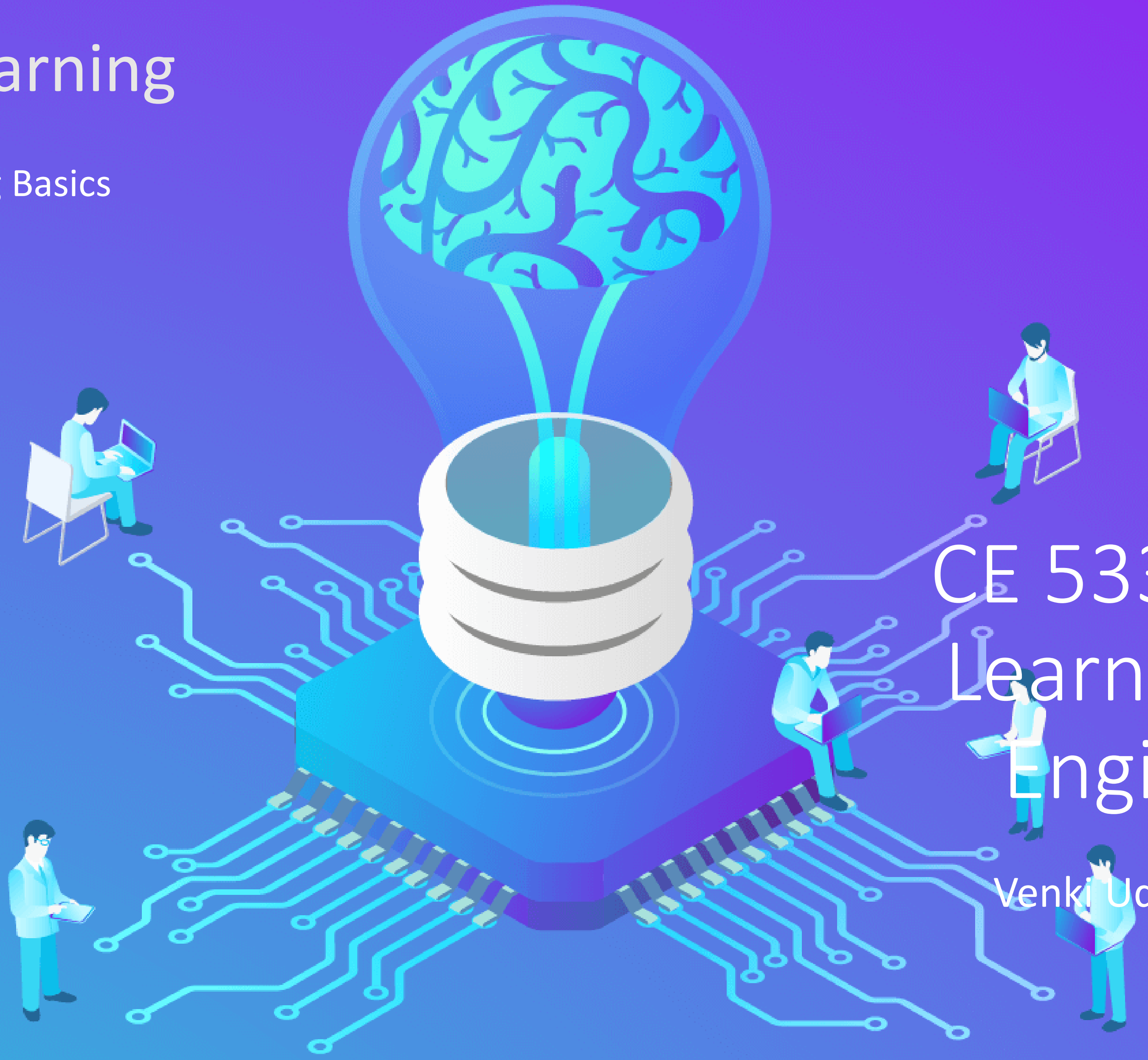
Machine Learning

Image Processing Basics



CE 5331 Machine Learning for Civil Engineers

Venki Uddameri, Ph.D. , P.E.

Recap

- What is Machine Learning
- How is it useful for Civil Engineers
- Overview of Machine Learning Methods
- Linear Regression
  - Bivariate
  - Regression interpretation
  - Multivariate
- Logistic Regression
  - Maximum likelihood estimation
  - Regularization (introduction)
- Naïve Bayesian Classifier
  - What is it
  - What makes it naïve
  - Bayes theorem
  - Prior, likelihood and posterior
- K-Nearest Neighbor
  - How does the algorithm work
  - Why is it a lazy learner
  - How to do regression and classification
- Introduction to Decision Trees
  - Fundamentals
  - Information Gain, Entropy and Gini Index
  - ID3 algorithm
  - Classification and Regression Trees (CART)
  - Multi-Adaptive Regression Splines (MARS)

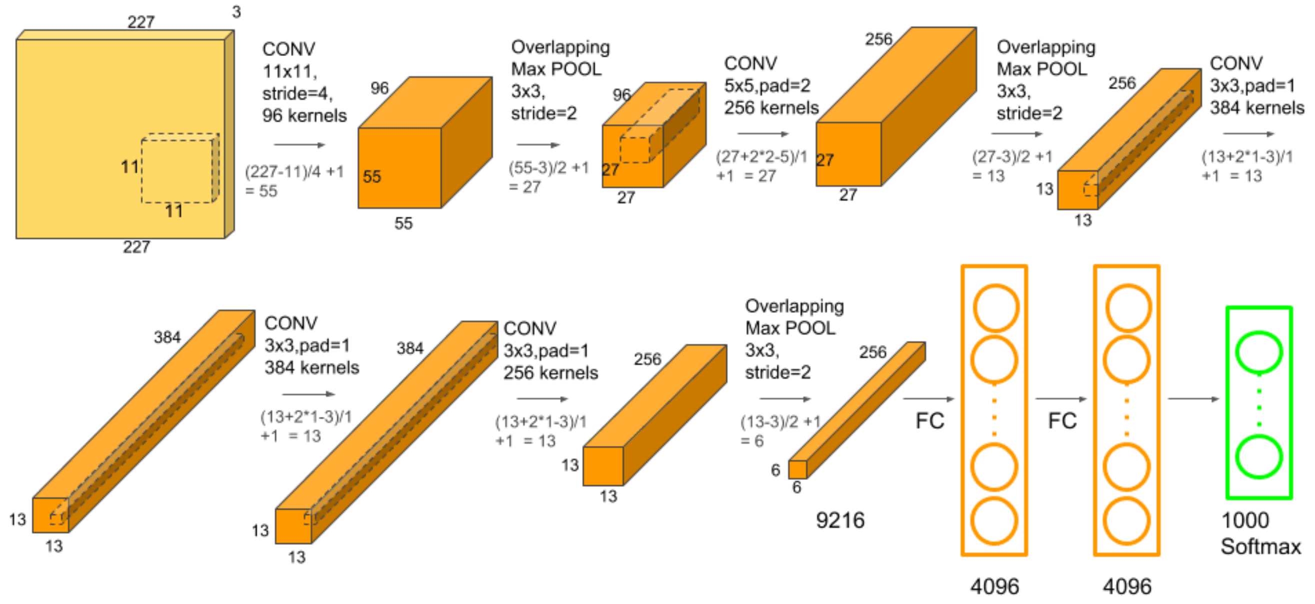
- Ensemble learners
  - Introduction
- Their benefits and drawbacks
- Simple (voting) ensemble learners
- Bagging and Pasting
- Generic bagging classifiers
- Random Forest classifiers
- Bagging Classifier
- Unsupervised classification
  - K-means clusterubg
- Perceptrons
- Multilayer Perceptrons
- Stochastic Gradient Descent
- ADAM
- Deep Neural Networks for classification and Regression

Python – Introduction
Python – Functions
Python - Pandas
Python – np, scipy, statsmodels
Python – Scikit learn – linear, metrics
Python – Matplotlib, seaborn
Python – Mixed_Naive_Bayes
Python – scikit learn neighbors module
Python – scikit learn ensemble voting
Python – scikit learn bagging classifier
Python – scikit learn RandomForestClassifer
Python - Tensorflow and Keras

R – Classification and Regression Trees using rpart
R – Drawing trees using rpart.plot
R - Multiadaptive Regression Splines (MARS) using Earth Algorithm
R - Resampling

Goals

- Image classification using Neural Networks
  - Using sequential models
  - Using Convolution Neural Networks



Why should Civil Engineers study Image Processing?

- Image sensors have become ubiquitous as well as have improved in quality
  - Cell phones!
- Obtaining images have therefore become cheap
- Other imaging data have also become inexpensive
  - Satellite Remote Sensing
  - LIDAR - Light Detection and Ranging
  - Airborne Sensors - Drones
- Image data have widespread applications in civil engineering
  - Structural Health Monitoring (Structural defects)
  - Civil Engineering properties (soil types, soil moisture, etc)
  - Environmental and Hazards communication and research
  - Material characterization (SEM, TEM and other high-resolution microscopy)

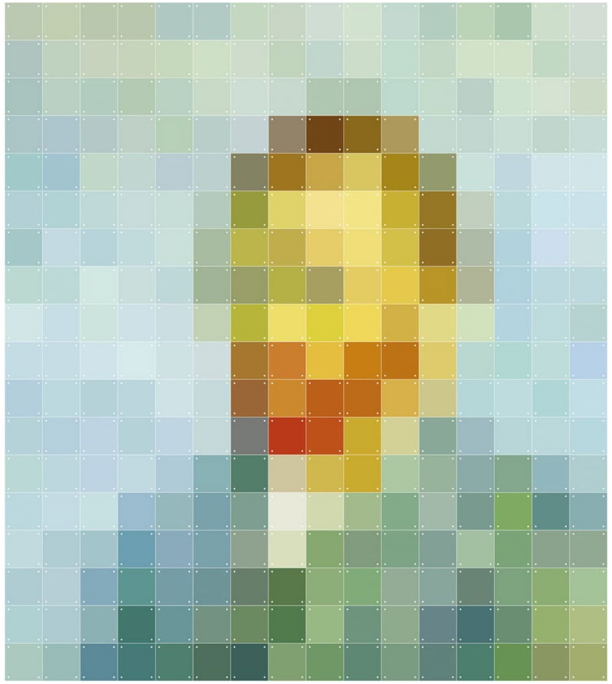
A wide range of analysis tools have also become available in recent times

Basic Image Processing Tasks

- Visualization
  - Look for visual patterns and search for trends
- Image sharpening
  - Make a better image to enhance certain features
    - Structural health monitoring
- Image retrieval
  - Seek and image of interest
    - Used in surveillence, remote sensing application
- Image measurement
  - Quantify image characteristics to manipulate images
- Image recognition
  - Identify objects in images

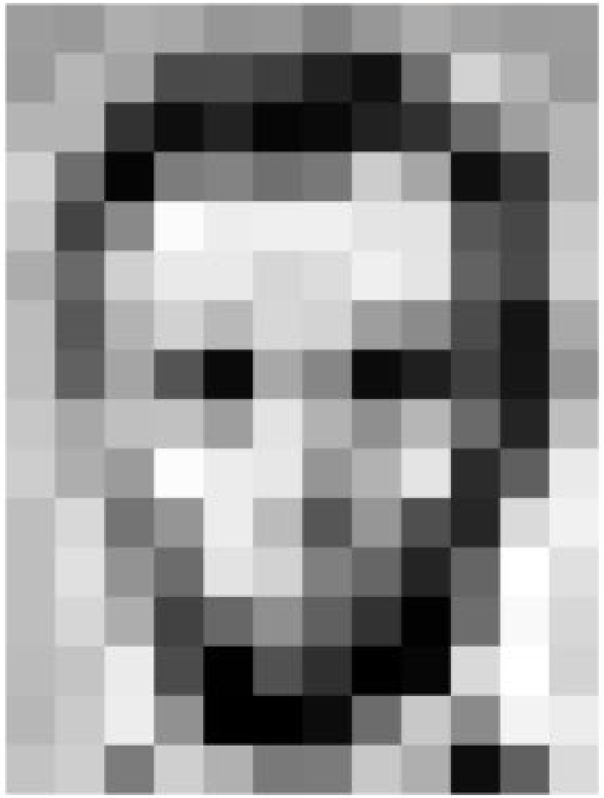
Images as Data

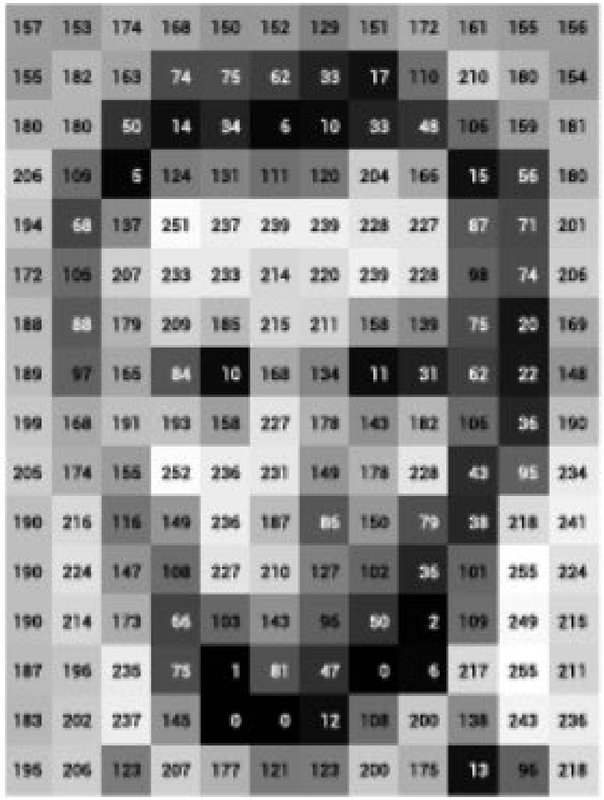
- Static images are a special form of data
  - Unlike vector outputs we have seen so far they are 2D
- An Image has a two-dimensional format
  - X and Y coordinates
- An image can be viewed to comprise many subimages
  - Each subimage is also 2D (square) called pixels



An Image comprising of pixels is called as a raster image

Images as Data





157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	106	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

You can visualize image as a graphical spreadsheet

Images as Data

- An image has a size
  - Size in units of length (in, mm)
  - Size in number of pixels (10 x 10)
- The resolution of the image connects physical and pixel sizes
  - Resolution = 5 pixels/inch
- A raster image is similar to a spreadsheet
  - Each cell (pixel) contains a value
  - It is called pixel intensity

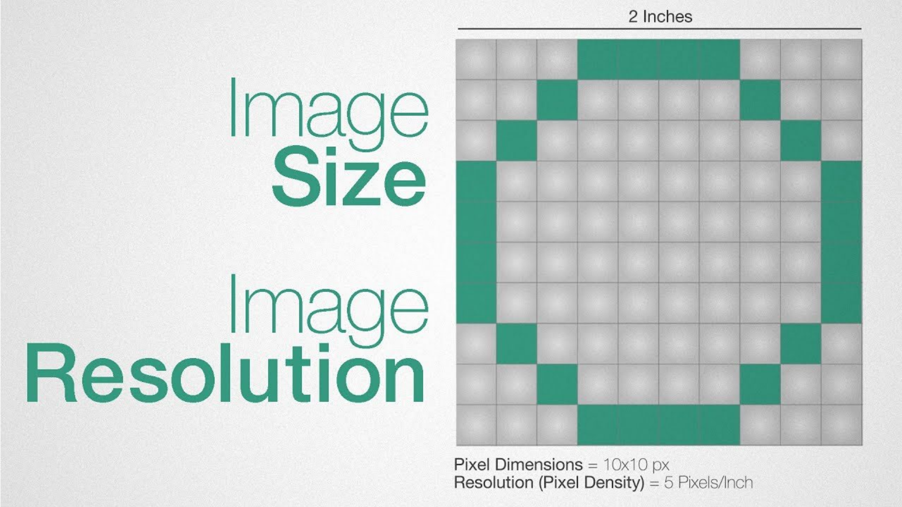
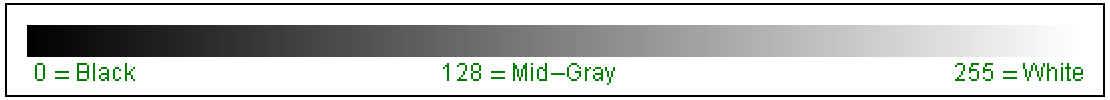


Image Type	Pixel Values
B&W	0 and 1
Greyscale 8 bit	0 - 255
Greyscale 16 bit	0 - 65535
Color 24 bit	three 8 bit R, G, B bands 0 - 255 each

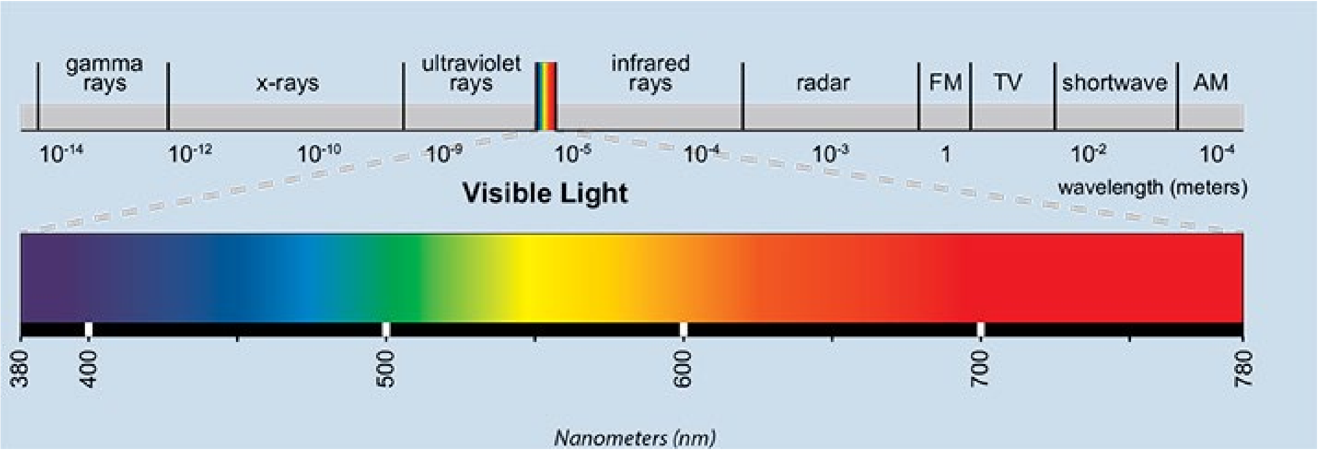
Images can have other bands especially outside the visible range

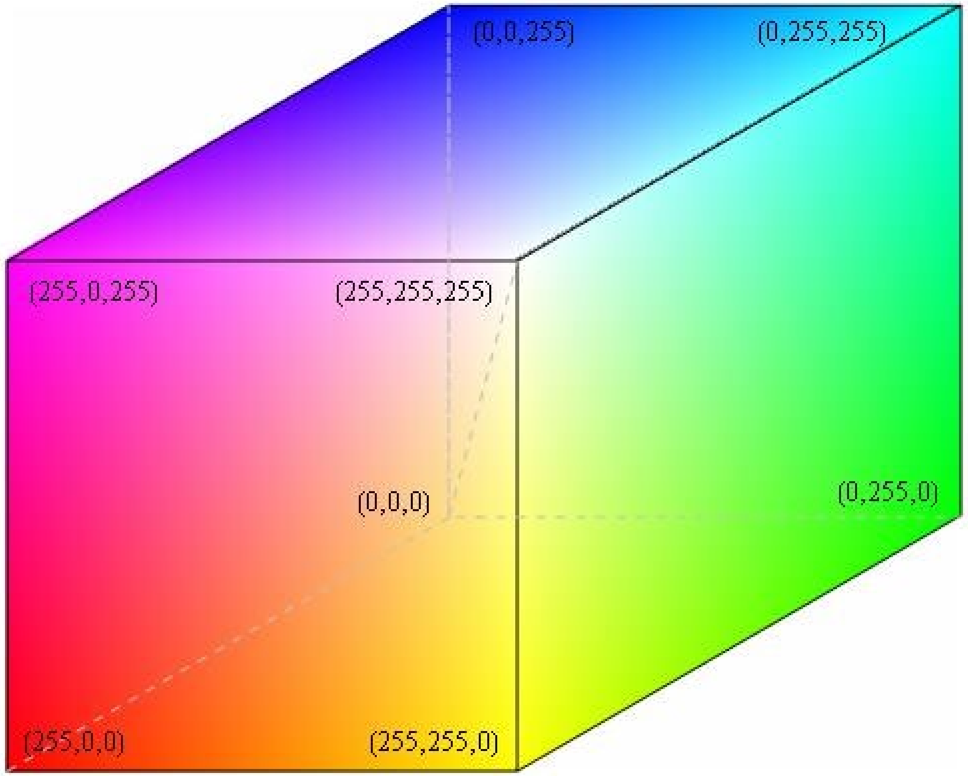
Pixel Intensities

Greyscale Intensities



Energy Spectra

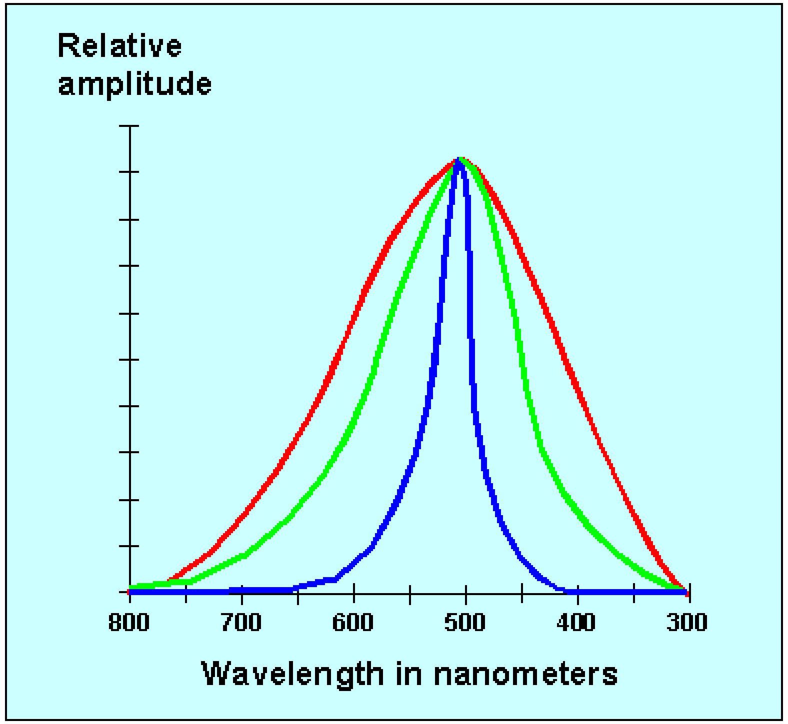




True Color Intensities

Satuation, Hue and Brightness

- Hue is the wavelength in the visible spectrum where the energy is the highest
  - Summed over RGB
- Saturation is the relative bandwidth of the visible output from a light source
  - Steepness of the curve
  - Red has low saturation; Green Medium and blue highest
  - As saturation increases color appears pure otherwise the color is washed out
- Brightness is the relative intensity
  - For RGB 0 has lowest and 255 the highest
  - Hexadecimal numbers 00 - FF



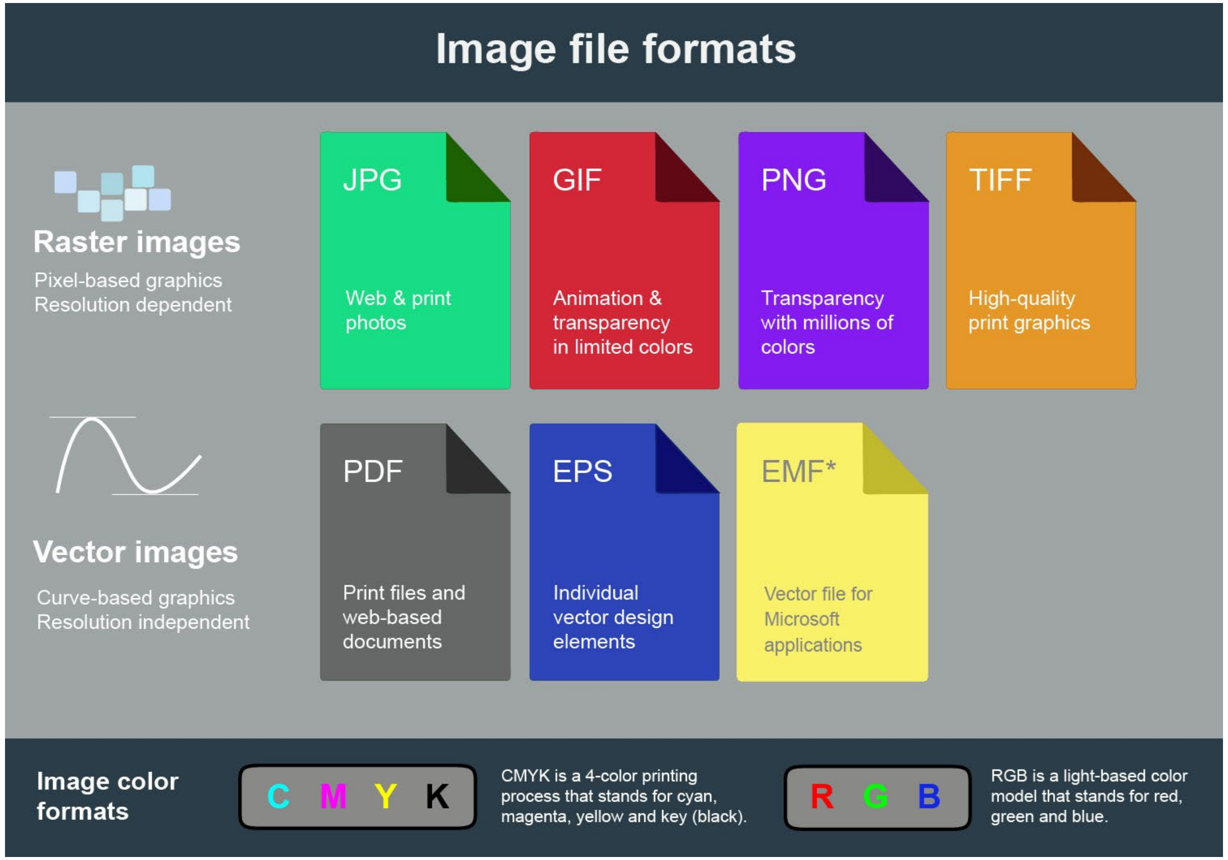
Color Space Conversion

- RGB is not very optimal
- Human brain is more attuned to changes in luminance and chrominance
  - Luminance is a measure of brightness
  - Chrominance is a measure of change from reference value at same brightness
- An RGB image can be described using 1 luminance and 2 chorminance values

Color space conversion is used in image compression (JPEG) and also to convert color to B&W pictures

Image formats

- There are a wide range of image formats
  - https://en.wikipedia.org/wiki/Image_file_formats
- Images can almost always be compressed
- There are two broad types of compression
  - Lossy Compression: trades Image quality for file size, often results in a reasonable but not perfect image reproduction
  - Lossless Compression: achieves compression while retaining a perfect copy of the image (used for image processing)
- Vector file formats use geometric description of the object instead of pixels
  - Allows more accurate reproduction at various scales



Many formats allow both lossy and lossless compression algorithms

Image Pre-Processing Tasks

- There are several image preprocessing tasks that may be necessary
  - Image reduction
  - Conversion from RGB to greyscale and viceversa
  - Image enhacement
    - Enhancement of one more more features
    - Removing unwanted effects
      - Haze, cloud cover, lighting effects
- There are several Python libraries that come in handy
  - sklearn and opencv are two commonly used packages
    - https://scikit-image.org/
  - Numpy and Scipy also have some excellent functionality

It is important that understand some basic image preprocessing if you want to work image data

Illustrative Example

Perform some basic image processing tasks on a famous civil engineering image

Basic tasks

- Load image into Python
- Understand the dimensions of the image
- Perform some basic manipulations

```
"""Created on Mon Nov 16 07:01:06 2020Basic
Image Processing tasks@author: vuddameri"""#

# Load Libraries
import osimport numpy as np
import skimage as skifrom skimage
import io
from matplotlib import pyplot as plt
```

```
# Function to plot image and add title
def showimg(image,title='Image',cmap_type='gray'):
    plt.imshow(image,cmap=cmap_type)
    plt.title(title)
    plt.axis('off')
    plt.show()
```

```
# setworking directory
dir = '/home/vuddameri/CE5319/CNN/Code'
os.chdir(dir)
```

```
# load RGB image
file = 'bridge.jpeg'
bridge = io.imread(file)
showimg(bridge,'Golden Gate Bridge')
bridge
```

```
# Convert to Greyscale
bridge_gs = ski.color.rgb2gray(bridge)
showimg(bridge_gs,'Golden Gate Bridge')
bridge_gs
```

```
# Use type to look at image types
# You see they are numpy arrays
type(bridge)
type(bridge_gs)
```

```
# Obtain Red, Green and Blue Values from NP array
red = bridge[:,:,0]
green = bridge[:,:,1]
blue = bridge[:,:,2]
type(red)
# Plot using plot function
fig,axs = plt.subplots(1,3)
axs[0].imshow(red)
axs[1].imshow(green)
axs[2].imshow(blue)
axs[0].title.set_text('Red')
axs[1].title.set_text('Green')
axs[2].title.set_text('Blue')
plt.show()
```

Basic Tasks

```
# Plot RGB colors using showimg
showimg(red,title='Red Component')
showimg(green,'Green Component')
showimg(blue, 'Blue Component')
```

```
# Get shape & size of the image
bridge.shape
bridge_gs.shape
bridge.size
bridge_gs.size
```

```
# Get the historgam of greyscale image
histval = np.round(bridge_gs.ravel() * 255,0)
plt.hist(histval,bins=256)
plt.ylabel('Counts')
plt.xlabel('Values')
plt.title('Histogram of Intensities')
plt.grid()
plt.show
```

```
# Thresholding Tasks
# Import the otsu threshold function
from skimage.filters import threshold_otsu
# Obtain the optimal threshold value
thresh = threshold_otsu(bridge_gs)
# Apply thresholding to the image
binary_global = bridge_gs > thresh
showimg(binary_global,"global Thresholding")
histval1 = np.round(binary_global.ravel() * 255,0)
plt.hist(histval1,bins=256)
```

```
# Flatten an image
bridge1d = bridge.flatten()
type(bridge1d)
```

Results

Golden Gate Bridge

R G B

Pixel

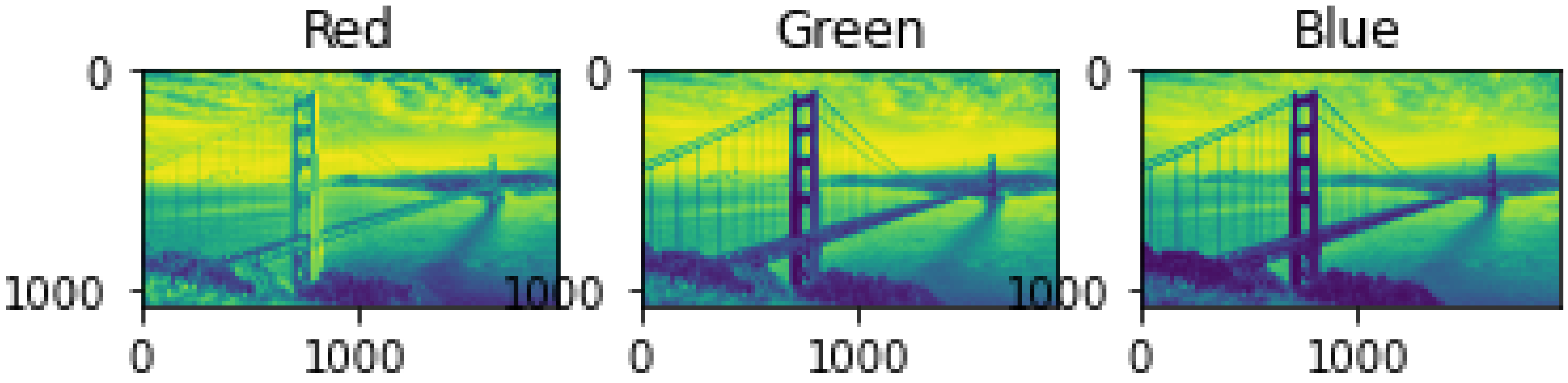
```
array([[[ 66, 112, 125],
        [ 66, 112, 125],
        [ 66, 112, 125],
        ...,
        [ 86, 146, 156],
        [101, 154, 162],
        [111, 160, 167]],

       [[ 67, 113, 126],
        [ 67, 113, 126],
        [ 67, 113, 126],
        ...,
        [ 96, 148, 159],
        [109, 151, 163],
        [116, 154, 165]],

       [[ 68, 114, 127],
        [ 68, 114, 127],
        [ 68, 114, 127],
        ...,
        [111, 153, 167],
        [120, 154, 166],
        [126, 154, 166]],

       ...,

        [ 10,  42,  39],
        [ 31,  63,  58],
        [ 45,  77,  72]]], dtype=uint8)
```

Grayscale Image

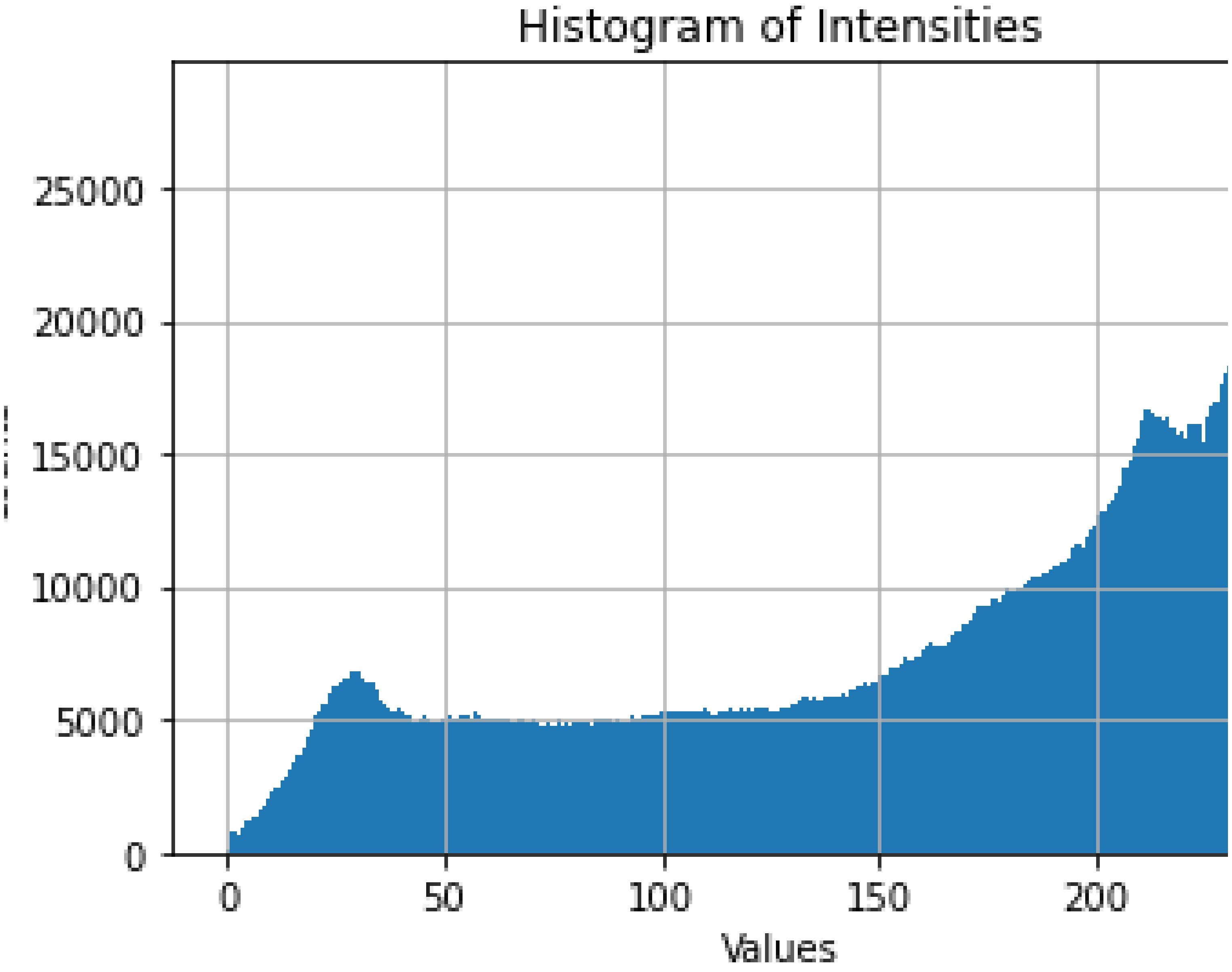
Golden Gate Bridge

Golden Gate Bridge

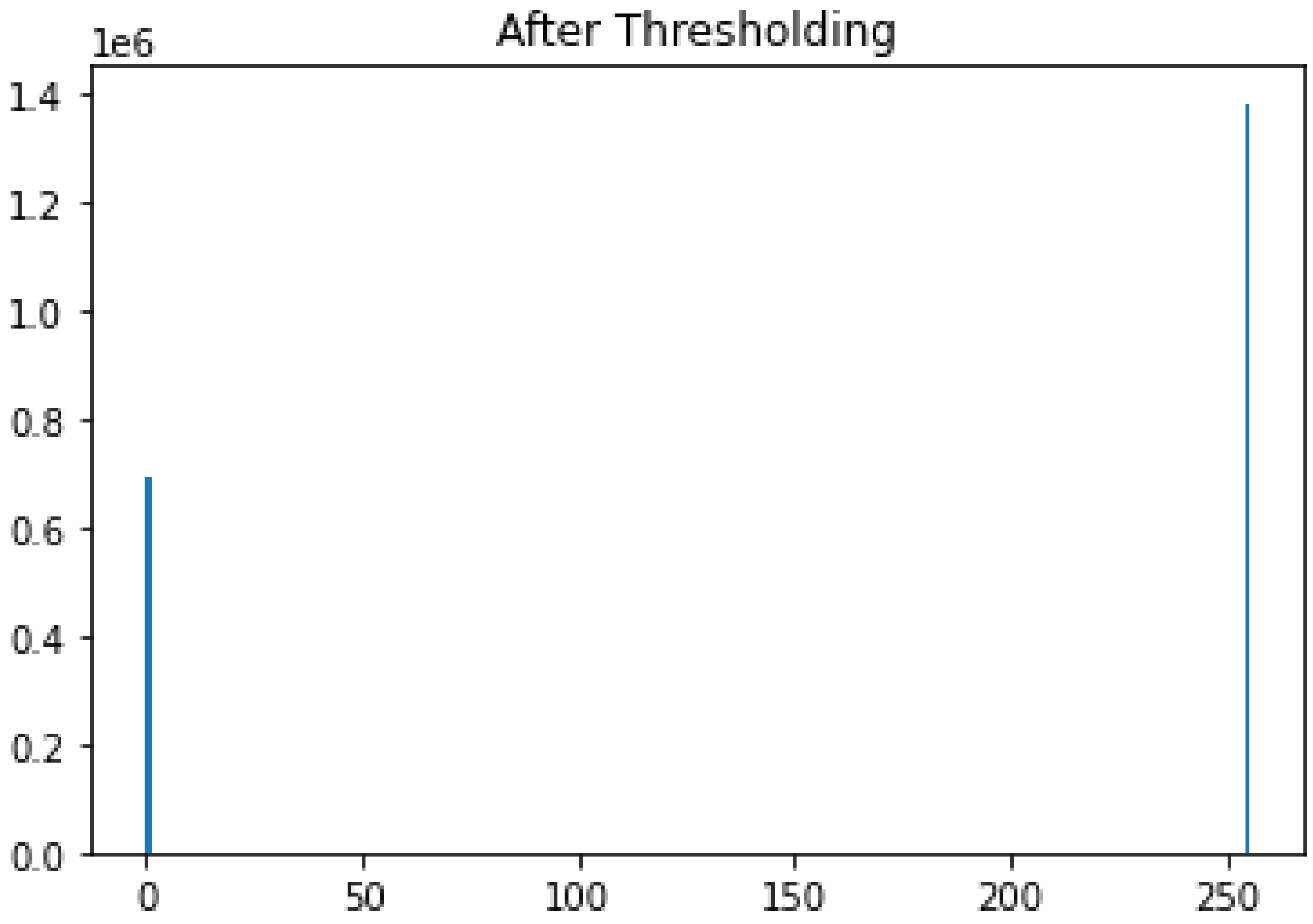
RGB Components



Histogram Components



Thresholding Effects



Data flattening

R G B

```
array([[[ 66, 112, 125],
        [ 66, 112, 125],
        [ 66, 112, 125],
        ...,
        [ 86, 146, 156],
        [101, 154, 162],
        [111, 160, 167]],

       [[ 67, 113, 126],
        [ 67, 113, 126],
        [ 67, 113, 126],
        ...,
        [ 96, 148, 159],
        [109, 151, 163],
        [116, 154, 165]],

       [[ 68, 114, 127],
        [ 68, 114, 127],
        [ 68, 114, 127],
        ...,
        [111, 153, 167],
        [120, 154, 166],
        [126, 154, 166]],

       ...,

        [ 10,  42,  39],
        [ 31,  63,  58],
        [ 45,  77,  72]]], dtype=uint8)
```

Pixel

```
bridge1d = bridge.flatten()
type(bridge1d)
Out[11]: numpy.ndarray

bridge1d
Out[12]: array([ 66, 112, 125, ...,  45,  77,  72], dtype=uint8)
```

You should know

- What is a raster image
- The role of pixels
- Difference between color (RGB) and greyscale images
- Saturation, hue and brightness
- Color spaces
- Common image formats
- Basic image operations
    - Grayscale conversion, histograms, thresholding, flattening