

2D Looping

Looping Through a 1D Array

- For a 1D array

```
>>> data = np.array([1,2,3,4,5,6])  
>>> for contents in data:  
    print(contents)
```

```
1  
2  
3  
4  
5  
6
```

Looping Through a 2D Array

- But for a 2D array

```
>>> data = np.array([[1,2,3],[4,5,6]])  
>>> for contents in data:  
    print(contents)
```

```
[1 2 3]  
[4 5 6]
```

- Not every single “item” but the two rows

Looping Through a 2D Array

- To investigate every single item

```
>>> data = np.array([[1,2,3],[4,5,6]])  
>>> for i in range(2):  
    for j in range(3):  
        print(data[i][j])
```

What if I don't
know how many
rows and columns
in the array?

1
2
3
4
5
6

Looping Through a 2D Array

```
>>> data = np.array([[1,2,3],[4,5,6]])
>>> for i in range(data.shape[0]):
    for j in range(data.shape[1]):
        print(data[i][j])
```

The member
“shape” tells you
the dimensions of
the array



1
2
3
4
5
6

```
>>> data.shape
(2, 3)
>>> data.shape[0]
2
>>> data.shape[1]
3
```

Image Processing

We have all these photo apps

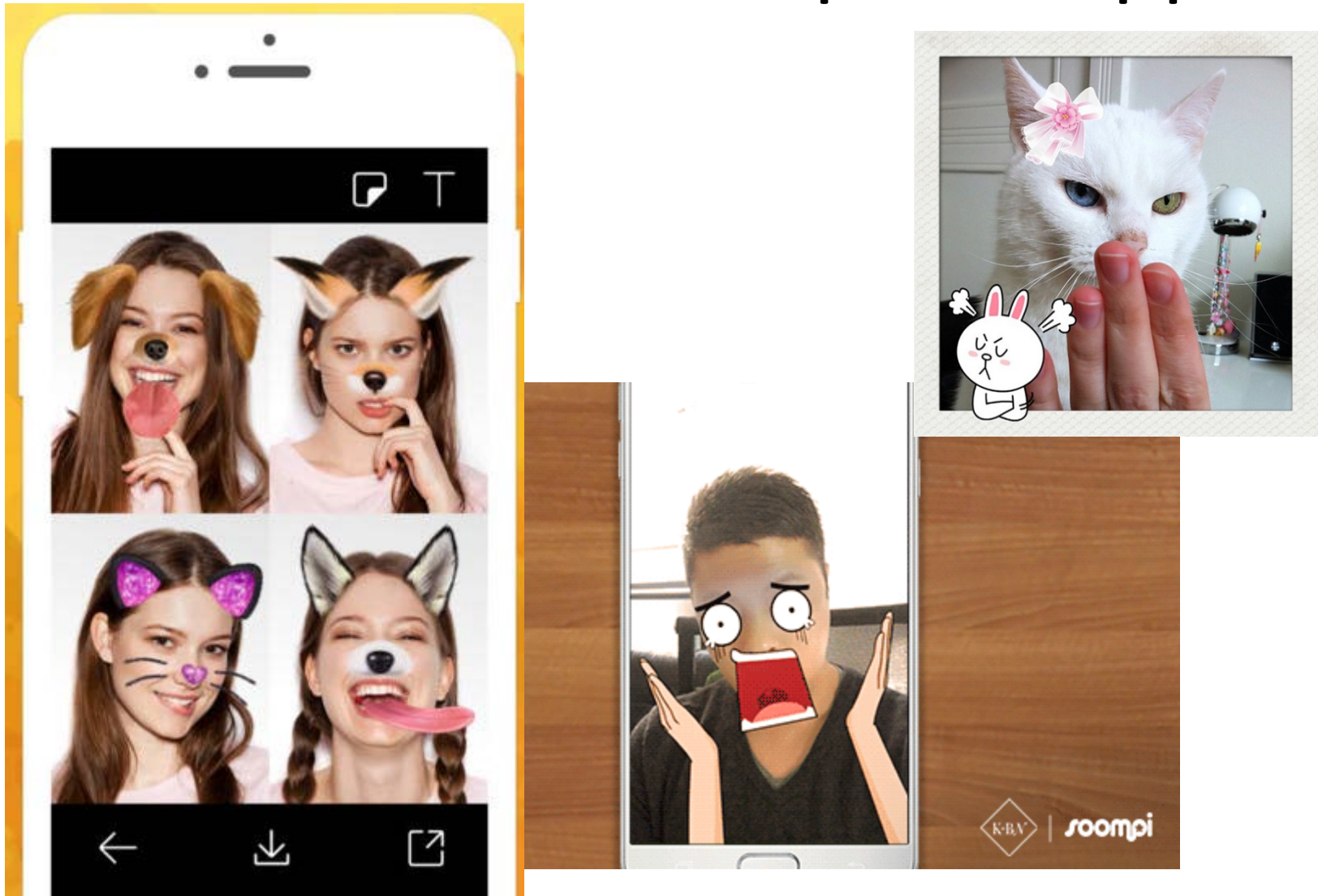


Image Processing

- To load an image, you can use the package 'misc' in 'scipy'

```
from scipy import misc  
import matplotlib.pyplot as plt
```

```
cat_pic = misc.imread('cute cat.jpg')
```

```
plt.imshow(cat_pic)  
plt.show()
```

```
>>> type(cat_pic)  
<class 'numpy.ndarray'>  
>>> cat_pic.shape  
(600, 600, 3)
```

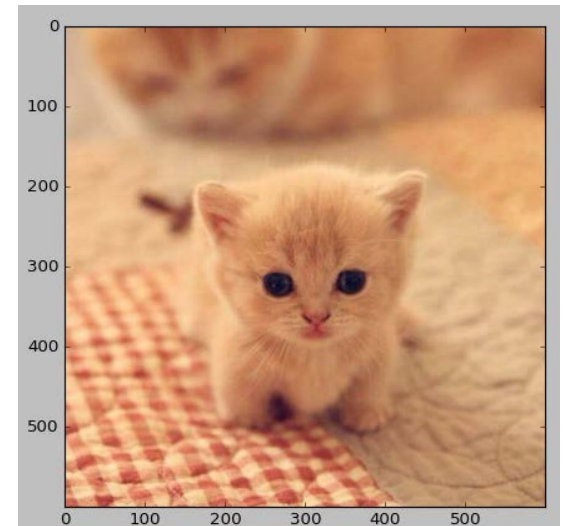


Image Processing

- 600 x 600 pixel
 - And each pixel has three values of R, G and B
 - [R, G, B]

```
>>> type(cat_pic)
<class 'numpy.ndarray'>
>>> cat_pic.shape
(600, 600, 3)
```

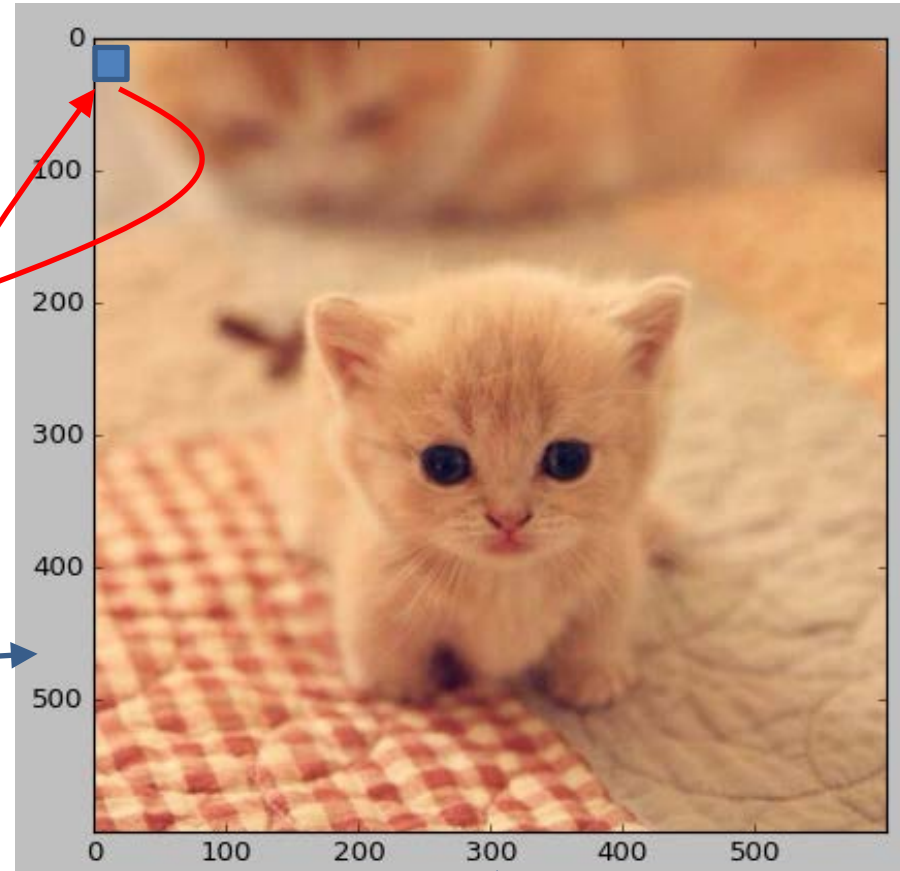


Image Processing

- 600 x 600 pixel
 - [R, G, B]
 - $0 \leq R, G, B \leq 255$

```
>>> type(cat_pic)
<class 'numpy.ndarray'>
>>> cat_pic.shape
(600, 600, 3)
```

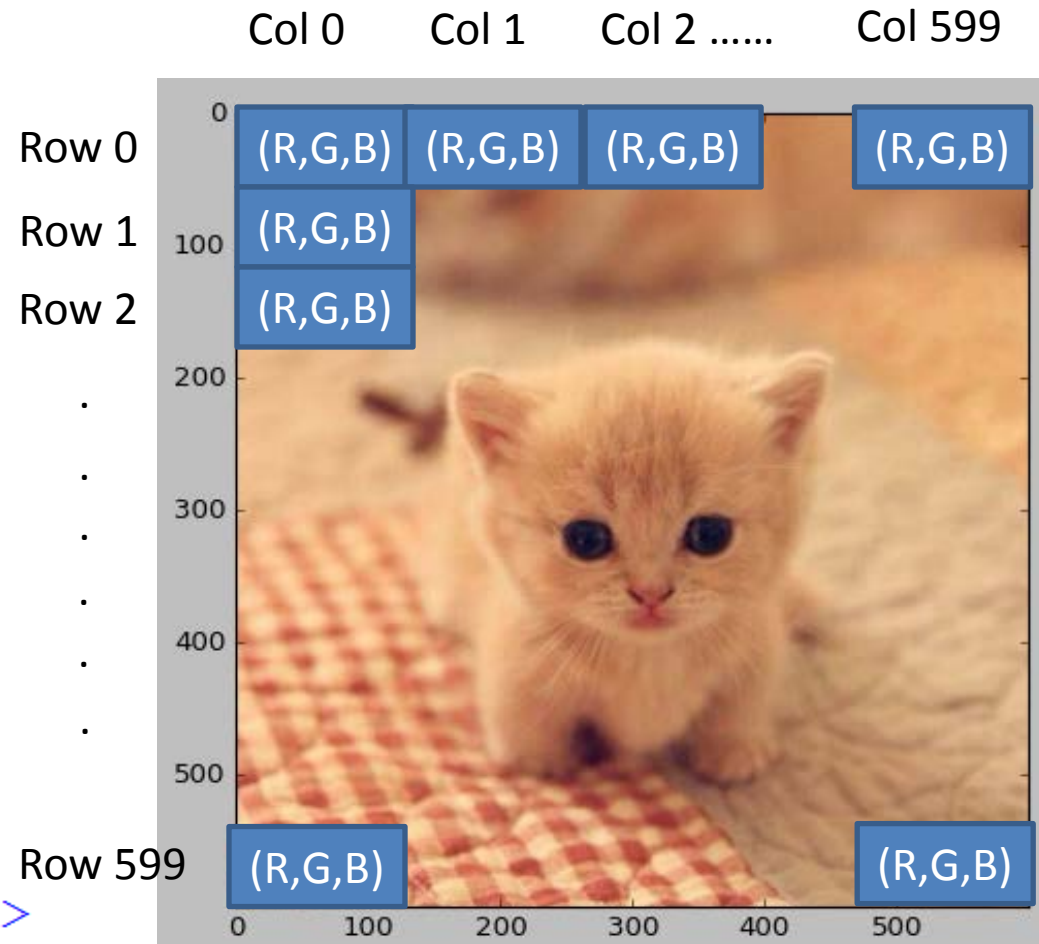
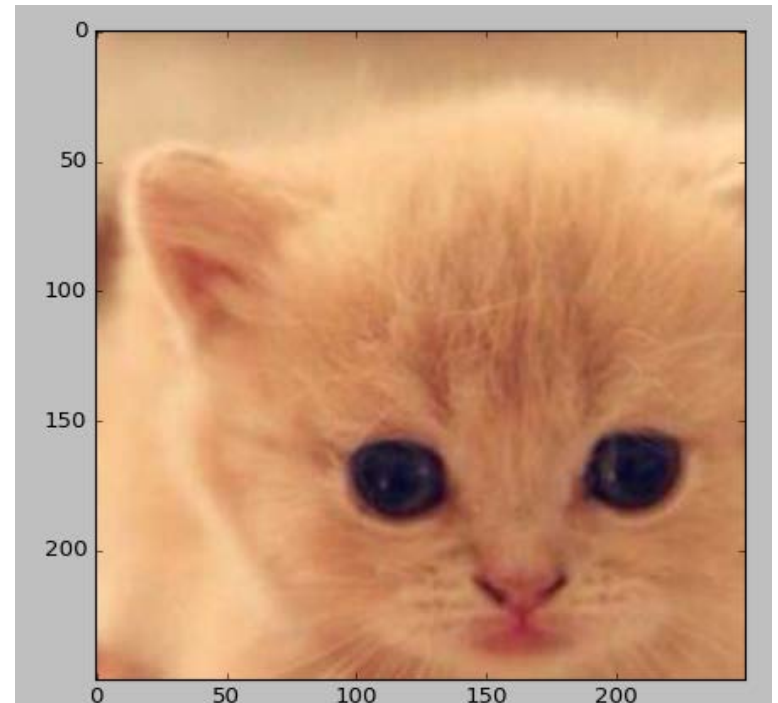


Image Processing

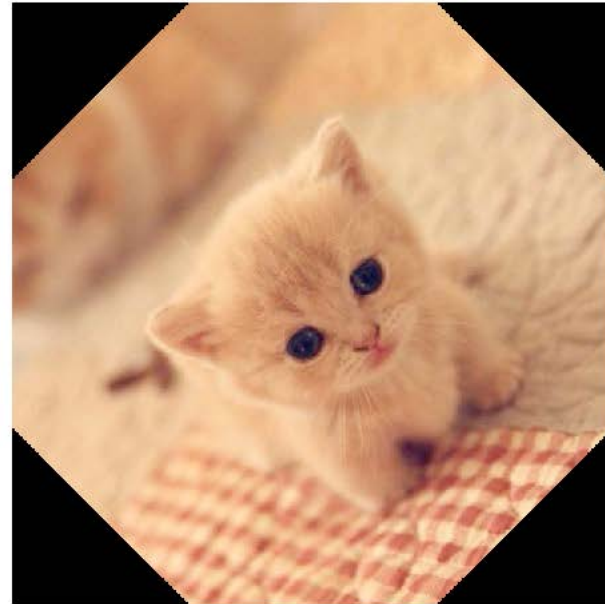
- Remember sub-matrix, string slicing, etc.?

```
>>> cat_pic2 = cat_pic[150:400,150:400,:]  
>>> plt.imshow(cat_pic2)  
<matplotlib.image.AxesImage object at 0x0000006940919F28>  
>>> plt.show()
```



Rotating an Image

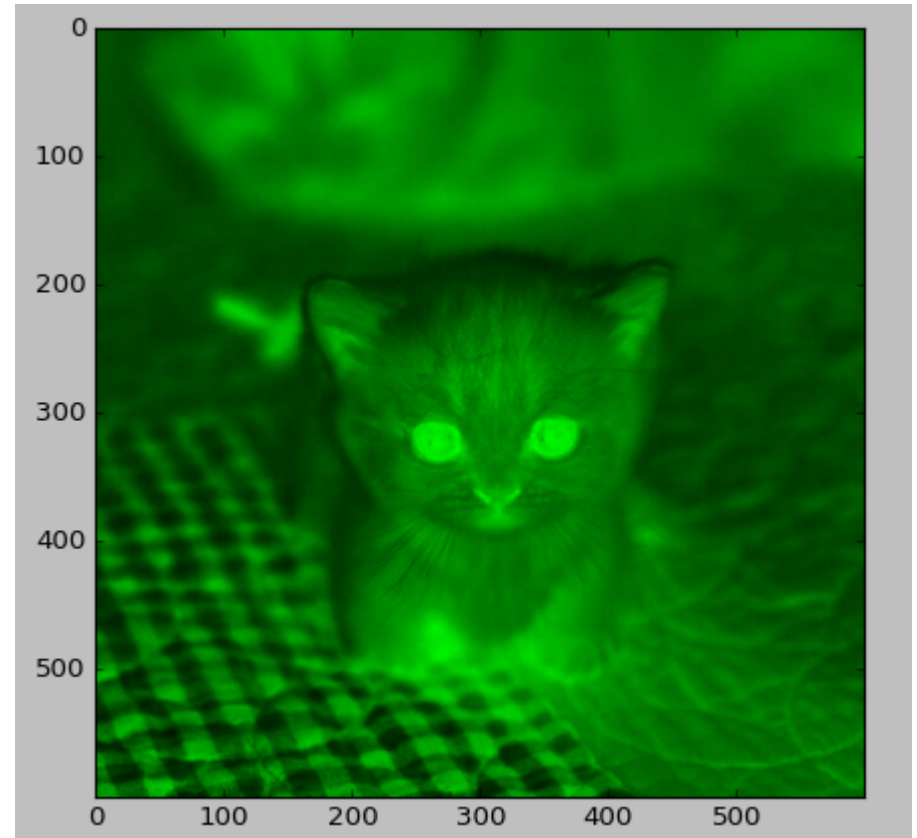
```
cat_pic = misc.imread('cute cat.jpg')  
rotate_cat = ndimage.rotate(cat_pic, 45)  
rotate_face_noreshape = ndimage.rotate(cat_pic, 45, reshape=False)
```



Broadcasting

```
>>> cat_pic2 = cat_pic * [0, 1.0, 0]
>>> plt.imshow(cat_pic2)
<matplotlib.image.AxesImage object at 0x0000006940980860>
>>> plt.show()
```

- every pixel multiply by
 - $[R, G, B] \times [0, 1, 0] =$
 - $[R \times 0, G \times 1, B \times 0]$
 - $[0, G, 0]$



Array Broadcasting

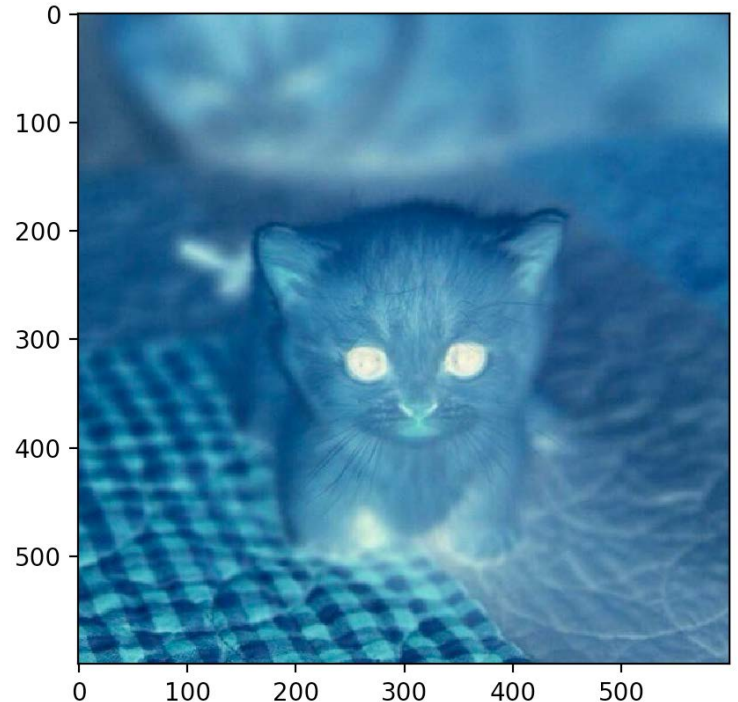
```
>>> a = np.array([1,2,3,4,5])
>>> a + 1  ←———— “Broadcasting”
array([2, 3, 4, 5, 6])
>>> a * 3  ←———— Different from LIST
array([ 3,  6,  9, 12, 15])
>>> a > 5
array([False, False, False, False, False], dtype=bool)
```

← Create another array with the Boolean results

Negative Image

```
from scipy import misc, ndimage  
import matplotlib.pyplot as plt  
import numpy as np
```

```
cat_pic = misc.imread('cute cat.jpg')  
cat_neg = 255 - cat_pic  
plt.imshow(cat_neg)  
plt.show()
```

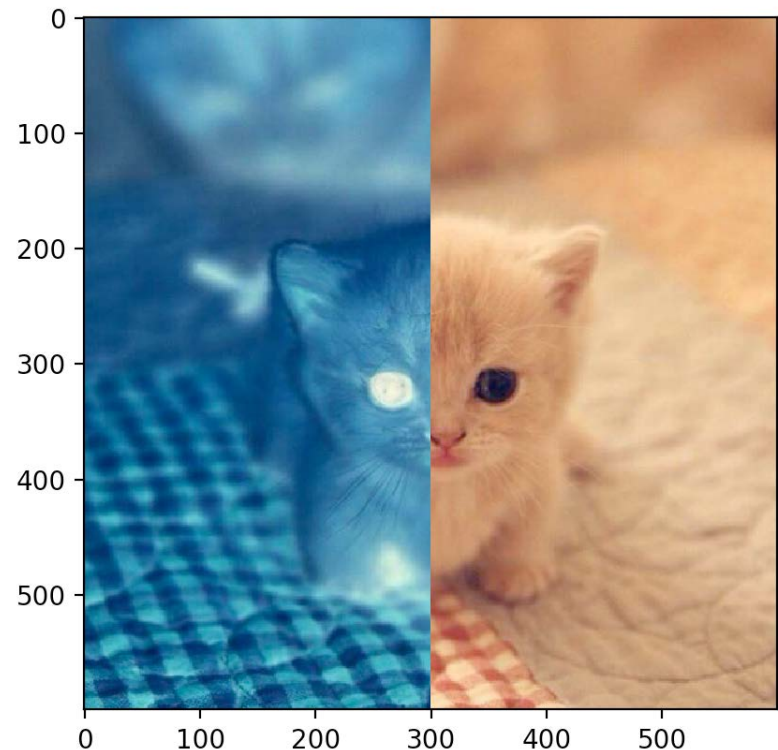



```
from scipy import misc, ndimage
import matplotlib.pyplot as plt
```

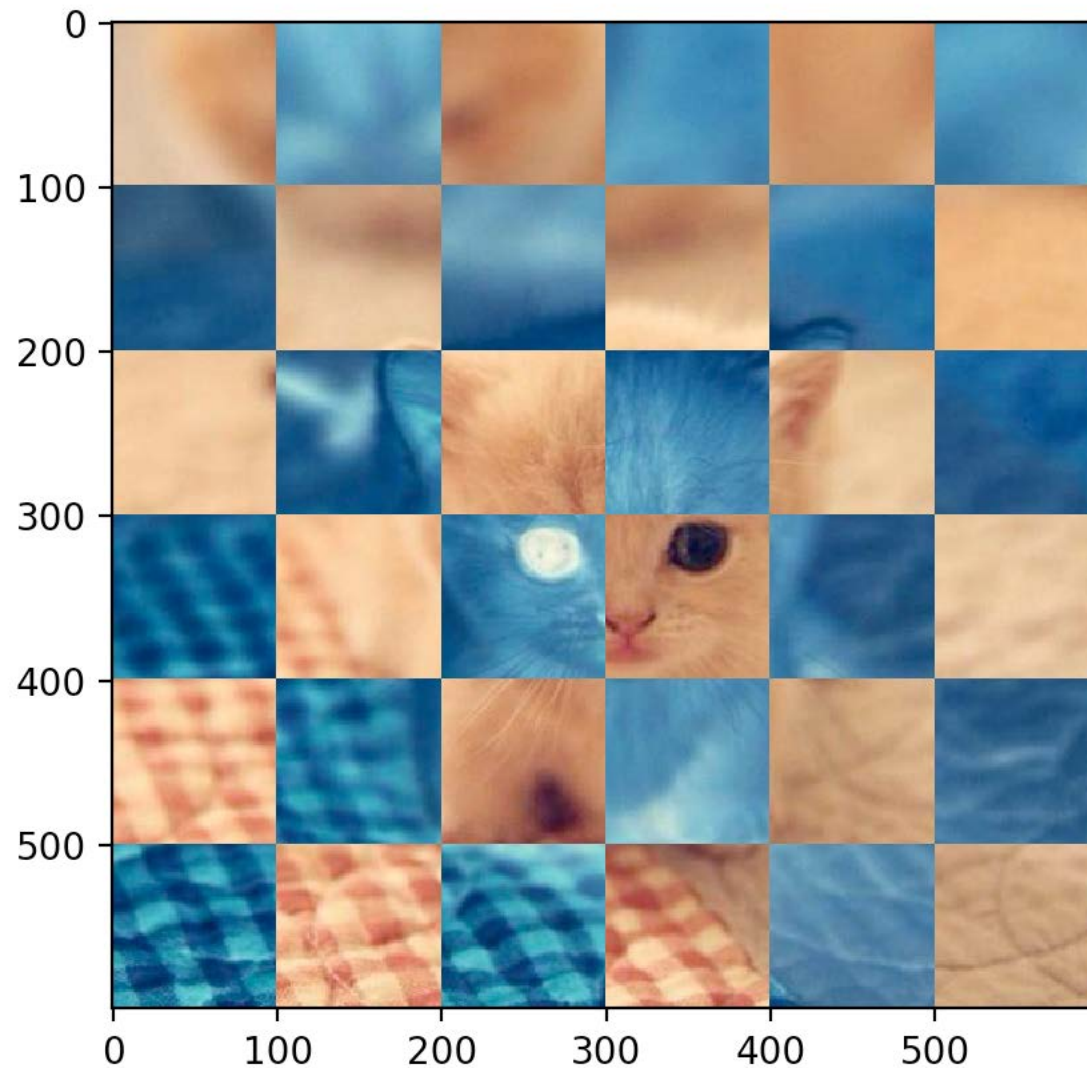
```
cat_pic = misc.imread('cute cat.jpg')
for i in range(cat_pic.shape[0]):
    for j in range(cat_pic.shape[1]):
        if j < cat_pic.shape[0]/2:
            cat_pic[i][j] = 255 - cat_pic[i][j]
```

```
plt.imshow(cat_pic)
plt.show()
```

2D Array looping



How to....?



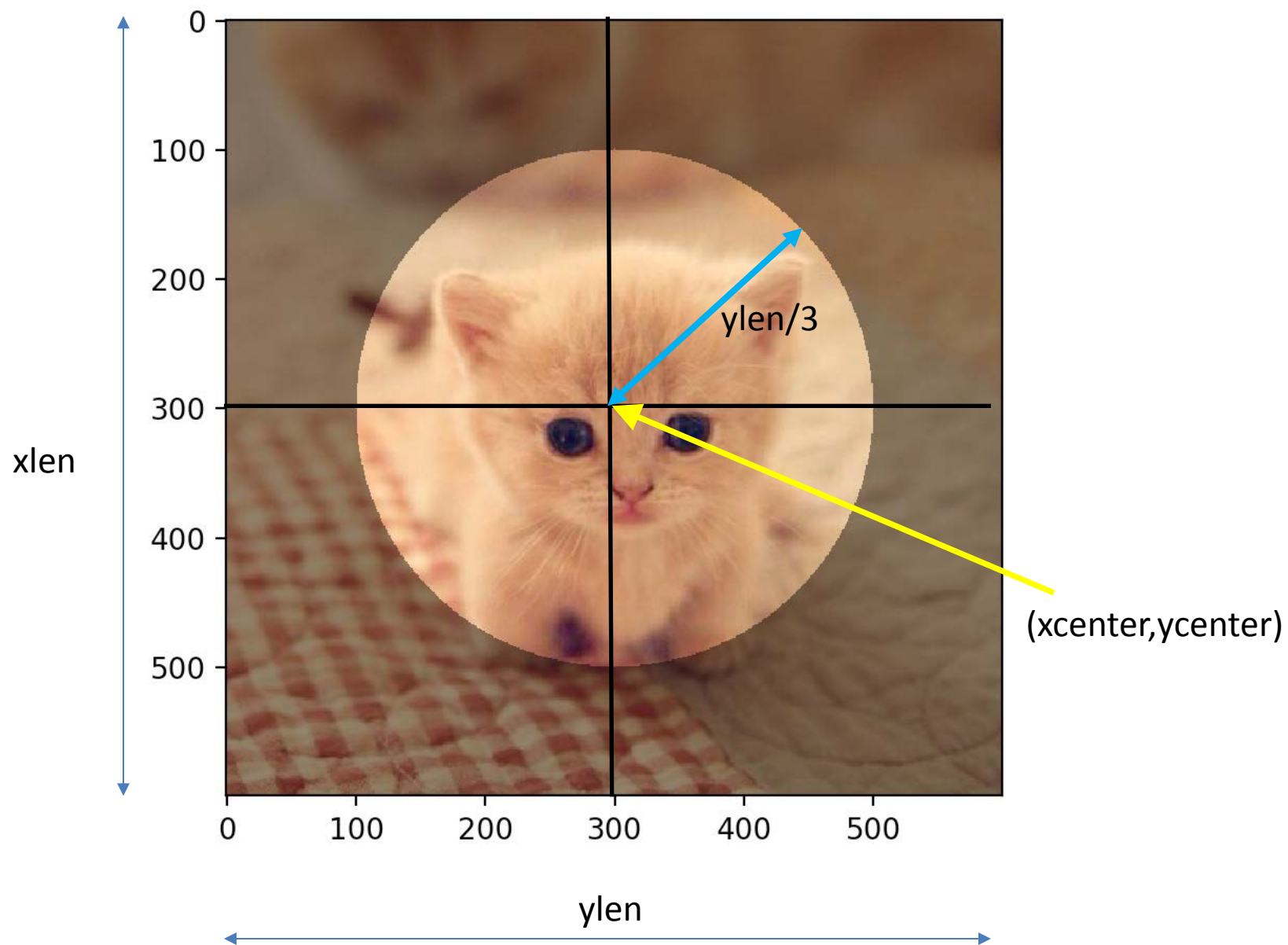
Making a Mask

```
from scipy import misc
import matplotlib.pyplot as plt
import numpy as np

cat_pic = misc.imread('cute cat.jpg')
xlen, ylen = cat_pic.shape[0], cat_pic.shape[1]
xcenter, ycenter = xlen/2, ylen/2

cat_pic2 = np.array(cat_pic)
for i in range(xlen):
    for j in range(ylen):
        if (i-xcenter)**2 + (j-ycenter)**2 > (xlen/3)**2:
            cat_pic2[i][j] = cat_pic2[i][j]/2

plt.imshow(cat_pic2)
plt.show()
```



Making a Mask

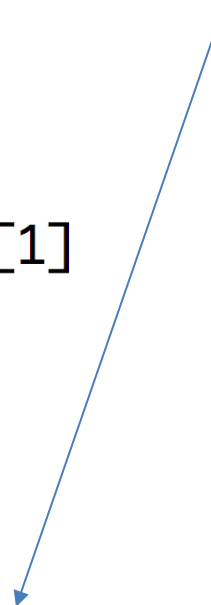
```
from scipy import misc
import matplotlib.pyplot as plt
import numpy as np
```

```
cat_pic = misc.imread('cute cat.jpg')
xlen, ylen = cat_pic.shape[0], cat_pic.shape[1]
xcenter, ycenter = xlen/2, ylen/2
```

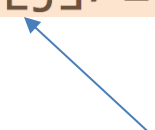
```
cat_pic2 = np.array(cat_pic)
for i in range(xlen):
    for j in range(ylen):
        if (i-xcenter)**2 + (j-ycenter)**2 > (xlen/3)**2:
            cat_pic2[i][j] = cat_pic2[i][j]/2
```

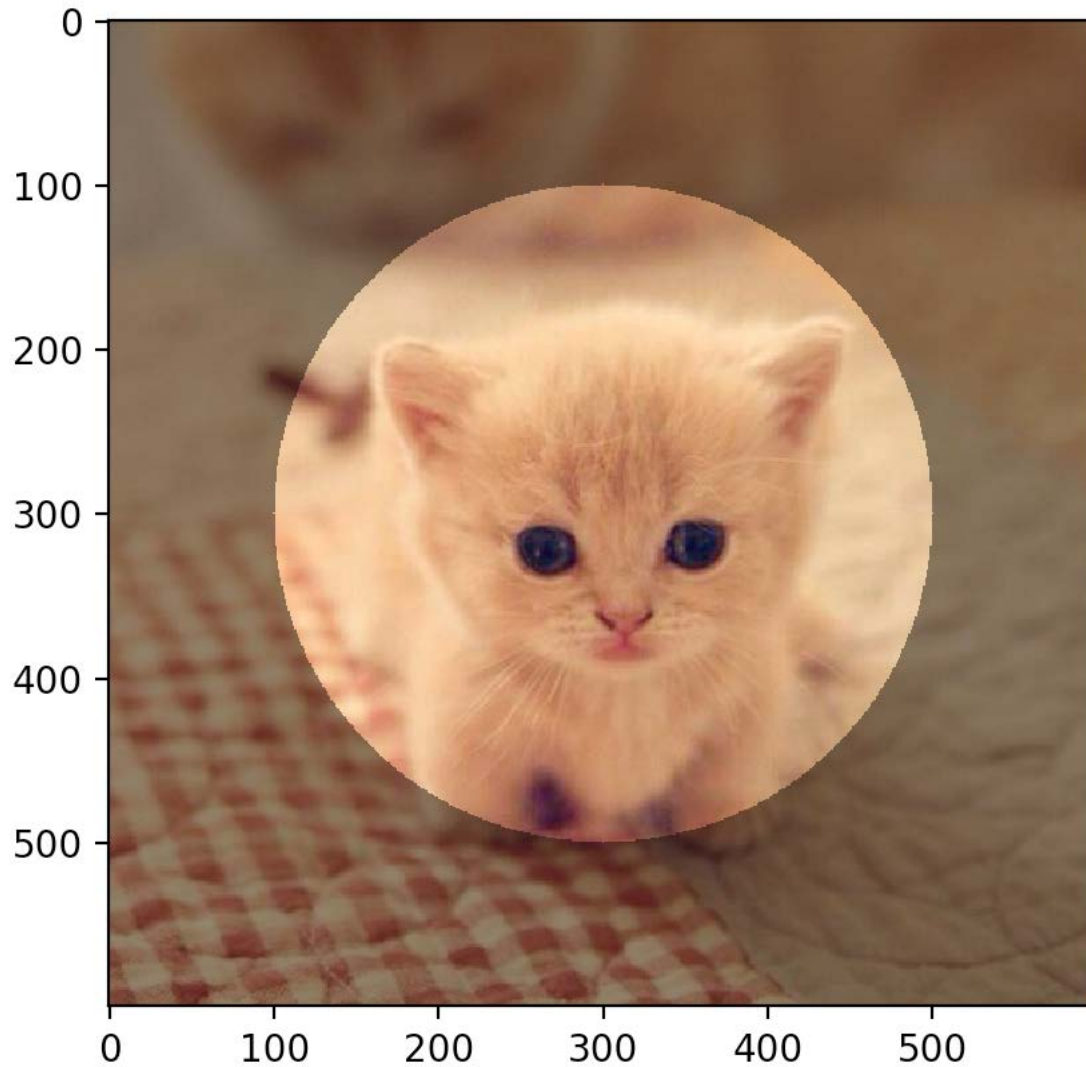
```
plt.imshow(cat_pic2)
plt.show()
```

If the pixel is out
of the circle



Each color of the
pixel is divided by 2





Your
picture
array

- Any time you want to save an image:
`misc.imshow('file name.png', cat_pic2)`

Applying Filters

```
from scipy import misc, ndimage
import matplotlib.pyplot as plt
import numpy as np
```

```
cat_pic = misc.imread('cute cat.jpg')
blurred_cat_pic = ndimage.gaussian_filter(
    cat_pic, sigma=(9,9,1))
```

```
plt.imshow(blurred_cat_pic)
plt.show()
```

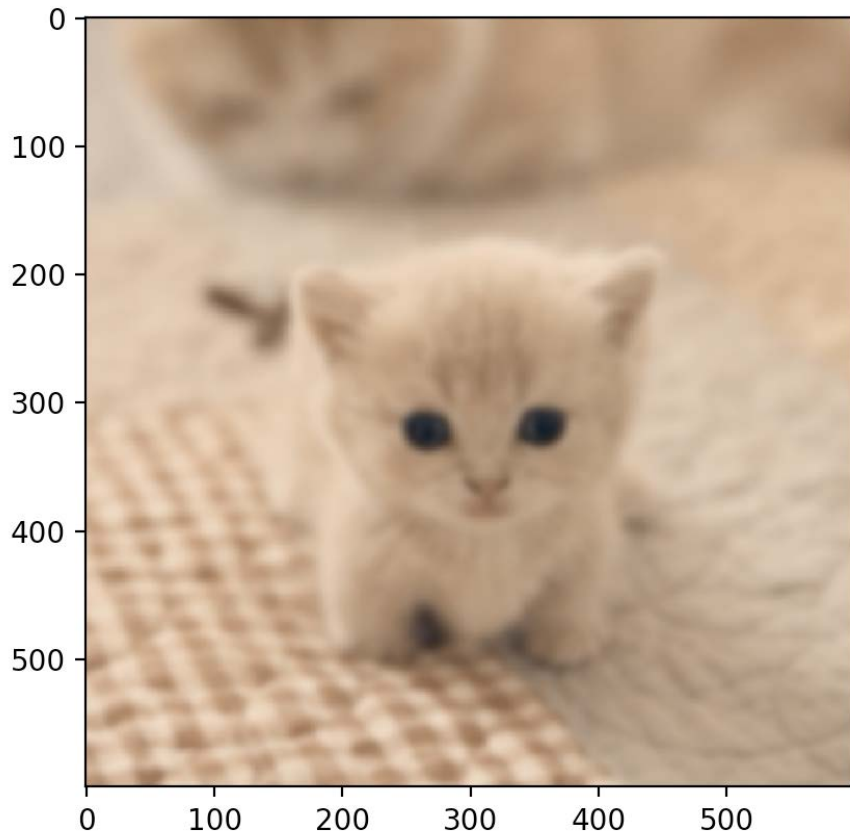
Blending on x

Blending on y

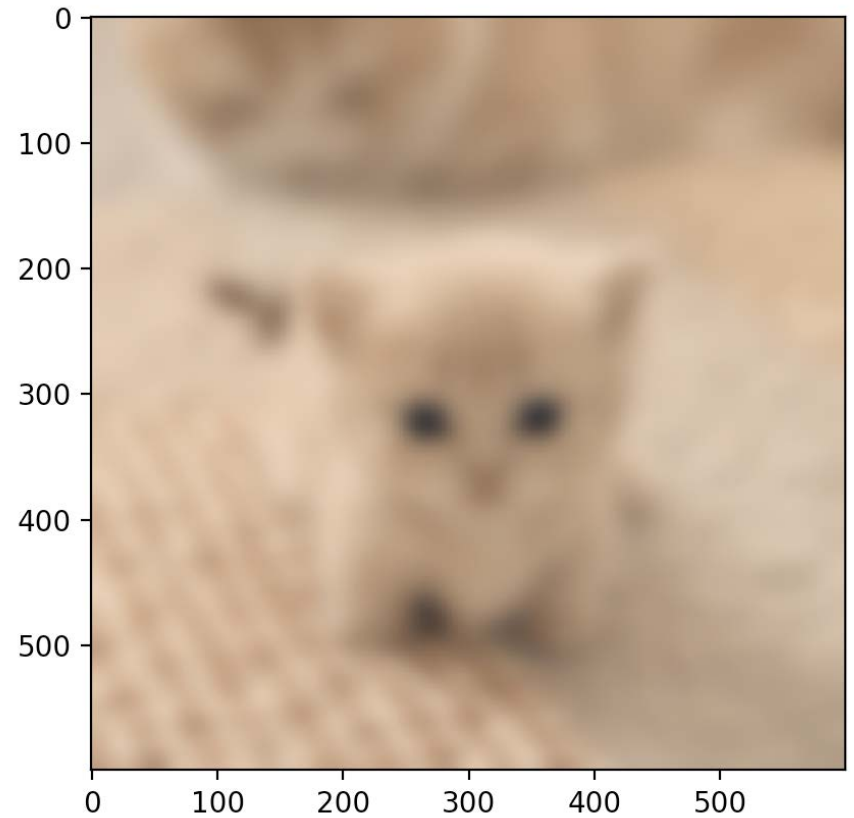
NO Blending on
colors

Different Sigma

- $\sigma = (3,3,1)$



- $\sigma = (9,9,1)$



Edge Detection

- Anyhow generate an image

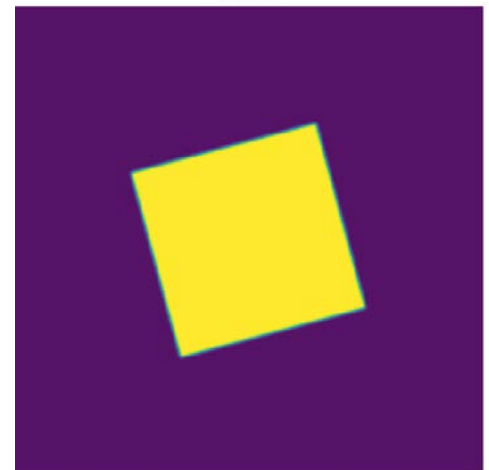
```
img = np.zeros((256, 256))
```

```
img[64:-64, 64:-64] = 1
```

```
img = ndimage.rotate(img, 15, mode='constant')
```

```
img = ndimage.gaussian_filter(img, 1)
```

square

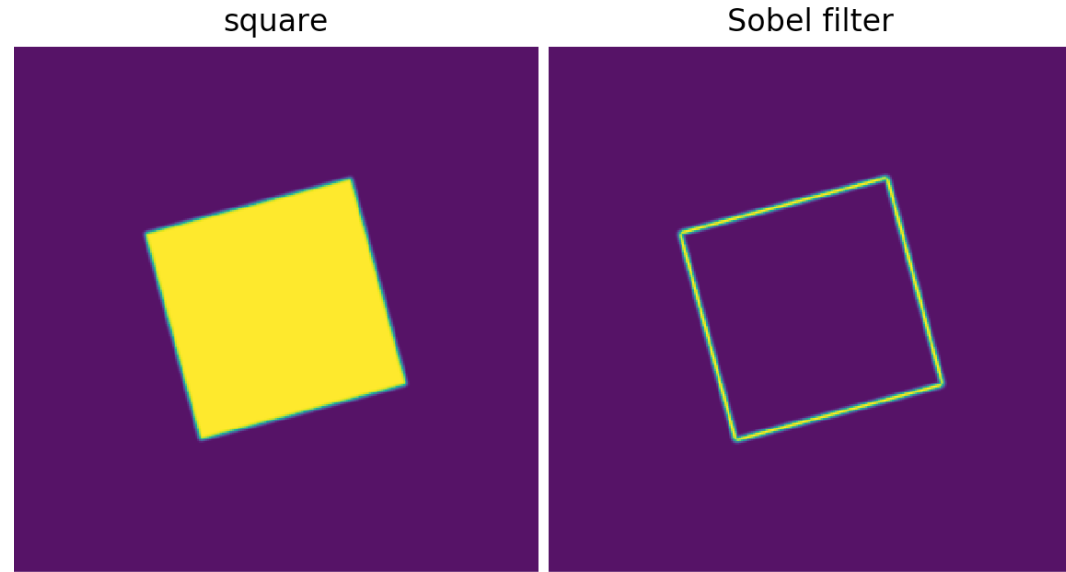


Applying Sobel filter to the image

```
sx = ndimage.sobel(img, axis=0, mode='constant')  
sy = ndimage.sobel(img, axis=1, mode='constant')  
sob = np.hypot(sx, sy)
```

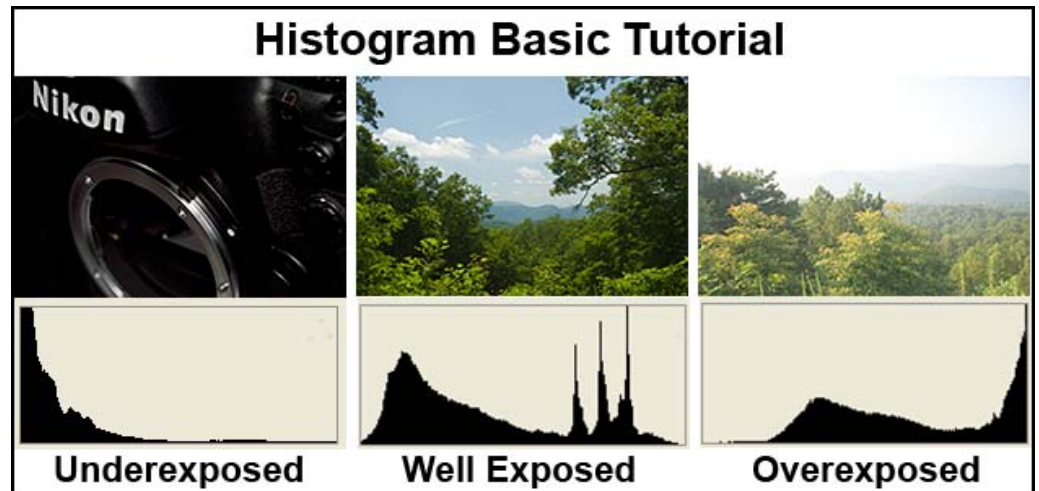
```
plt.subplot(121)  
plt.imshow(img)  
plt.axis('off')  
plt.title('square')
```

```
plt.subplot(122)  
plt.imshow(sob)  
plt.axis('off')  
plt.title('Sobel filter')
```



More in Numpy and Scipy

- Fourier Transform
- Uniform filter
- Histogram
- Laplace... etc



- More on
 - <https://docs.scipy.org/doc/scipy-0.16.1/reference/ndimage.html>

PILLOW

A Fork in PIL

PILLOW is a fork of PIL

- PIL stands for Python Imaging Library

```
from PIL import Image
```

```
pic = Image.open('my flight delay.JPG')
```

```
pic.show()
```

```
|
```

Let's get the secret out

STD	FLIGHT	VIA/TO	GATE	STATUS
14:05	DZ6237		43	
15:25	HU7663	Nanjing	15	Last Call
15:25	1H9923	C Nantong	60	ed
15:30	ZH9809	Wuxi	59	Last Call
15:40	HU7263	Hohhot	16	Delayed
15:45		(Huangshan	39	Gate
15:55	HU7255	Zhengzhou	17	W
15:55	HU7326	Taiyuan	46	ed
15:55	3076	Wuxi	78	ed
16:00	314	Beijing	14	ed
16:00	672	Tianjin	26	ed
16:05	010	1	32	ed
16:05			61A	ed
16:20	1454		71	ed
16:25	053	Beijing	62B	ed
16:30	ZH9877	Hangzhou	54	ed
16:30	350 CZ97	Shi	44	ed
16:35	3U8784	Chongqing	48	
16:35	KN5851	Nanyuan	62A	elled
16:35	CA4846	Yantai	50A	ed
16:40	3509	Nanchang	61B	ed
16:50	CZ8250	Chi	29	ed

尊敬的旅客请注意Page: 1/3

STD	FLIGHT	VIA/TO	GATE	STATUS
16:55	3U Chengdu		53	Delayed
16:55	87 MF141	iao	21	Delayed
16:55	3915 KY	Qingdao	58	Delayed
17:00	ZH 1		27	Delayed
17:00	48 ZH434	Chongqing	61A	Delayed
17:05	CZ6473	Bijie	41	Delayed
17:05	44 MF156	IN	50B	Delayed
17:10	10 MF191	Shenyang	30	Delayed
17:15	ZH Hefei		32	Delayed
17:20	114 ZH431	Chengdu	35	Delayed
17:20	MU5751		36	Delayed
17:25	HU7708	Beijing	20	Delayed
17:25	HU7715	Jinan	25	Delayed
17:25	BK2860	Tianjin	54	Delayed
17:30	CZ6913	Nanjing	34	Cancelled
17:30	FM9334	iao	51	Delayed
17:30	CZ8649	Xining	28	Delayed
17:35	63 MF131	Hangzhou	38	Cancelled
17:35	ZH Nanjing		77	Cancelled
17:40		Chongqing	62A	Delayed
17:40	91 MF101	Beijing	63	Delayed
17:40	HU7067	Xuzhou	44	Delayed

尊敬的旅客请注意:受本场天气影响, 部 Page: 2/3

STD	FLIGHT	VIA/TO	GATE	STATUS
17:45		(Hangzhou	59	e Weather Co
17:45	TV9832	igdu/linzhi	24	e Weather Co
17:45	Z9334	ibo/Qingdao	41	e Weather Co
17:50	A3272	ighai Hongqiao	42	Delayed
17:50	9C8776	ighai Hongqiao	49	e Weather Co
17:55	HU7763	Nanjing	25	e Weather Co
17:55	Z9215	Nanjing	46	e Weather Co
18:00		(Beijing	14	e Weather Co
18:00		ighai Pudong	76	e Weather Co
18:05		Hangzhou	61A	e Weather Co
18:05		gzhou/Shenyang	52	e Weather Co
18:10	021	Chongqing	18	e Weather Co
18:10	17 CA370	Chengdu	78	e Weather Co
18:15		gzhou/Hohhot	34	e Weather Co
18:15	ZH9171	hot/Haila'er	72	e Weather Co
18:15		hen/Shanghai Hi	17	e Weather Co
18:20	CZ6328	Dalian	33	e Weather Co
18:20	1912 MU3	ing/Nanyuan	43	e Weather Co
18:25	ZH9111	Yichun/Beijing	77	
18:30		Guiyang	22	e Weather Co
18:30	MU5354	ighai Hongqiao		Cancelled
18:35	CZ Chengdu		63	e Weather Co

心等候, 不便之处敬请谅解!

Page: 3/3

```
from PIL import Image
from PIL.ExifTags import TAGS, GPSTAGS

pic = Image.open('my flight delay.JPG')

def get_exif_data(image):
    exif_data = {}
    info = image._getexif()
    if info:
        for tag, value in info.items():
            decoded = TAGS.get(tag, tag)
            if decoded == "GPSInfo":
                gps_data = {}
                for t in value:
                    sub_decoded = GPSTAGS.get(t, t)
                    gps_data[sub_decoded] = value[t]
                exif_data[decoded] = gps_data
            else:
                exif_data[decoded] = value

    return exif_data

print(get_exif_data(pic) ['GPSInfo'])
pic.show()
```

PILLOW

- Cannot escape!

```
--  
{'GPSLatitudeRef': 'N', 'GPSLatitude': ((22, 1),  
(38, 1), (1484, 100)), 'GPSLongitudeRef': 'E', 'G  
PSLongitude': ((113, 1), (48, 1), (2726, 100)), 'G  
PSAltitudeRef': b'\x00', 'GPSAltitude': (2761, 2  
25), 'GPSTimeStamp': ((10, 1), (34, 1), (1420, 10  
0)), 'GPSSpeedRef': 'K', 'GPSSpeed': (0, 1), 'GPS  
ImgDirectionRef': 'T', 'GPSImgDirection': (11511,  
542), 'GPSDestBearingRef': 'T', 'GPSDestBearing':  
(11511, 542), 'GPSDateStamp': '2017:07:17', 'GPSH  
PositioningError': (1414, 1)}
```

PILLOW

```
from PIL import Image
from PIL import ImageFilter
pic = Image.open('cute cat.jpg')

pic.show()

blurred_pic = pic.filter(ImageFilter.BLUR)
blurred_pic.show()

sharpen_pic = pic.filter(ImageFilter.SHARPEN)
sharpen_pic.show()
```




Original



Blurred



Sharpen

Copy And Paste

```
from PIL import Image
```

```
pic = Image.open('cute cat.JPG')  
part = pic.crop((200, 200, 400, 400))
```

Copy (crop)
the part of
the picture

```
pic.paste(part, (0, 400))  
pic.show()
```

Paste it on
the position
(0,400)



```
from PIL import Image, ImageDraw, ImageFont

pic = Image.open('cute cat.JPG')

draw = ImageDraw.Draw(pic)
draw.ellipse((20, 30, 160, 120), fill='blue')
draw.text((60, 65), 'IT1007 Cat', fill = 'gray')

pic.show()
```

|



Other operations

- resize
- rotation/flipping
- transpose
- Drawing shapes
- etc. etc..

<https://pillow.readthedocs.io/en/4.2.x/>

- `Image` Module
- `ImageChops` ("Channel Operations") Module
- `ImageColor` Module
- `ImageCms` Module
- `ImageDraw` Module
- `ImageEnhance` Module
- `ImageFile` Module
- `ImageFilter` Module
- `ImageFont` Module
- `ImageGrab` Module (macOS and Windows only)
- `ImageMath` Module
- `ImageMorph` Module
- `ImageOps` Module
- `ImagePalette` Module
- `ImagePath` Module
- `ImageQt` Module
- `ImageSequence` Module
- `ImageStat` Module
- `ImageTk` Module
- `ImageWin` Module (Windows-only)
- `ExifTags` Module
- `TiffTags` Module
- `PSDraw` Module
- `PixelAccess` Class
- `PyAccess` Module

Other Than Scipy and Numpy

- OpenCV
- skimage
 - scikit-image

