Lab 02

Lab 02

- Image manipulation
- Transformation

(recap)

Using numpy

- import numpy as np

(recap)

Arrays

- Arrays represent matrices in numpy

```
a = np.array([1, 2, 3]) # Create a rank 1 array
print(type(a), a.shape, a[0], a[1], a[2])
a[0] = 5 # Change an element of the array
print(a)

b = np.array([[1,2,3],[4,5,6]]) # Create a rank 2 array
print(type(b), b.shape)
print(b)
print(b[0, 0], b[0, 1], b[1, 0])
```

```
<class 'numpy.ndarray'> (3,) 1 2 3
[5 2 3]
<class 'numpy.ndarray'> (2, 3)
[[1 2 3]
 [4 5 6]]
1 2 4
```

(recap)

Image manipulation

- scikit-image
  - from skimage import io
  - from skimage import color

(recap)

Load image

- io.imread returns numpy array of shape (height, width, 3)

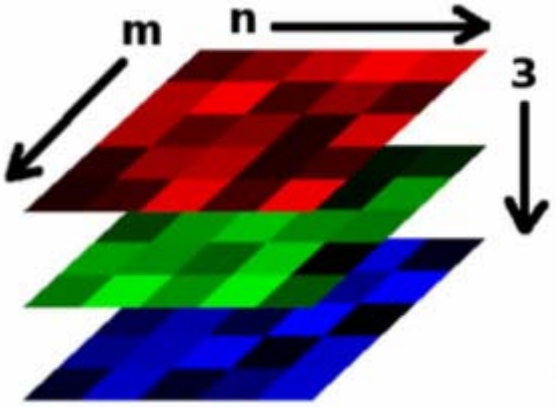
```
>>> from skimage import io
>>> asdf = io.imread('image1.jpg')
>>> asdf.shape
(300, 300, 3)
```

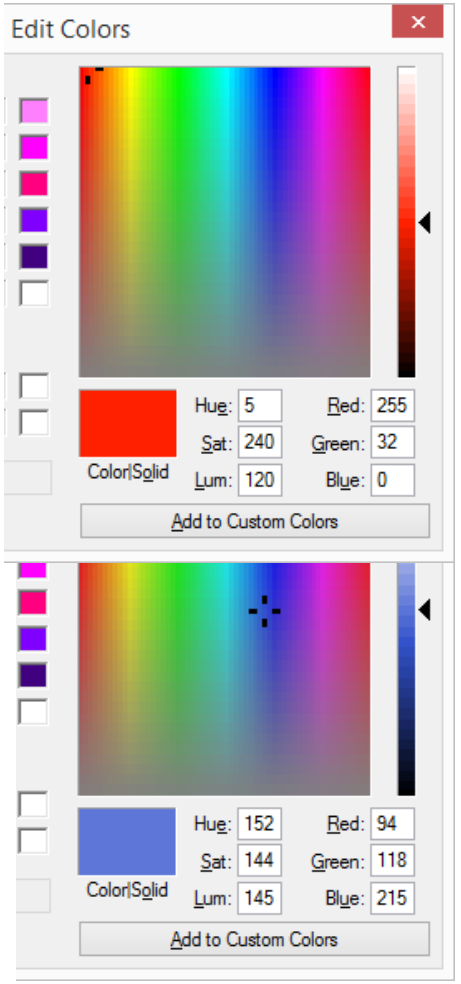
(recap)

Image representation

- numpy array of (height, width, channels)
  - (300,300,3)
  - 3 channels are for R,G,B
  - Uint8 →[0,255]
  - Float→[0,1]

=



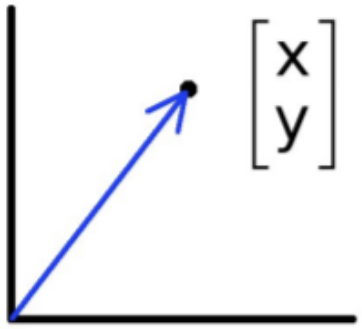


Draw (small) square

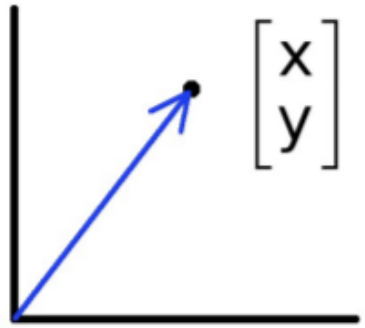
```
def draw_square(img,p):
    x=p[0]
    y=p[1]
    # do not worry about out of range
    img[                    ] = 1
img = np.zeros((200,300,3)) # default float
draw_square(img,5,8)
io.imsave('image.png',img)
```

Vectors

- Vectors can represent an offset in 2D or 3D space



Transformation

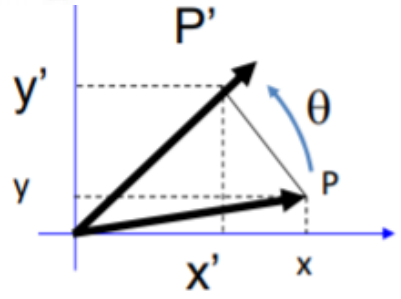


- Multiply a matrix to a vector to transform the vector

scaling

$$\begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \times \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} s_x x \\ s_y y \end{bmatrix}$$

Counter-clockwise rotation by an angle θ



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

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

$$\mathbf{P'} = \mathbf{R}\,\mathbf{P}$$

Scaling

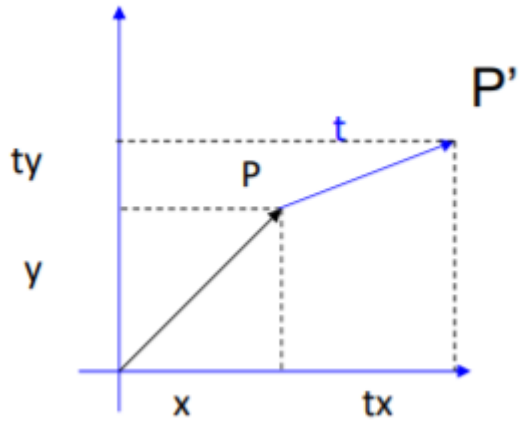
```
img = np.zeros((200,300,3))
p1 = np.array([5,6])
S=
p2 =
draw_square(img,p1)
draw_square(img,p2)
io.imsave('p2.png', img)
```

Rotation

```
from math import cos, sin, pi
img = np.zeros((200,300,3))
p1 = np.array([150,20])
theta = 0.25*pi
R =

p3 = np.dot(R,p1)
draw_square(img,p1)
draw_square(img,p3)
io.imsave('p3.png', img)
```

Translation



$$\mathbf{P} = (x, y) \rightarrow (x, y, 1)$$

$$\mathbf{t} = (t_x, t_y) \rightarrow (t_x, t_y, 1)$$

$$\mathbf{P'} \rightarrow \begin{bmatrix} x + t_x \\ y + t_y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

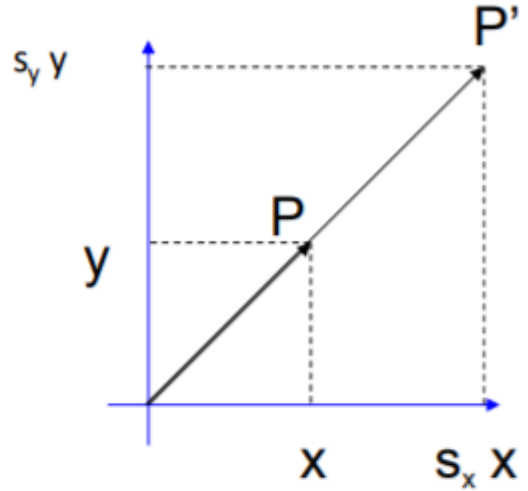
t P

$$= \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ 0 & 1 \end{bmatrix} \cdot \mathbf{P} = \mathbf{T} \cdot \mathbf{P}$$

Translation

```
img = np.zeros((200,300,3))
p1 = np.array([5,6,1])
T =
p4 =
draw_square(img,p1)
draw_square(img,p4)
io.imsave('p4.png',img)
```

Scaling



$$\mathbf{P} = (x, y) \rightarrow \mathbf{P}' = (s_x x, s_y y)$$

$$\mathbf{P} = (x, y) \rightarrow (x, y, 1)$$

$$\mathbf{P}' = (s_x x, s_y y) \rightarrow (s_x x, s_y y, 1)$$

$$\mathbf{P}' \rightarrow \begin{bmatrix} s_x x \\ s_y y \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{S}} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{S}' & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \cdot \mathbf{P} = \mathbf{S} \cdot \mathbf{P}$$

homework

- Write a function "transform(p, s_x, s_y, theta, t_x, t_y)" to do the followings:
  - Scale by s_x, s_y,
  - Rotate (ccw) by theta
  - Translate by t_x, t_y

specification

- Write hw_[student_id].py containing a function transform()
- zip to submit

Pre-check your score

- Use grader.py
  - It should print 0