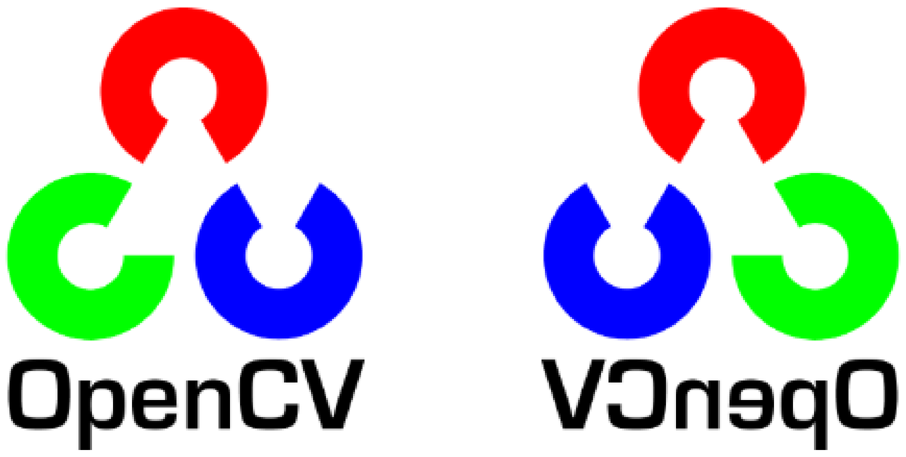
```
output_pts = np.float32([[cols-1,0], [0,0], [cols-1,rows-1]])

# Calculate the transformation matrix using cv2.getAffineTransform()
M= cv2.getAffineTransform(input_pts , output_pts)

# Apply the affine transformation using cv2.warpAffine()
dst = cv2.warpAffine(img, M, (cols,rows))

# Display the image
out = cv2.hconcat([img, dst])
cv2.imshow('Output', out)
cv2.waitKey(0)
```

Below is the output image. Here, left image represents the original image while the right one is the transformed mirror image.



Now define the 3 different point pairs and see how the transformation looks like. That's all for this blog. In the next blog, we will discuss Perspective transformation. Hope you enjoy reading.

If you have any doubts/suggestions please feel free to ask and I will do my best to help or improve myself. Good-bye until next time.

This entry was posted in Image Processing and tagged Affine Transformation meaning, Affine Transformation opencv, cv2.getAffineTransform(), cv2.warpAffine(), geometric transformation, image processing, opencv python on 4 Nov 2020 [https://theailearner.com/2020/11/04/affine-transformation/] by kang & atul.