COMP 204
Introduction to image analysis with scikit-image (part one)

Yue Li, based on slides from Mathieu Blanchette, Christopher J.F. Cameron and Carlos G. Oliver
Image processing and analysis in Python

Goal: Process and analyze digital images.

▶ Very useful for processing microscopy images, medical imaging, etc.
▶ Closely linked with machine learning for image analysis

**scikit-image module or (skimage)**
▶ image processing module in Python
▶ holds a wide library of image processing algorithms: filters, transforms, point detection
▶ API
  ▶ [http://scikit-image.org/docs/dev/api/api.html](http://scikit-image.org/docs/dev/api/api.html)
Our mascot for today
Reading an image into memory

We’ll start with an example image using the `io module`

- basic I/O submodule of scikit-image
- API

```
import skimage.io as io
import matplotlib.pyplot as plt

# read image into memory
image = io.imread('monkey.jpg')

# show the image
plt.imshow(image)
plt.show()

# write image to disk
io.imsave('monkey_copy.jpg', image)
```
What’s an image in Python?

An image is stored as a NumPy ndarray (n-dimensional array).

- ndarrays are easier and more efficient than using 2-dimensional lists as we’ve seen before.

A color image with $R$ rows and $C$ columns is

- represented as a 3-dimensional ndarray of dimensions $R \times C \times 3$
- element at position $(i,j)$ of the array corresponds to the RGB value at row $i$ and column $j$
- each pixel is represented by 3 numbers, each between 0 and 255: Red, Green, Blue
import skimage.io as io
import skimage.color as color
import matplotlib.pyplot as plt

# read image into memory
image = io.imread("monkey.jpg")

n_row, n_col, n_colours = image.shape
print(n_row, n_col, n_colours)  # prints (1362, 2048, 3)

# print pixel value at row 156 and column 293
pixel = image[156, 292]
print(pixel)  # prints [112 158 147] = green—bluish color

# change pixel value to red
image[156, 292] = (255, 0, 0)

# Create a pink rectangle between rows 1000—1200
# and column 500—900
for i in range(1000, 1200):
    for j in range(500, 900):
        image[i, j] = (255, 0, 255)

plt.imshow(image)
plt.show()
io.imsave("monkey_bar.jpg", image)
import skimage.io as io
import skimage.color as color
import matplotlib.pyplot as plt

# read image into memory
image = io.imread("monkey.jpg")

# Create the negative of an image
for i in range(image.shape[0]):
    for j in range(image.shape[1]):
        for c in range(3):
            image[i,j,c] = 255 - image[i,j,c]

# we could just have written:
#image = 255 - image

plt.imshow(image)
plt.show()
io.imsave("monkey_negative.jpg",image)
Flipping the image horizontally

```python
import skimage.io as io
import skimage.color as color
import matplotlib.pyplot as plt

# read image into memory
image = io.imread("monkey.jpg")
row, col, colours = image.shape

# Flip the image horizontally
for i in range(0, row):
    for j in range(0, int(col/2)):
        image[i, j] = image[i, col-j-1]

plt.imshow(image)
plt.show()
io.imsave("monkey_flipped_wrong.jpg", image)
```
```python
import skimage.io as io
import skimage.color as color
import matplotlib.pyplot as plt

# read image into memory
image = io.imread("monkey.jpg")
n_row, n_col, colours = image.shape

# Flip the image horizontally
for i in range(0, n_row):
    for j in range(0, int(n_col/2)):
        t = image[i, j].copy()
        image[i, j] = image[i, n_col-j-1]
        image[i, n_col-j-1] = t

plt.imshow(image)
plt.show()
io.imsave("monkey_flipped_right.jpg", image)
```
Combining images

Since images are just arrays of numbers, we can easily combine them.
Example: Create an image that is the average of monkey and tiger.
Combining images

```python
import skimage.io as io
import matplotlib.pyplot as plt
import numpy as np
from skimage.transform import resize

monkey = io.imread("monkey.jpg")
tiger = io.imread("tiger.jpg")

# resize images to 500x1000 pixels
monkey_resized = resize(monkey, (500, 1000))
tiger_resized = resize(tiger, (500, 1000))

combined = monkey_resized/2 + tiger_resized/2

# this is equivalent to:
# for i in range(500):
#     for j in range(1000):
#         for c in range(3):
#             combined[i,j,c]=monkey[i,j,c]/2 + tiger[int(i/monkey_row*tiger_row), int(j/monkey_col*tiger_col), c]/2

plt.imshow(combined)
plt.show()
io.imsave("combined.jpg", combined)
```
Combining images
import skimage.io as io
import skimage.color as color
import matplotlib.pyplot as plt

image = io.imread("monkey.jpg")

# create three copies of the image
red = image.copy()
blue = image.copy()
green = image.copy()

red[:, :, (1, 2)] = 0  # NumPy indexing
green[:, :, (0, 2)] = 0
blue[:, :, (0, 1)] = 0

io.imsave("monkey_red.jpg", red)
iio.imsave("monkey_green.jpg", green)
io.imsave("monkey_blue.jpg", blue)
red intensity

green intensity

blue intensity
import skimage.io as io
import skimage.color as color
import matplotlib.pyplot as plt
import numpy as np

image = io.imread("monkey.jpg")
n_row, n_col, colors = image.shape

# create a blank image
new_image = np.zeros((n_row, n_col, 3), dtype=np.uint8)

# assemble a new image made of shifted colors
# blue is shifted right by 100 pixels
# green is shifted up by 100 pixels

for i in range(n_row):
    for j in range(n_col):
        new_image[i, j, 0] = image[i, j, 0]  # keep red
        if i >= 100:
            new_image[i, j, 1] = image[i - 100, j, 1]  # move green
        if j >= 100:
            new_image[i, j, 2] = image[i, j - 100, 2]  # move blue

plt.imshow(new_image)
plt.show()
io.imsave("monkey_shifted.jpg", new_image)
Grayscaling

Many image processing algorithms assume a 2D matrix
- not an image with a third dimension of color

To bring the image into two dimensions
- we need to summarize the three colors into a single value
- this process is more commonly know as grayscaling
- where the resulting image only holds intensities of gray
  - with values between 0 and 1

skimage submodule color has useful functions for this task
- API
  http://scikit-image.org/docs/dev/api/skimage.color.html
import skimage.io as io
import skimage.color as color
import matplotlib.pyplot as plt
from skimage.color import rgb2gray

# read image into memory
image = io.imread("monkey.jpg")

# convert to grayscale
gray_image = rgb2gray(image)

print(image[0,0])  # prints [255,255,255]
print(gray_image[0,0])  # prints 1.0
plt.imshow(gray_image)
plt.show()
io.imsave("monkey_grayscale.jpg",gray_image)
import skimage.io as io
import skimage.color as color
import matplotlib.pyplot as plt
from skimage.color import rgb2gray
import numpy as np

image = io.imread("monkey.jpg")
gray_image = rgb2gray(image)

# this creates a new array, # with 1's everywhere gray_image > 0.5, and 0 elsewhere
black_and_white = np.where(gray_image > 0.5, 1, 0)

plt.imshow(black_and_white)
plt.show()
io.imsave("monkey_black_and_white.jpg", black_and_white * 255)