Computer Visions

01416500

Course Code

Image Transformation

Asst. Prof. Dr. Anakkapon Saenthon

King Mongkut's Institute of Technology Ladkrabang

Image Transformation

Overview

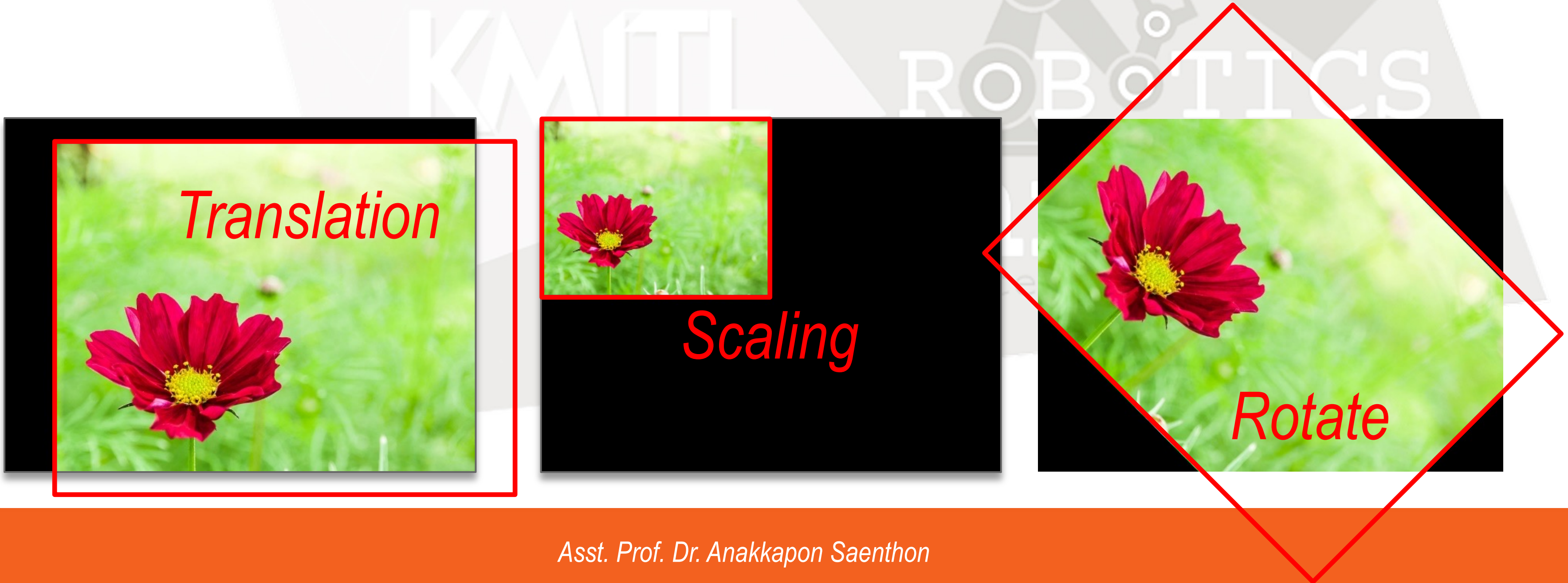
1. Basic geometric transformation
2. Applying geometry transformation to images
3. Resizing And Cropping Images

Objective

1. Learn to apply different geometric transformation to images like translation, rotation, affine transformation etc.
2. Learn to apply perspective Transforms.

Basic geometric transformation

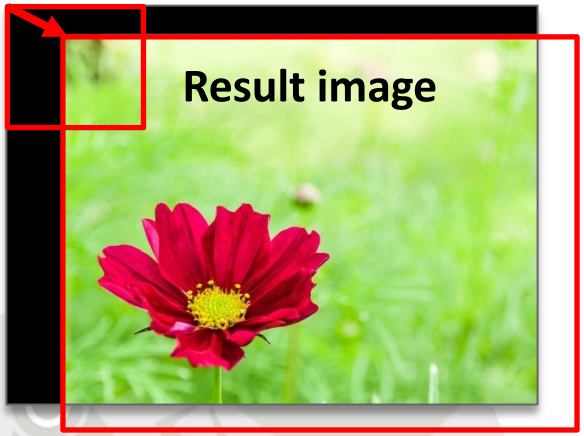
How to transform this image?

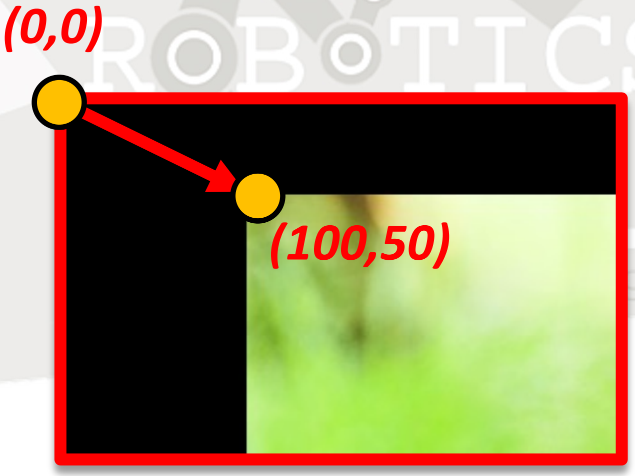


Basic geometric transformation

Example

Translation





This moves all the image pixels in the x-y direction, ***100 and 50 pixels***.

Basic geometric transformation

The image transformation or geometric transformation moves a pixel at coordinates (x,y) to a new position, (x',y'). The movement is specified by a pair of transformation equations:

Linear transformations

$$x' = Ax$$

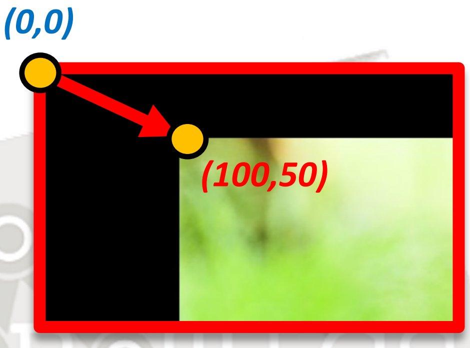
Affine transformations 1D

$$x' = Ax + b$$

Affine transformation 2D

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

Example



$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 100 \\ 50 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 100 \\ 50 \end{bmatrix}$$

Basic geometric transformation

2D affine transformation

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

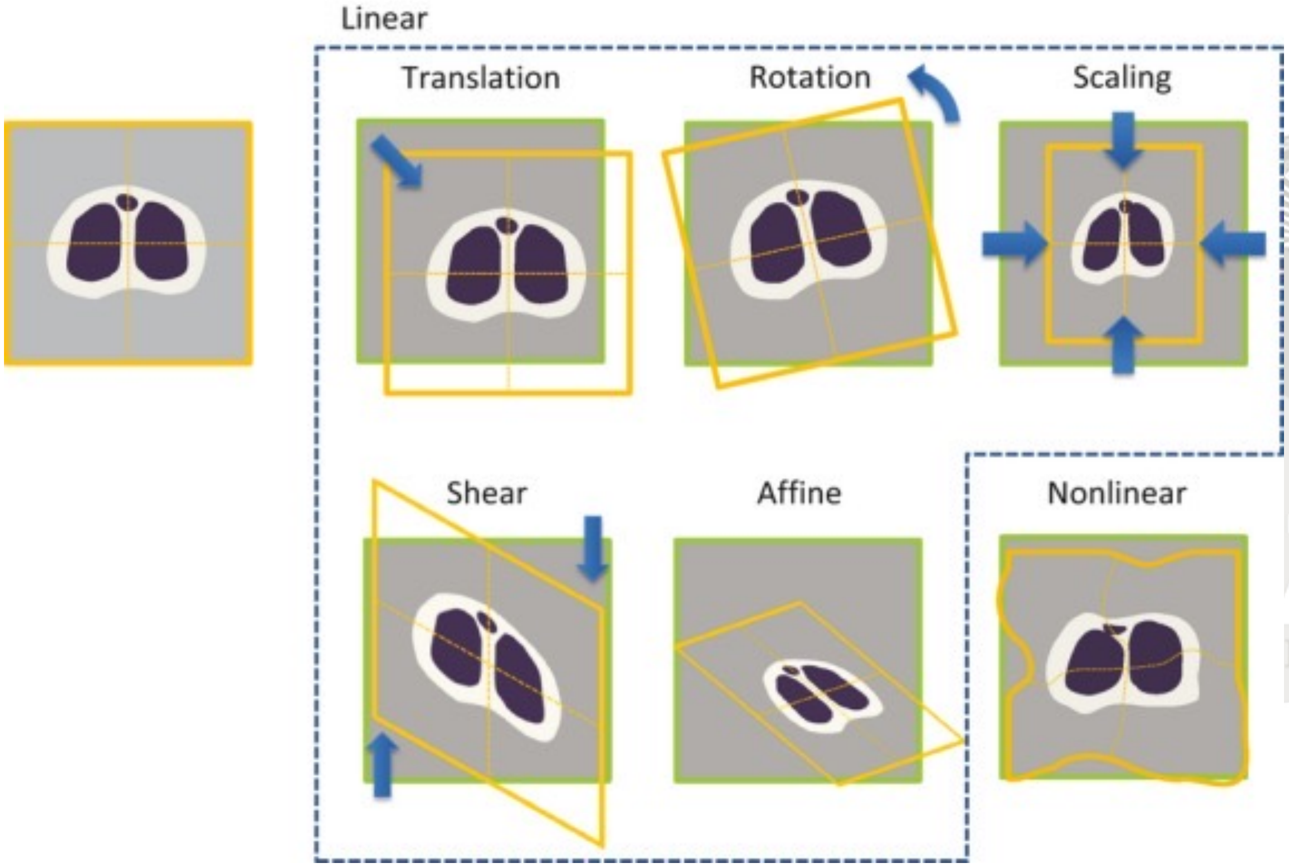
OpenCV function :
cv2.warpAffine()

This is transformation matrix of 2D affine.

$$M = \begin{bmatrix} a_1 & a_2 & b_1 \\ a_3 & a_4 & b_2 \end{bmatrix}_{2x3}$$

Basic geometric transformation

Transformation means changing some graphics into something else by applying rules.

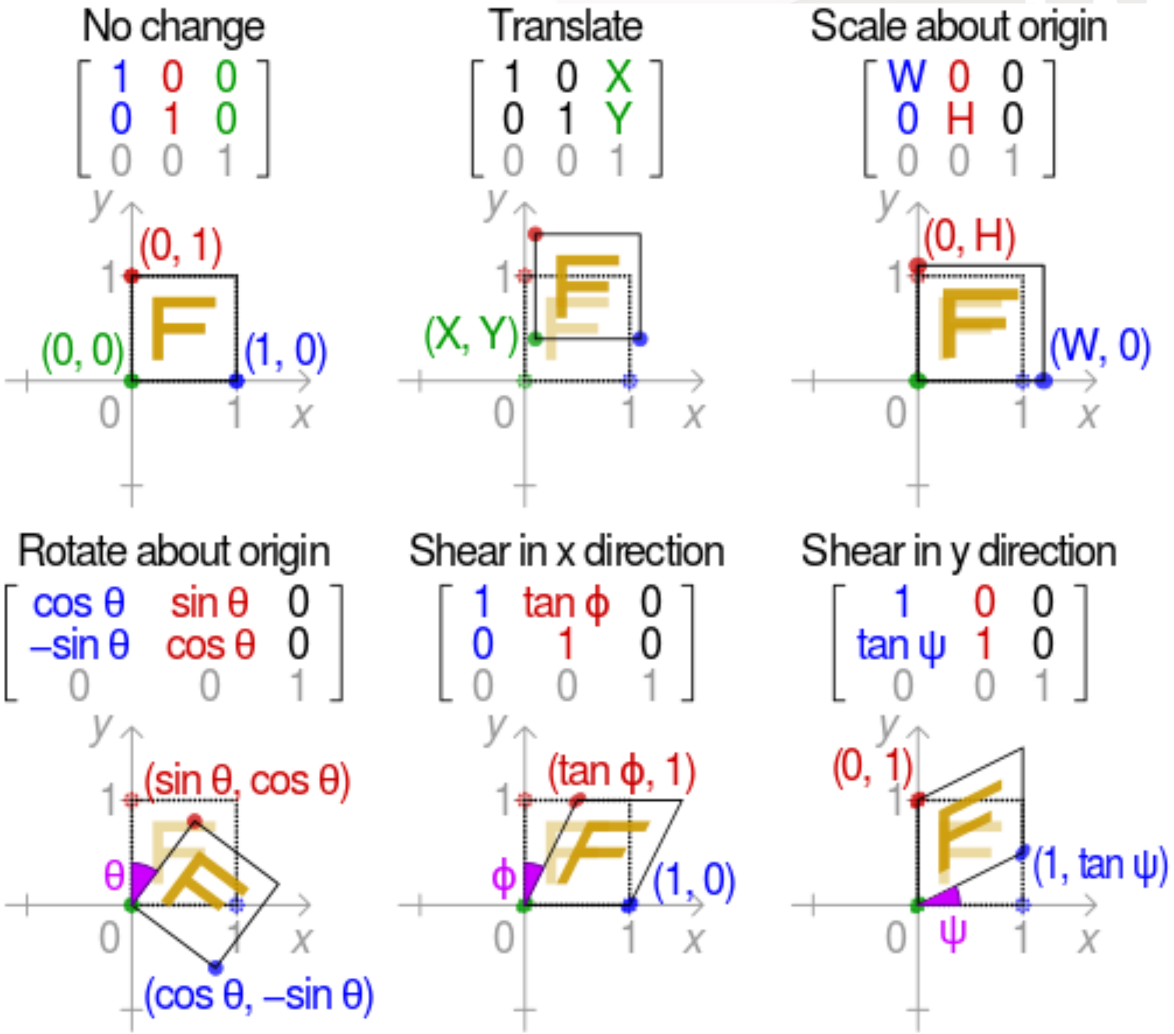


KMITL60 BEYOND THE LIMIT

Computer Visions

01416500
Course Code

Transformation Matrix



No change

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Translate

$$\begin{bmatrix} 1 & 0 & X \\ 0 & 1 & Y \\ 0 & 0 & 1 \end{bmatrix}$$

Scale about origin

$$\begin{bmatrix} W & 0 & 0 \\ 0 & H & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rotate about origin

$$\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Shear in x direction

$$\begin{bmatrix} 1 & \tan\phi & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Shear in y direction

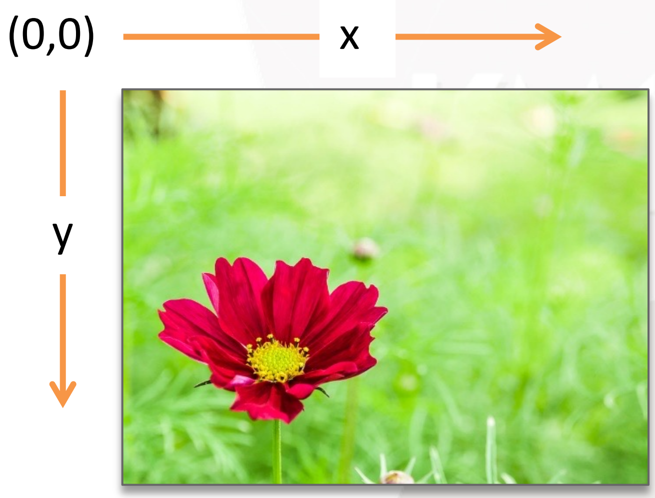
$$\begin{bmatrix} 1 & 0 & 0 \\ \tan\psi & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Asst. Prof. Dr. Anakkapon Saenthon
King Mongkut's Institute of Technology Ladkrabang

1. Translation

A translation moves an object to a different position on the screen

$$M = \begin{bmatrix} 1 & 0 & x \\ 0 & 1 & y \end{bmatrix}$$

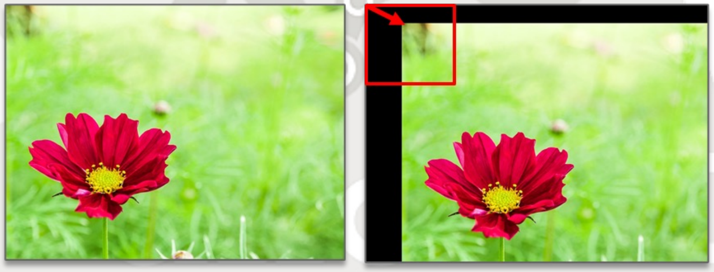


The dimension of translation matrix = 2x3

Example

See below example for a shift to **(100,50)**

$$M = \begin{bmatrix} 1 & 0 & 100 \\ 0 & 1 & 50 \end{bmatrix}$$



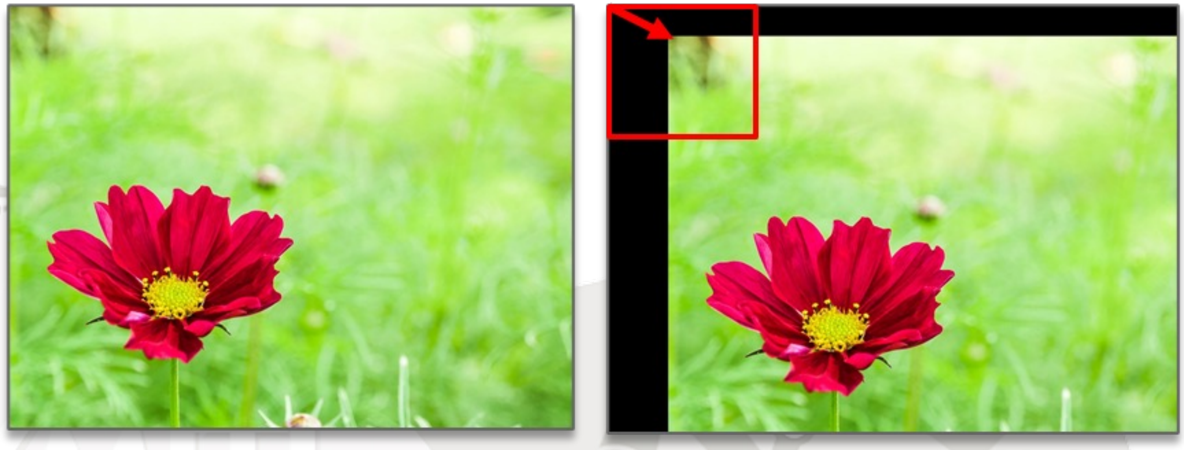
OpenCV function :

cv2.warpAffine()

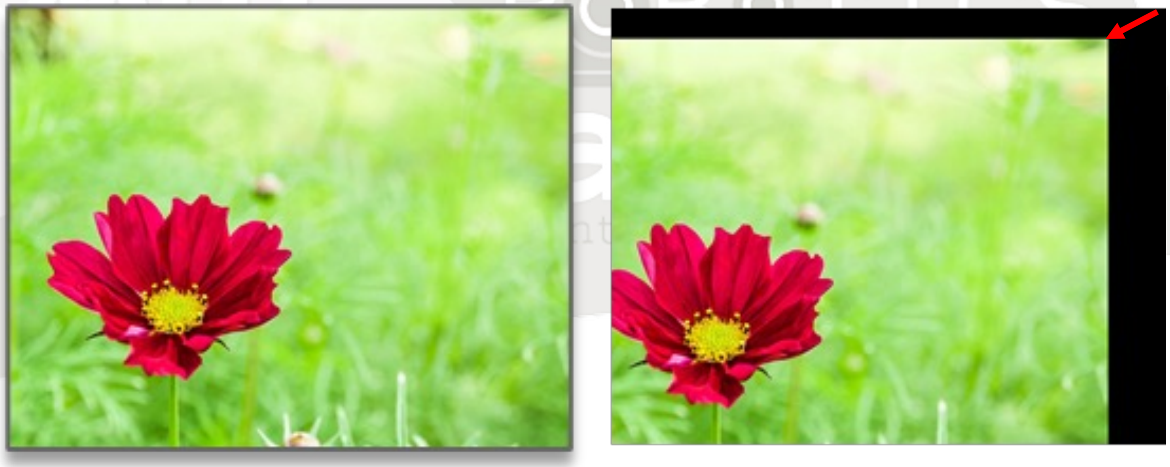
1. Translation *(continue)*

This example shows the different of x-value (+/-)

$$M = \begin{bmatrix} 1 & 0 & 100 \\ 0 & 1 & 50 \end{bmatrix}$$



$$M = \begin{bmatrix} 1 & 0 & -100 \\ 0 & 1 & 50 \end{bmatrix}$$



2. Scaling

Scaling is a linear transformation, and a special case of homothetic transformation.

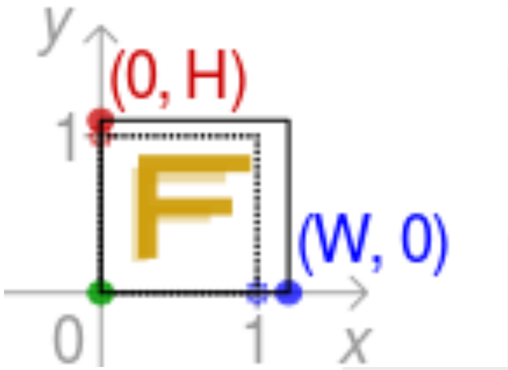
$$M = \begin{bmatrix} W & 0 & 0 \\ 0 & H & 0 \end{bmatrix}$$

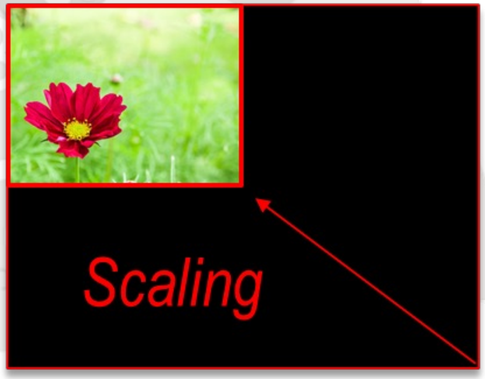
Example

See below example for scale image

0.5x

$$M = \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \end{bmatrix}$$





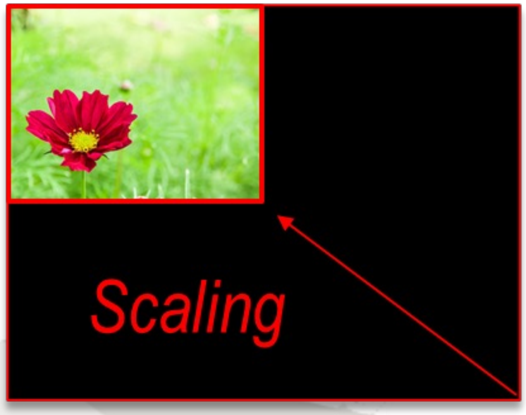
* W,H < 1, object is small , W,H > 1, object is big

2. Translation + Scaling (continue)

Example

See below example for

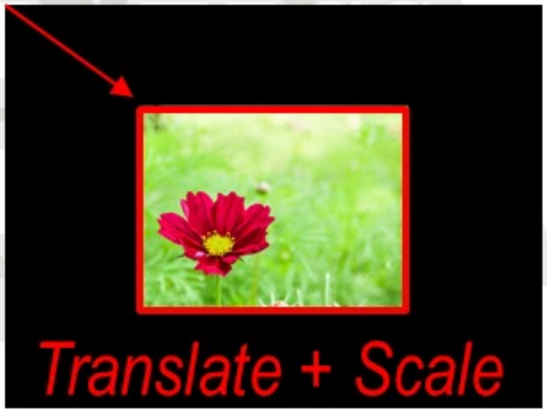
$$M = \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \end{bmatrix}$$



Translation : **(240,180)**

Scaling: **0.5x**

$$M = \begin{bmatrix} 0.5 & 0 & 240 \\ 0 & 0.5 & 180 \end{bmatrix}$$



3.Rotation

- Rotation of an image for an angle is achieved by the transformation matrix of the form

$$M = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \end{bmatrix}$$

- But OpenCV provides scaled rotation with adjustable center of rotation so that you can rotate at any location you prefer. Modified transformation matrix is given by

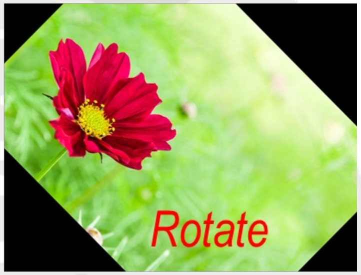
$$\begin{bmatrix} \alpha & \beta & (1-\alpha)\cdot \text{center.x} - \beta\cdot \text{center.y} \\ -\beta & \alpha & \beta\cdot \text{center.x} + (1-\alpha)\cdot \text{center.y} \end{bmatrix}$$

where

$$\alpha = \text{scale}\cdot\cos\theta,$$
$$\beta = \text{scale}\cdot\sin\theta$$

Example

See below example for rotate -**45 deg.**

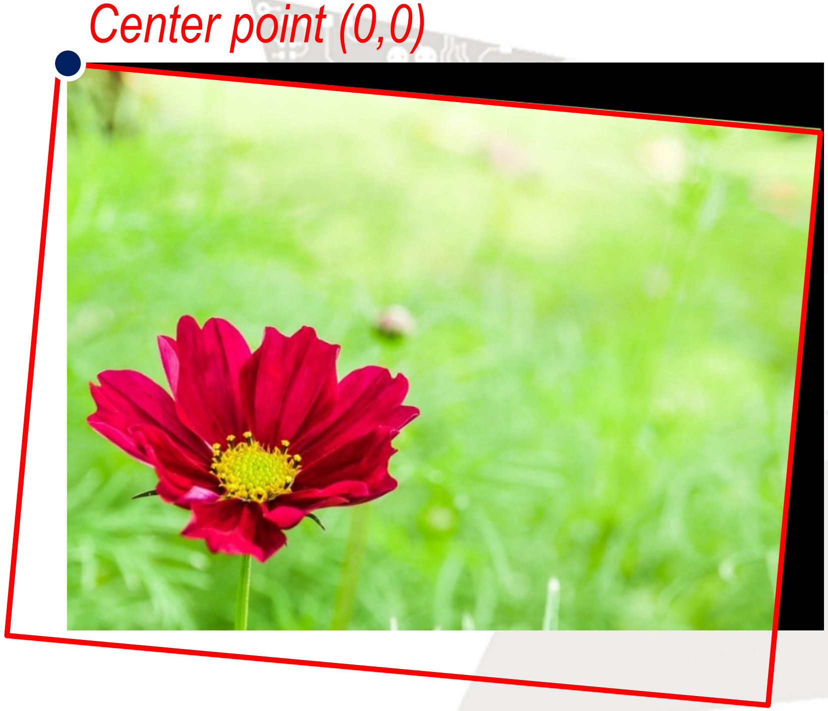


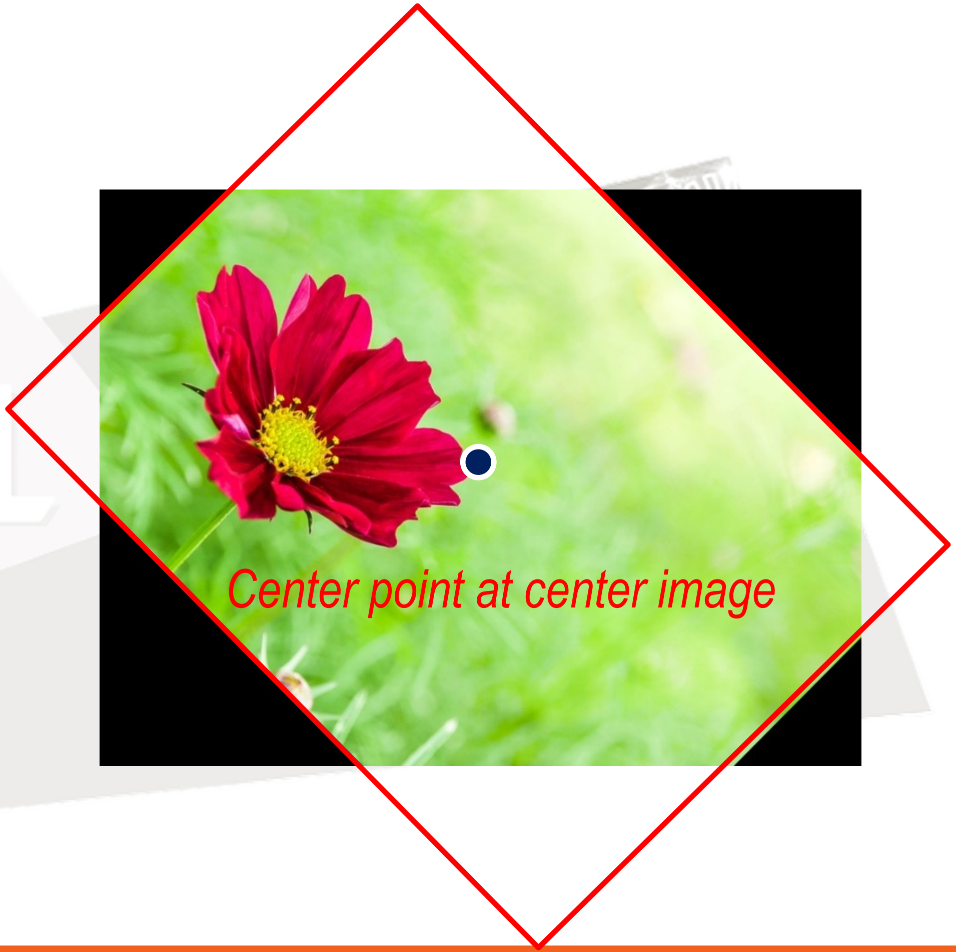
OpenCV function :

cv2. getRotationMatrix2D()

3.Rotation (continue)

This is the reference point for the different rotations





3.Rotation (continue)

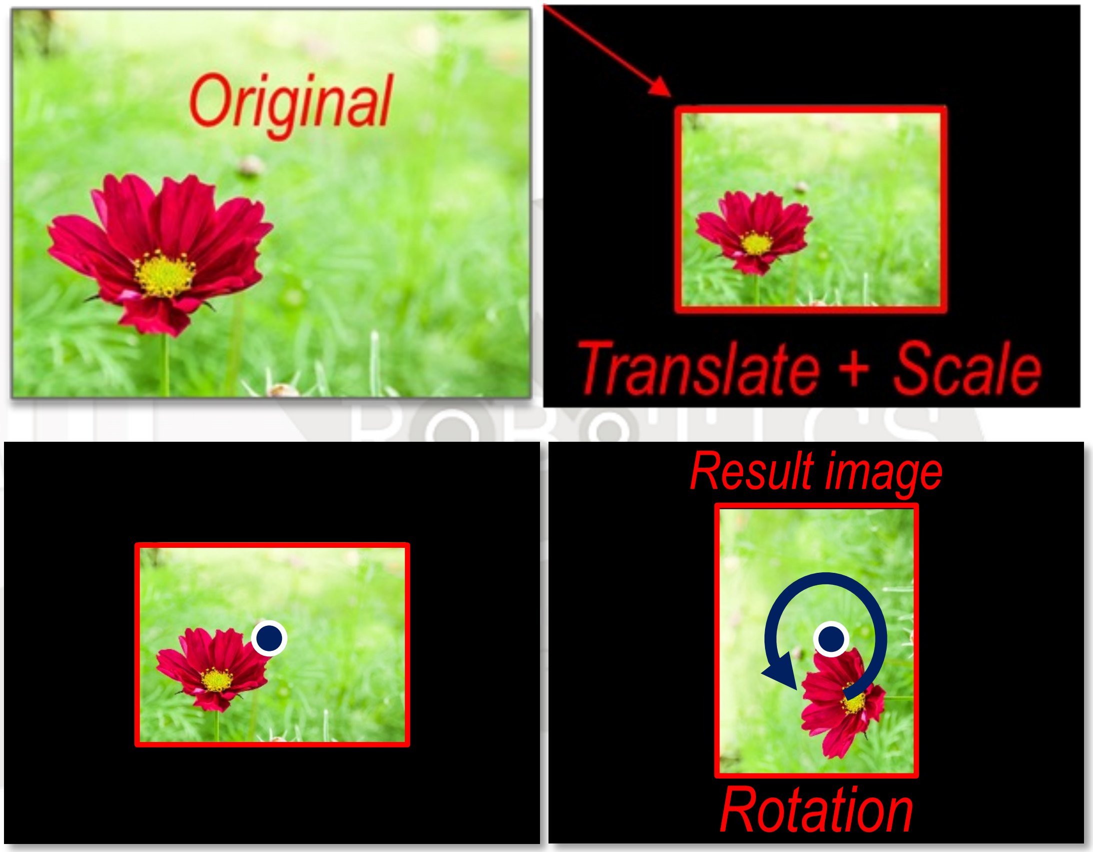
Translation + Scaling + Rotation

Step1: Translation + Scaling

$$M = \begin{bmatrix} 0.5 & 0 & 240 \\ 0 & 0.5 & 180 \end{bmatrix}$$

Step2: Rotation 90 deg (ref a center point)

$$M = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix}$$



4.Shear

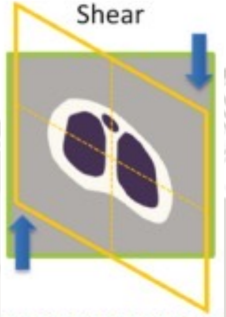
A transformation that slants the shape of an object is called the shear transformation. There are two shear transformations **X-Shear** and **Y-Shear**

X-Shear

$$M = \begin{bmatrix} 1 & shx & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

Y-Shear

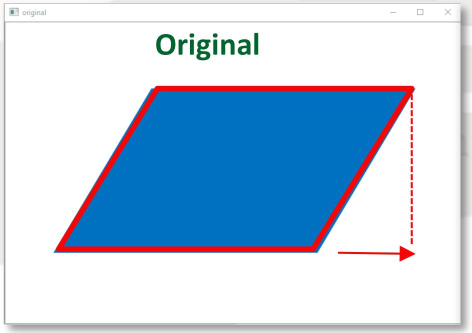
$$M = \begin{bmatrix} 1 & 0 & 0 \\ shy & 1 & 0 \end{bmatrix}$$

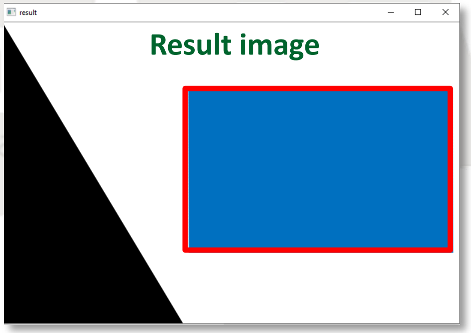


Example

See below example for a shear-X

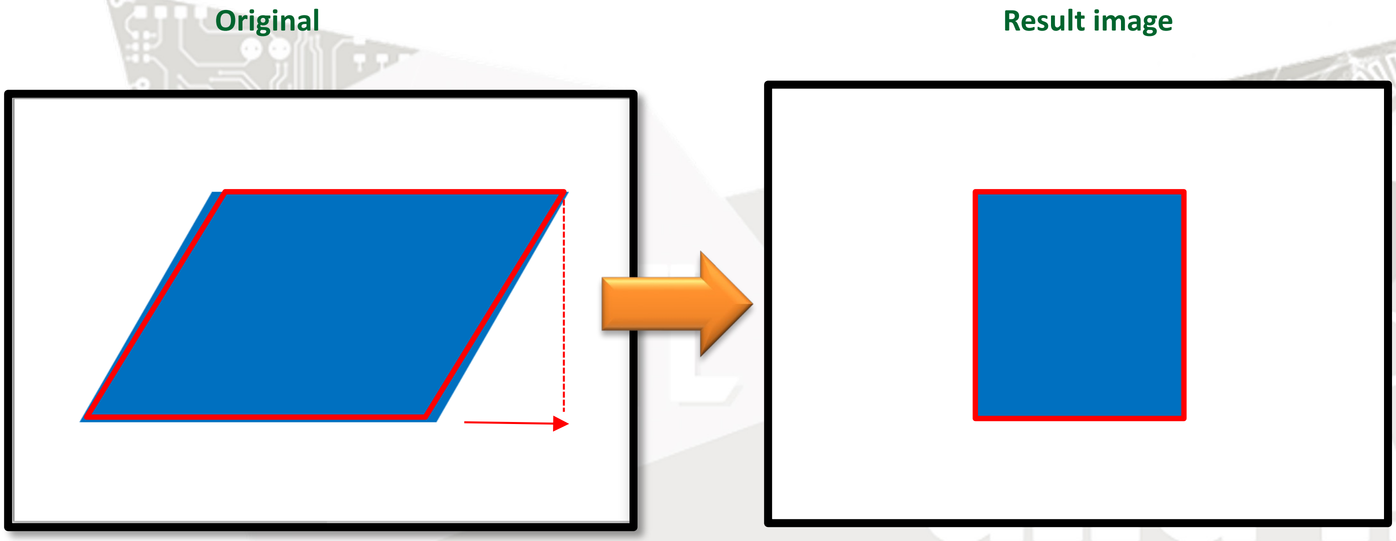
$$M = \begin{bmatrix} 1 & 0.6 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$





Quiz#3.1 (15 min)

Let’s you show how to apply transformation technique for convert original image to this result image ?



“rectangle2.png”

Condition

The object is the center of the picture
Object ratio (width/Height) = 1

5. Affine Transformation

- In affine transformation, all parallel lines in the original image will still be parallel in the output image.

$$M = \begin{bmatrix} a_1 & a_2 & b_1 \\ a_3 & a_4 & b_2 \end{bmatrix}$$

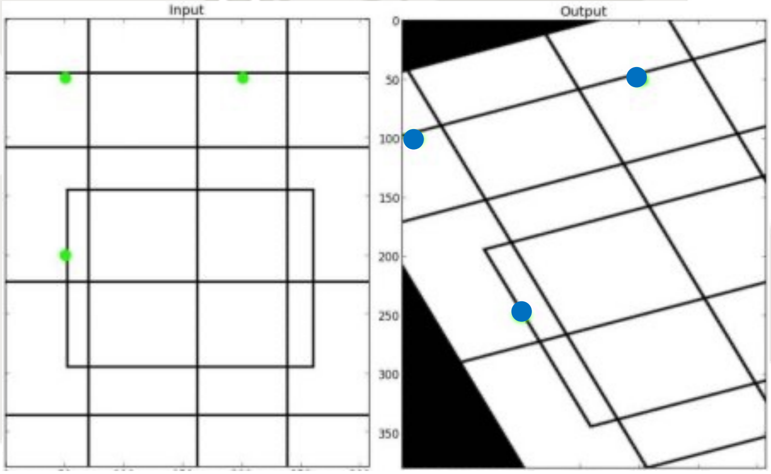
- To find the transformation matrix, we need **three points from input image** and their corresponding locations in output image.
  - Original point **(x0,y0), (x1,y1), (x2,y2)**
  - Target point **(x0,y0), (x1,y1), (x2,y2)**

OpenCV function :

cv2.getAffineTransform()

Example

See below example for affine transformation from ***3 points***



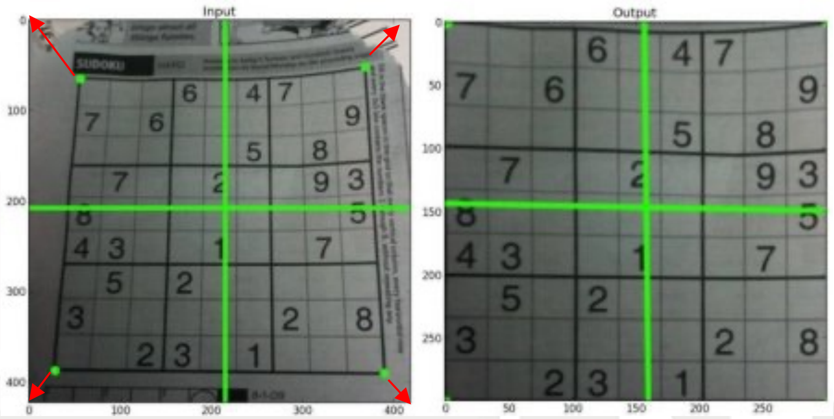
6.Perspective Transformation

- For perspective transformation is a **3x3 matrix.**
- Straight lines will remain straight even after the transformation.

- To find this transformation matrix, you **need 4 points** on the input image and corresponding points on the output image.

Example

See below example for Perspective transformation from ***4 points***



OpenCV function :

cv2.getPerspectiveTransform()
cv2.warpPerspective()

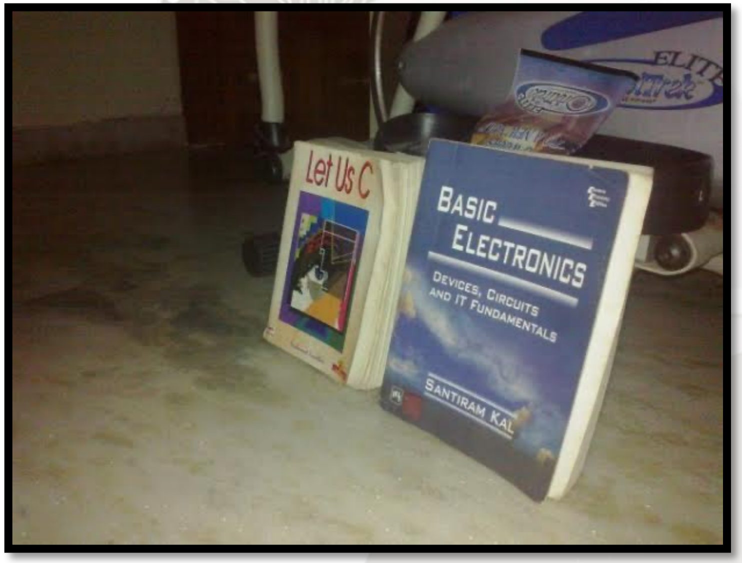
Quiz#3.2 (15 min)

Let's you apply **perspective technique** fit blue book to the border image.

"right.jpg"

W = 300 pixel

H = 420 pixel



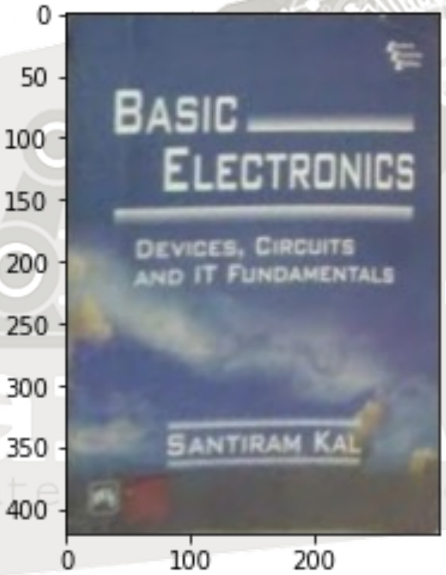


Image Transformation

- OpenCV provides two transformation functions, **cv2.warpAffine** and **cv2.warpPerspective**,

- You can have all kinds of transformations.

  cv2.warpAffine takes a 2x3 transformation matrix,

  cv2.warpPerspective takes a 3x3 transformation matrix as input.

KMITL 60
BEYOND THE LIMIT

COMPUTER VISIONS

01416500
Course Code

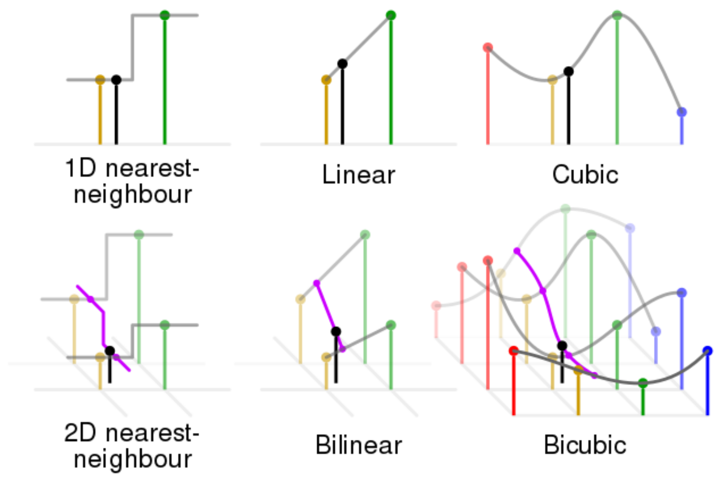
OpenCV

Resizing And Cropping Images

Asst. Prof. Dr. Anakkapon Saenthon
King Mongkut's Institute of Technology Ladkrabang

1. cv2.resize()

- Scaling is just resizing of the image.
- The size of the image can be specified manually, or you can specify the **scaling factor**. Different interpolation methods are used. Preferable interpolation methods are
  - **cv2.INTER_AREA** for shrinking
  - **cv2.INTER_CUBIC** (slow) zooming
  - **cv2.INTER_LINEAR** for zooming.



- By default, interpolation method used is **cv2.INTER_LINEAR** for all resizing purposes. You can resize an input image either of following methods

1. OpenCV function

Example

The image has changed the resolution

OpenCV function :
cv2.resize()

Resolution (720, 960, 3)

Resolution (1440, 1920, 3)

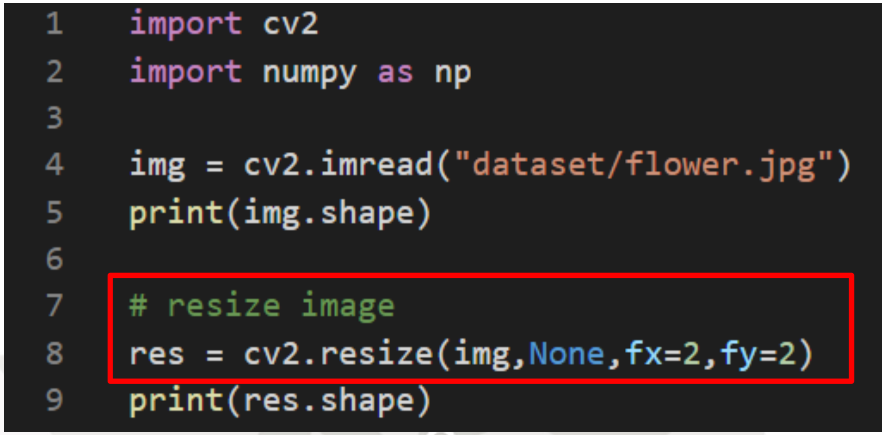
1. OpenCV function

OpenCV function :

cv2.resize()

Solution #1

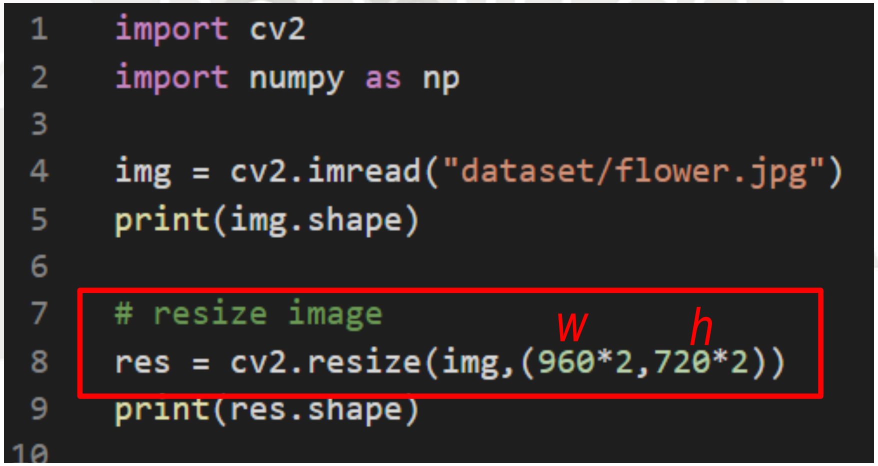
Set factor (fx,fy)



```
import cv2
import numpy as np

img = cv2.imread("dataset/flower.jpg")
print(img.shape)

# resize image
res = cv2.resize(img,None,fx=2,fy=2)
print(res.shape)
```

Solution #2

Set Image resolution (width, height)



```
import cv2
import numpy as np

img = cv2.imread("dataset/flower.jpg")
print(img.shape)

# resize image
res = cv2.resize(img,(960*2,720*2))
print(res.shape)
```

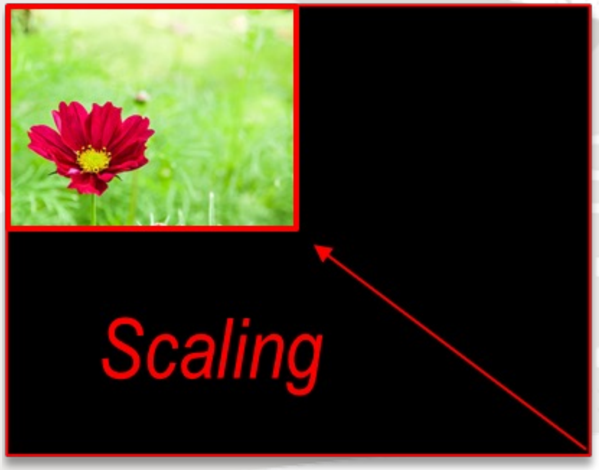
1. OpenCV function

What's the different between resize() with Scaling (image transformation) ?

resize

The result image is new resolution

Scaling



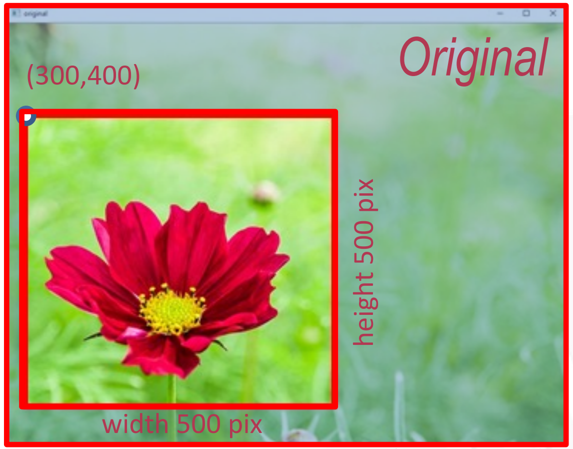
The result image will keep the background

1. OpenCV function

Cropping is the **removal of unwanted outer areas** from a photographic or illustrated image.

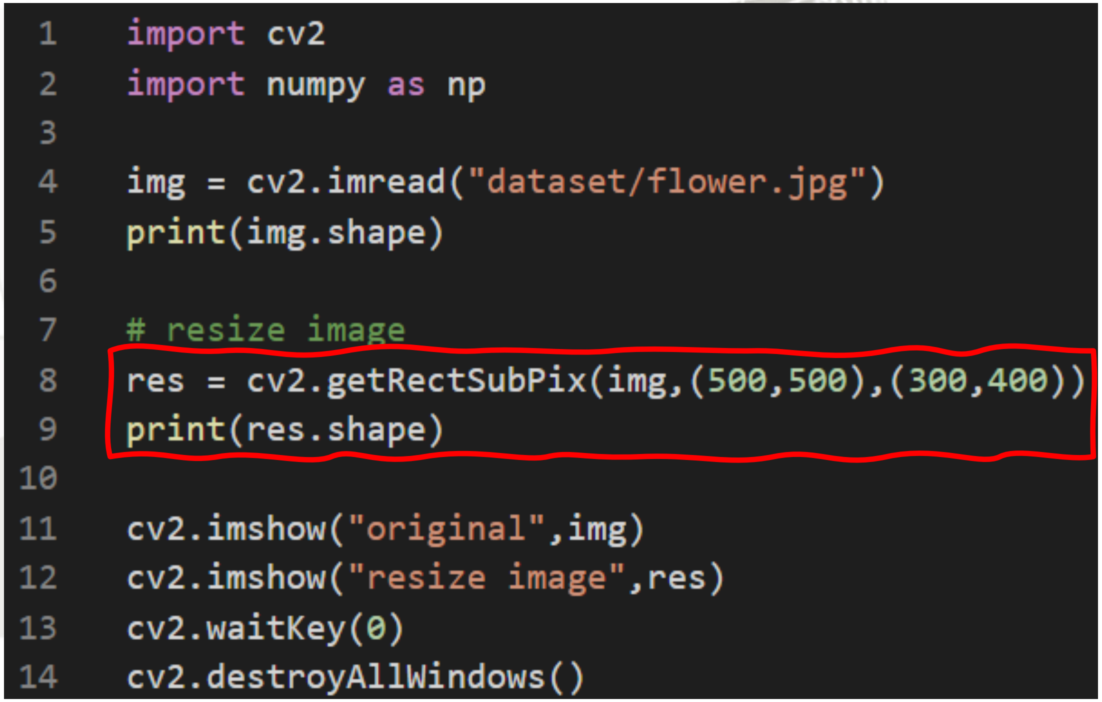
OpenCV function :

cv2.getRectSubPix()

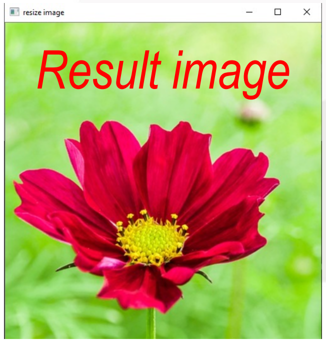


(720, 960, 3)

cv2.getRectSubPix(**img , (width , height) , (x , y)**)



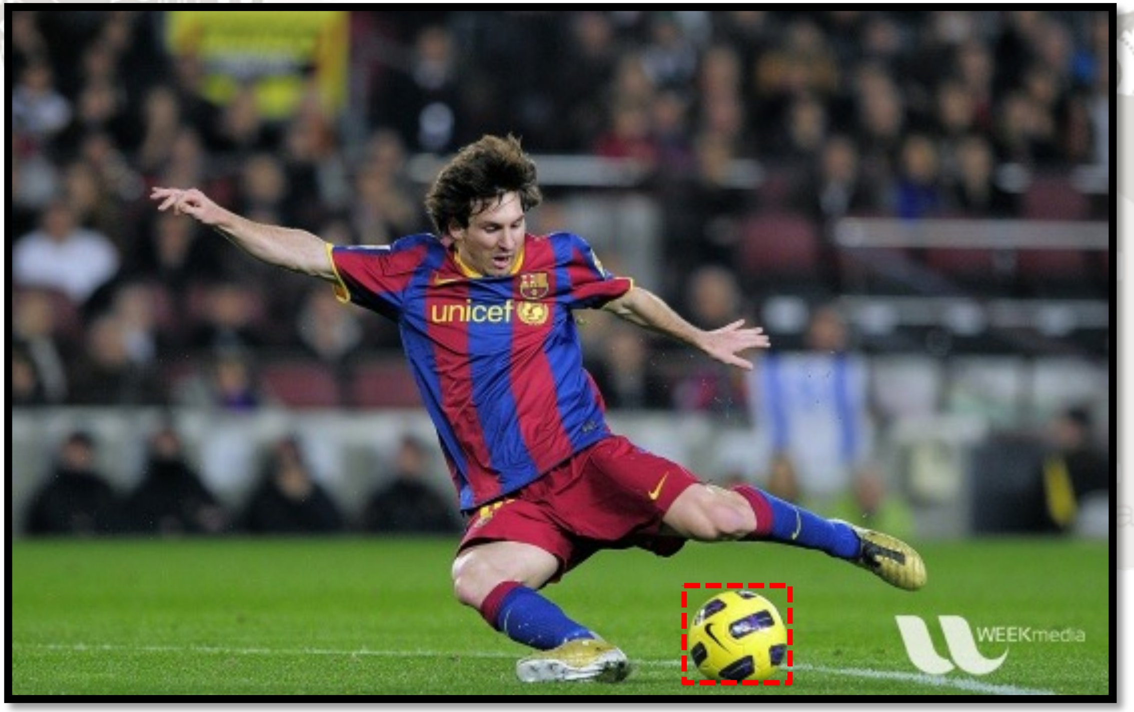
```
import cv2
import numpy as np

img = cv2.imread("dataset/flower.jpg")
print(img.shape)

# resize image
res = cv2.getRectSubPix(img,(500,500),(300,400))
print(res.shape)

cv2.imshow("original",img)
cv2.imshow("resize image",res)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



(500, 500, 3)

Quiz#3 (15 min)

Let's you show how to crop the ball and resize the image to width, height (300,300). "messi.jpg"



Computer Visions

01416500
Course Code

Image Transformation

1. Translation
2. Scaling
3. Rotation
4. Affine transformation
5. Shearing
6. Perspective transformation

Next class >>> *Edge Detection*

You can see on YouTube link

Asst. Prof. Dr. Anakkapon Saenthon
King Mongkut's Institute of Technology Ladkrabang