Scaling and Transformation

Miran Baban
Komar University


```
import cv2
```

Import packages

```
img =  cv2.imread('box.jpg')
```

Read Images

```
cv2.imshow(img)
```

Show Images

```
cv2.waitKey(0)
```

Waiting for key press

```
cv2.destroyAllWindows()
```

Destroy window (Close)

```
import cv2
```
→ Import packages

```
img =  cv2.imread('box.jpg')
```
→ Read Images

```
cv2.imshow(img)
```
→ Show Images

Scaling,Translation, RGB2Gray, bitwise operation, blending….
→ Image Manipulation

```
cv2.imshow(img) after manipulation
```
→ Image Manipulation

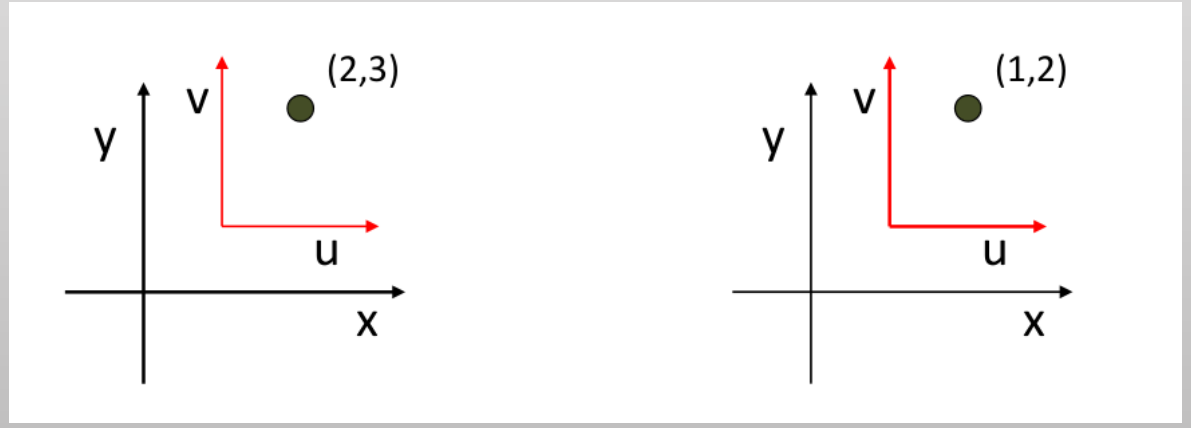
```
cv2.imwrite(img, “filename.jpg”)
```
→ Show Images

```
cv2.waitKey(0)
```
→ Waiting for key press

```
cv2.destroyAllWindows()
```
→ Destroy window (Close)

Image Transformation

- Picture processing is a technique for applying operations on an image in order to improve it or extract relevant information from it.
- It's a sort of signal processing in which the input is an image and the output is either that image or its characteristics/features.
- Image processing is one of the most quickly evolving technology today. It is also a critical research field in engineering and computer science.



Translation

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

Rotation

$$\begin{bmatrix} \cos(\Theta) & -\sin(\Theta) & 0 \\ \sin(\Theta) & \cos(\Theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Scale

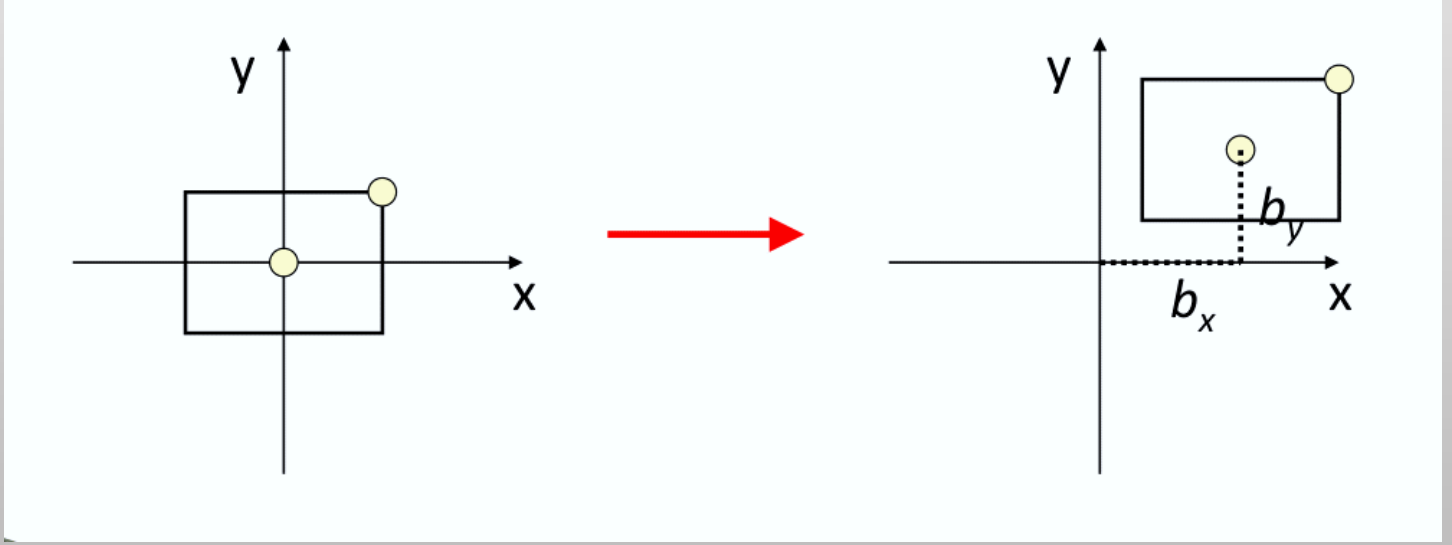
$$\begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Shear

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

Image Scale

- Image transformation is a coordinate-changing function that transforms (x, y) coordinates in one coordinate system to (x', y') points in another.
- If we plot the same point in u-v coordinate with (2, 3) points in x-y coordinate, the same point is represented in multiple ways.

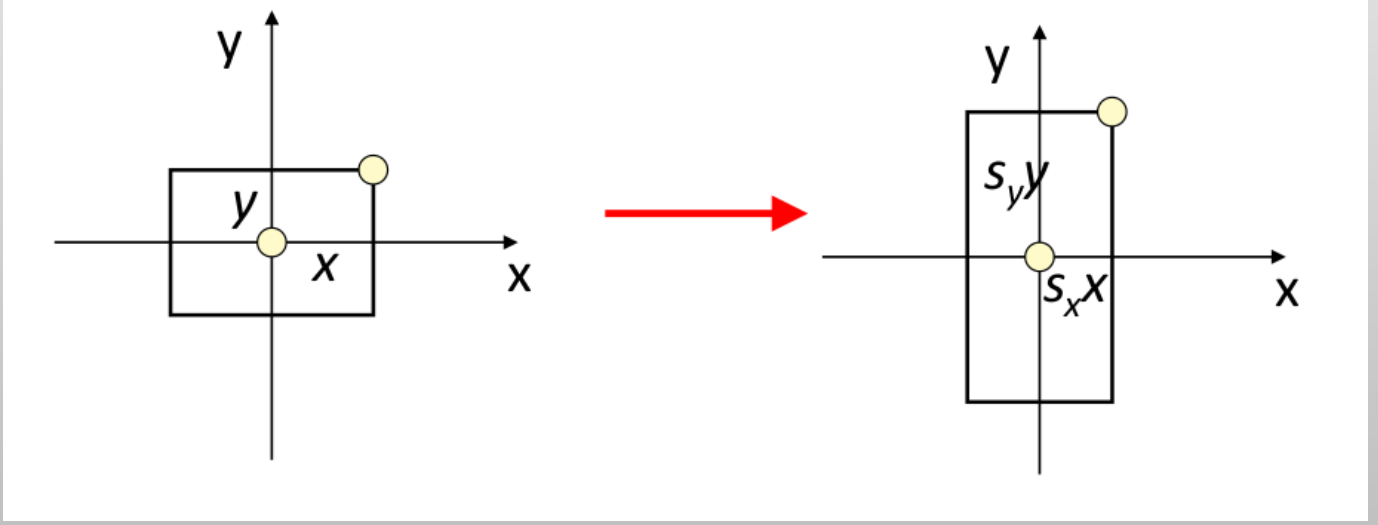


```
M = np.float32(
[[1, 0, 50],
[0, 1, 50],
[0, 0, 1]]
)
```

```
translated_img = cv2.warpPerspective(img, M, (cols, rows))
```

Image Scaling

- The technique of resizing a digital image is known as image scaling. OpenCV has a built-in function cv2.resize(), but we will perform transformation using matrix multiplication as previously.
- S_x and S_y are the scaling factors for x-axis and y-axis, respectively.



```
M = np.float32(
[[1.5, 0 , 0],
[0, 1.8, 0],
[0, 0, 1]]
)
```

```
scaled_img = cv2.warpPerspective(img,M,(cols*2,rows*2))
```

Image Rotation

- Rotation is a mathematical term that describes the motion of a certain space while preserving at least one point.
- Image rotation is a typical image processing method that has applications in matching, alignment, and other picture-based algorithms, as well as data augmentation, parti ation.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

2D *in-plane* rotation

```
M = np.float32(

[[np.cos(angle), -(np.sin(angle)), 0],
[np.sin(angle), np.cos(angle), 0],
[0, 0, 1]]

)
```

```
cv2.warpPerspective(img, M, (int(cols),int(rows)))
```

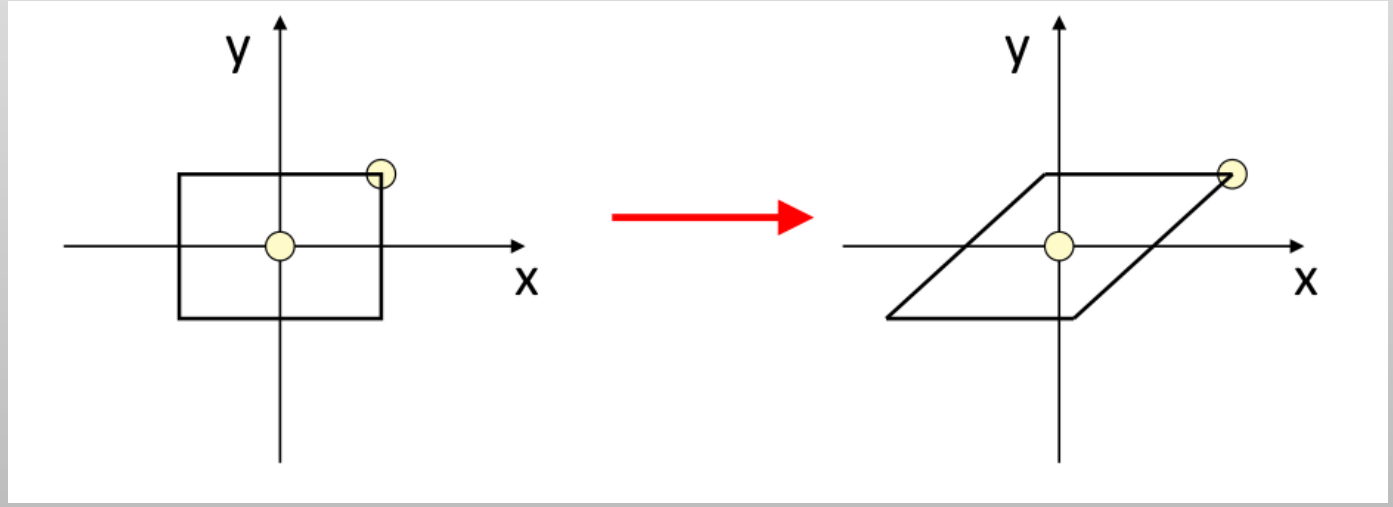
Image Crop

- Cropping is the process of removing undesired exterior sections from an image.
- Many of the examples above included black pixels, which you can simply remove by cropping.

```
cropped_img = img[100:300, 100:300]
```

Image Shearing

- Shear mapping is a linear map that displaces each point in a specified direction, substituting a specific value in proportion to its x or y coordinates for each point horizontally or vertically. There are two sorts of shearing effects.
- Shearing in the x-axis direction and shearing in the y-axis direction are two different types of shearing.



Matrix Operation

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & sh_x & 0 \\ sh_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Shear

Image Shearing X-axis

- When shearing is done in the x-axis direction, the image's boundaries parallel to the x-axis stay put, while the borders parallel to the y-axis shift about depending on the shearing factor.

```
M = np.float32(
[[1, 0.5, 0],
[0, 1 , 0],
[0, 0 , 1]]
)
```

Image Shearing Y-axis

- When shearing is done in the y-axis direction, the image's boundaries parallel to the y-axis stay put, while the borders parallel to the x-axis shift about depending on the shearing factor.

```
 M = np.float32(
[[1, 0, 0],
[0.5, 1, 0],
[0, 0, 1]]
)
```

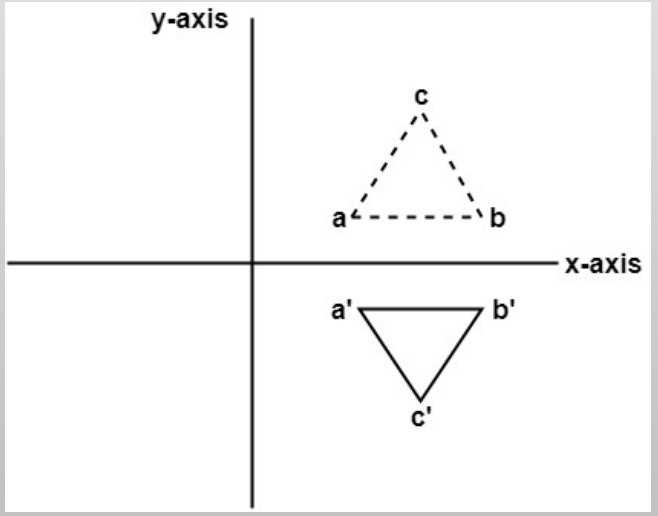
Image Reflection

- When shearing is done in the y-axis direction, the image's boundaries parallel to the y-axis stay put, while the borders parallel to the x-axis shift about depending on the shearing factor.

```
 M = np.float32(
[[1, 0, 0],
[0.5, 1, 0],
[0, 0, 1]]
)
```

Image Reflection

- Picture reflection (or mirroring) is a type of scaling that can be used to flip an image vertically or horizontally.
- We set Sy to -1 and Sx to 1 for reflection along the x-axis, and vice versa for reflection along the y-axis.



$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & \text{rows} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Reflection x-axis

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & \text{cols} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Reflection y-axis

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

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

```
cv2.warpPerspective(img,M,(int(cols),int(rows)))
```