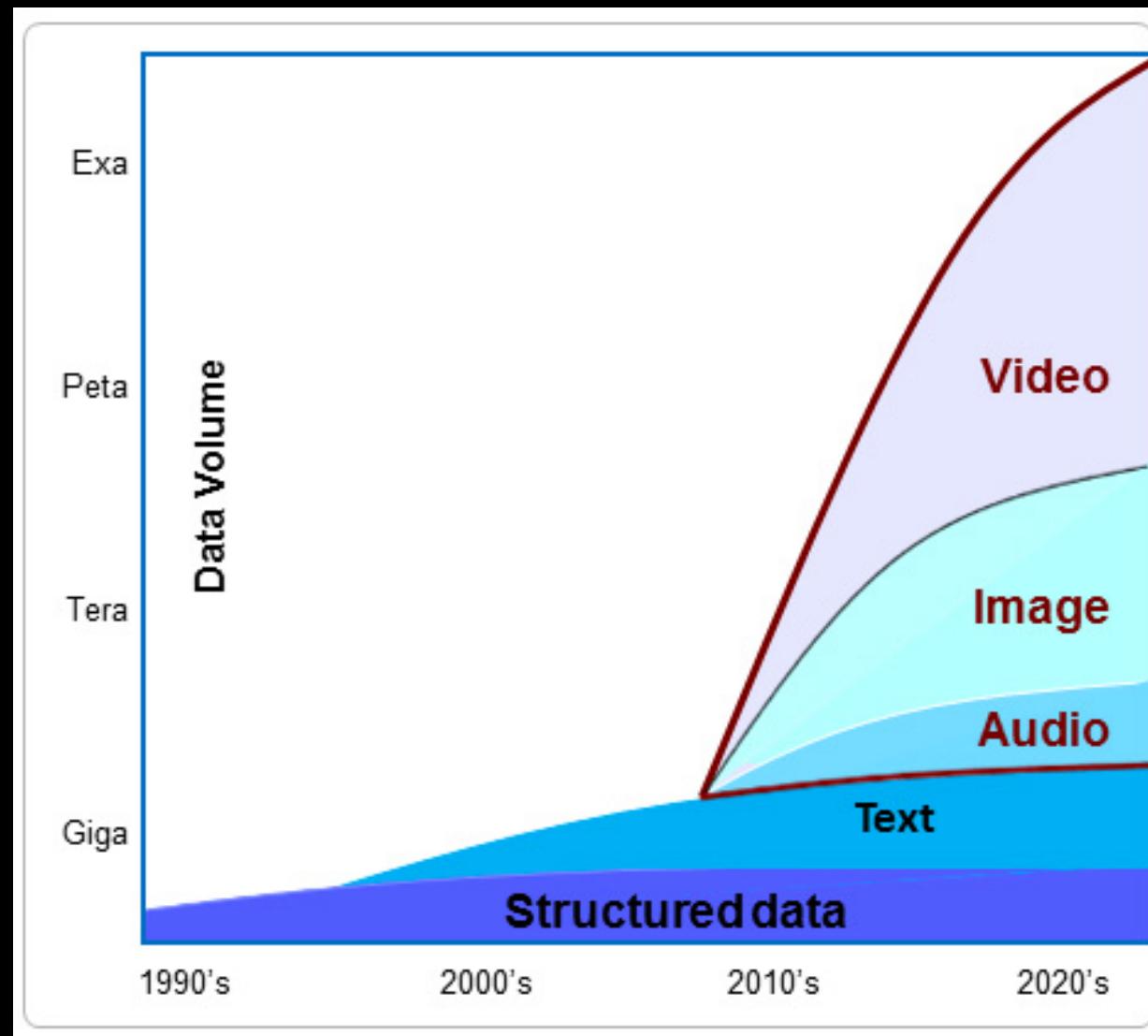


Image Processing

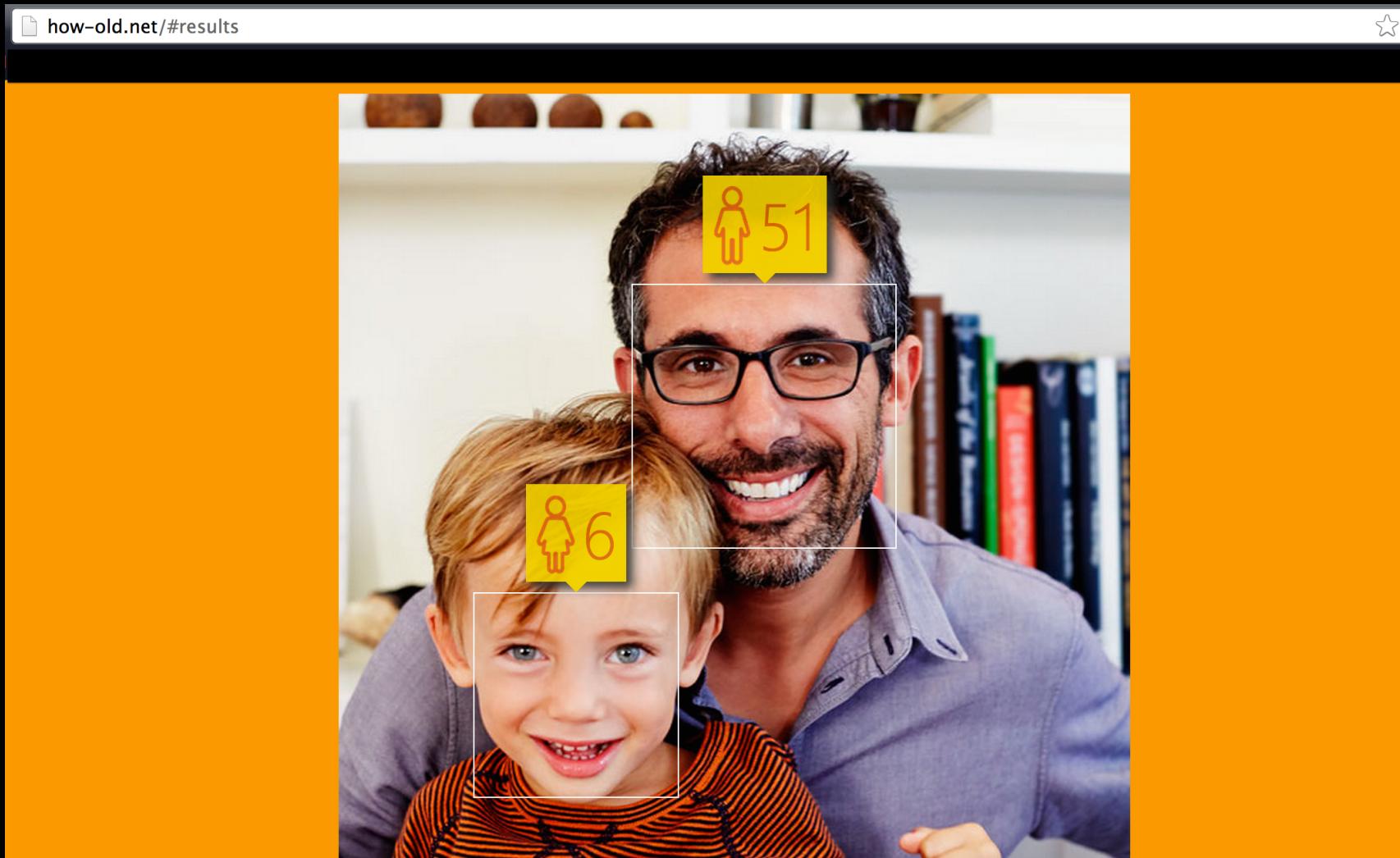
Goals

- Motivation and Applications
- Tools and Data pipeline
- Basic computer vision techniques

Motivation



Application



Framework

- Read
- Resize
- Transformations
- Featurize

Python Libraries

- Scikit-image (Skimage)
- OpenCV (Based on C++)
 - ★ Face detection

Framework

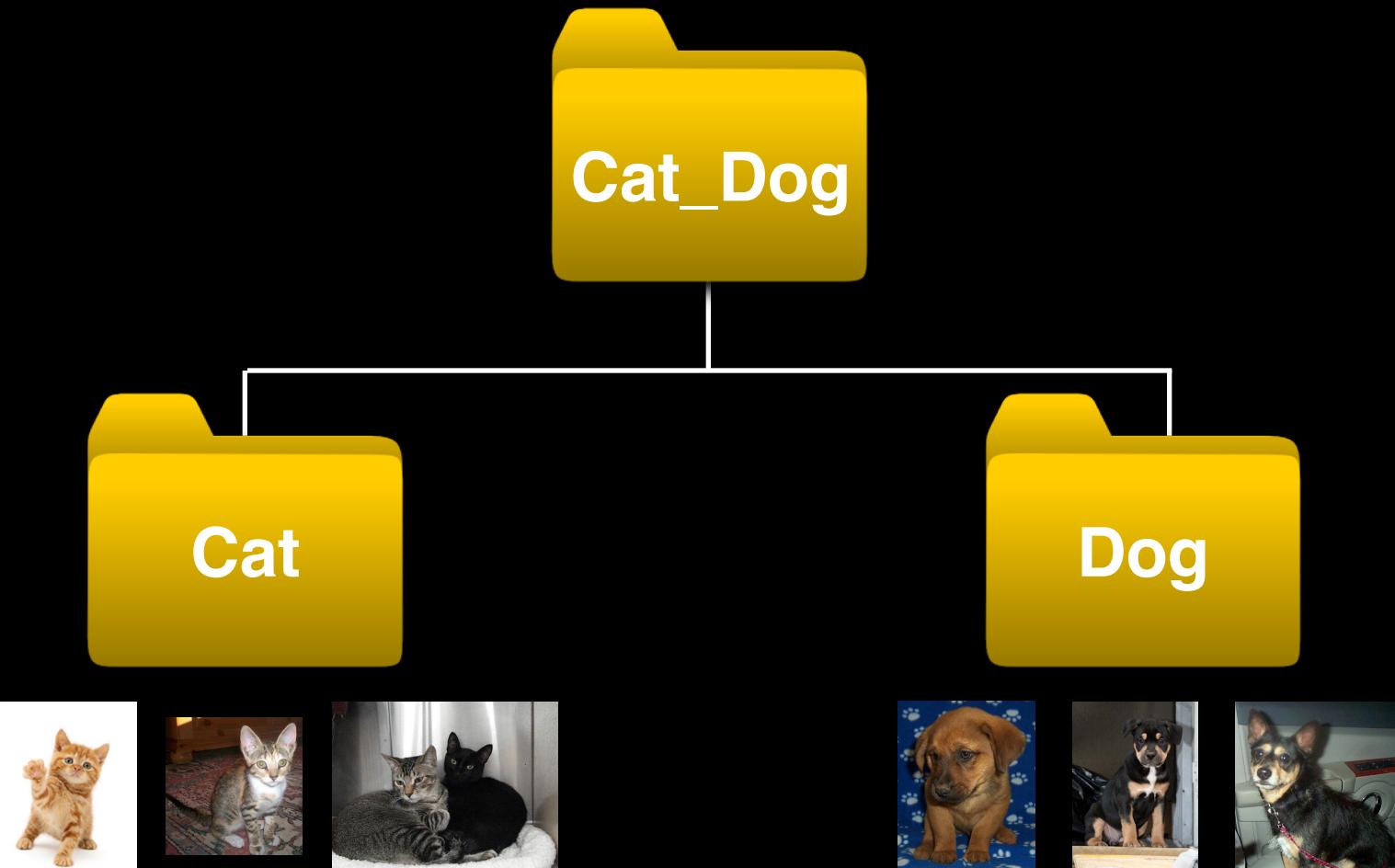
- Read
- Resize
- Transformations
- Featurize

Read

- Read an image into a matrix of numbers
- For image classification, read in sub-directories of images
- Each sub-directory is named after the label

Directory Structure

Parent Directory



$\left[\left[\begin{array}{c} \text{Image of an orange tabby kitten} \\ \text{Image of two kittens sitting together} \\ \text{Image of two cats sitting together} \end{array} \right], \left[\begin{array}{c} \text{Image of a brown puppy} \\ \text{Image of a black and tan puppy} \\ \text{Image of a small black and tan dog} \end{array} \right] \right]$

Image Tensor



```
array([
    [R G B, R G B],
    [[108, 50, 13], [111, 55, 18]],
    [R G B, R G B],
    [[115, 61, 23], [130, 129, 127]]])
```

Framework

- Read
- Resize
- Transformations
- Featurize

Resize

- Make image a specified shape
- Not cropping
- Uniform feature matrix downstream

(See Featurization later)

Resize Cont.

- For colored images:

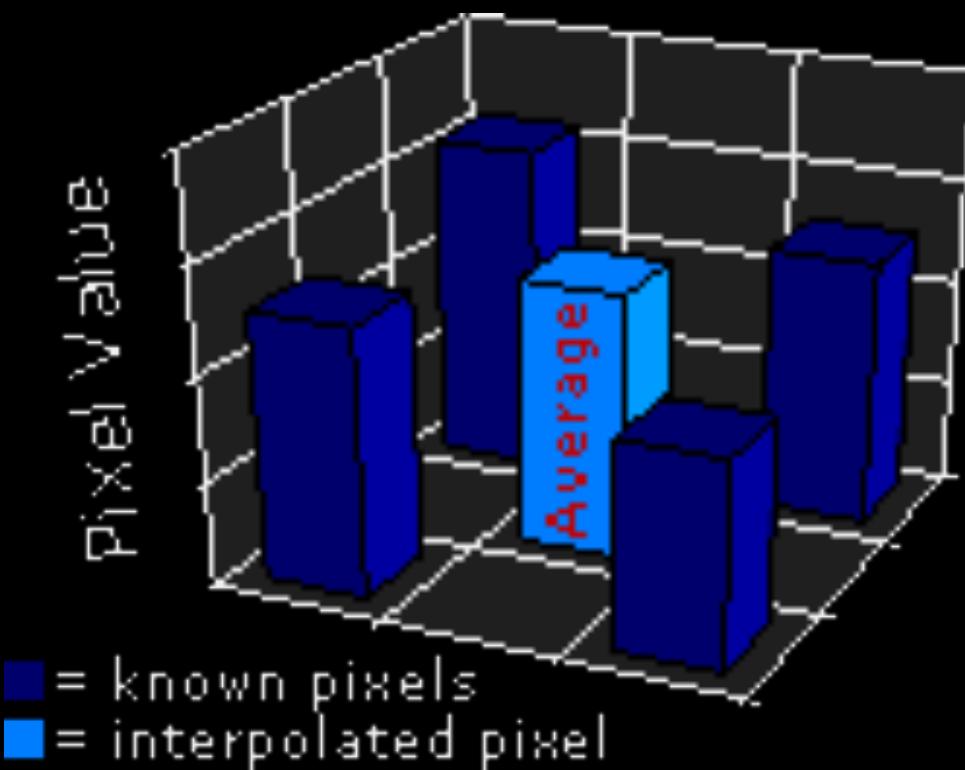
```
shape = (width, height, 3)
```

- For grayscale images:

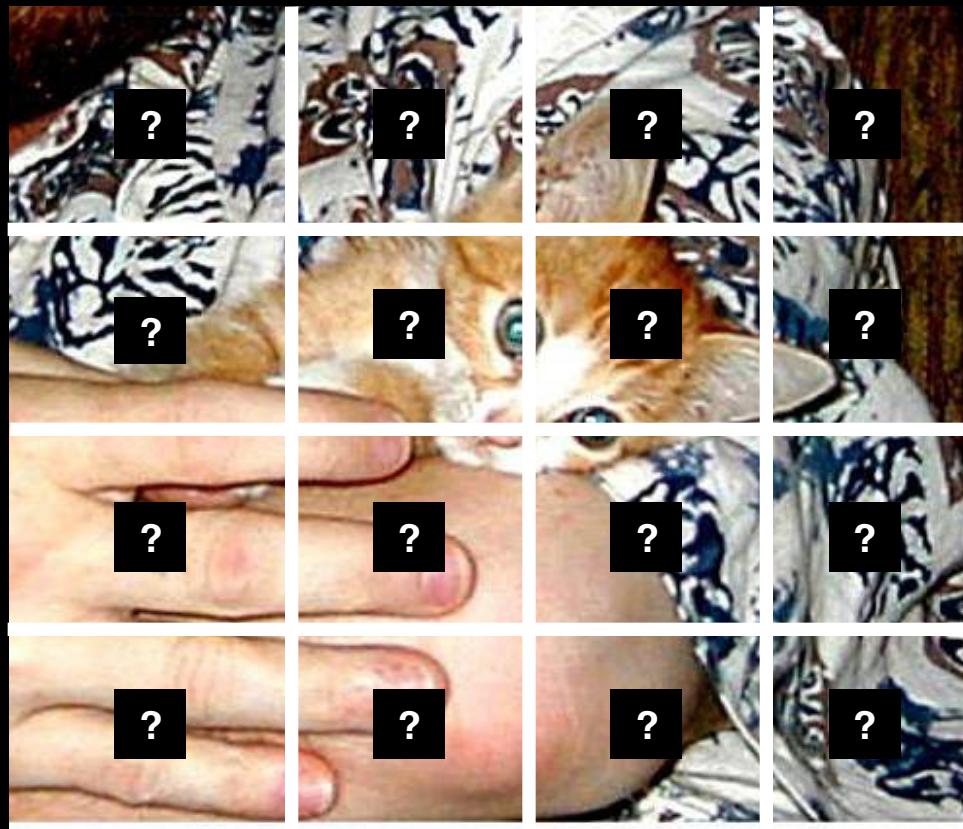
```
shape = (width, height)
```

Interpolation

Bi-linear Interpolation



Down-Sample Interpolation



Up-Sample Interpolation



Down-sampling

- Reduce size / dimensions
- Less processing time downstream
- Lose resolution
- Visually verify

Framework

- Read
- Resize
- Transformations
- Featurize

Image Transformations

- Convert an image from one domain to another

Grayscale

- Compute luminance of the **RGB** image
- $Y = 0.2125 \text{ } \mathbf{R} + 0.7154 \text{ } \mathbf{G} + 0.0721 \text{ } \mathbf{B}$
- Normalize to range from 0 to 1
- 3D tensor matrix \rightarrow 2D matrix



Denoise

- Removing unnecessary details of an image
- Better generalization of the class of image



Method 1: Convolution

35	40	41	45	50
40	40	42	46	52
42	46	50	55	55
48	52	56	58	60
56	60	65	70	75



0	1	0		
0	0	0		
0	0	0		



Input

Kernel

Output

Gaussian Kernel

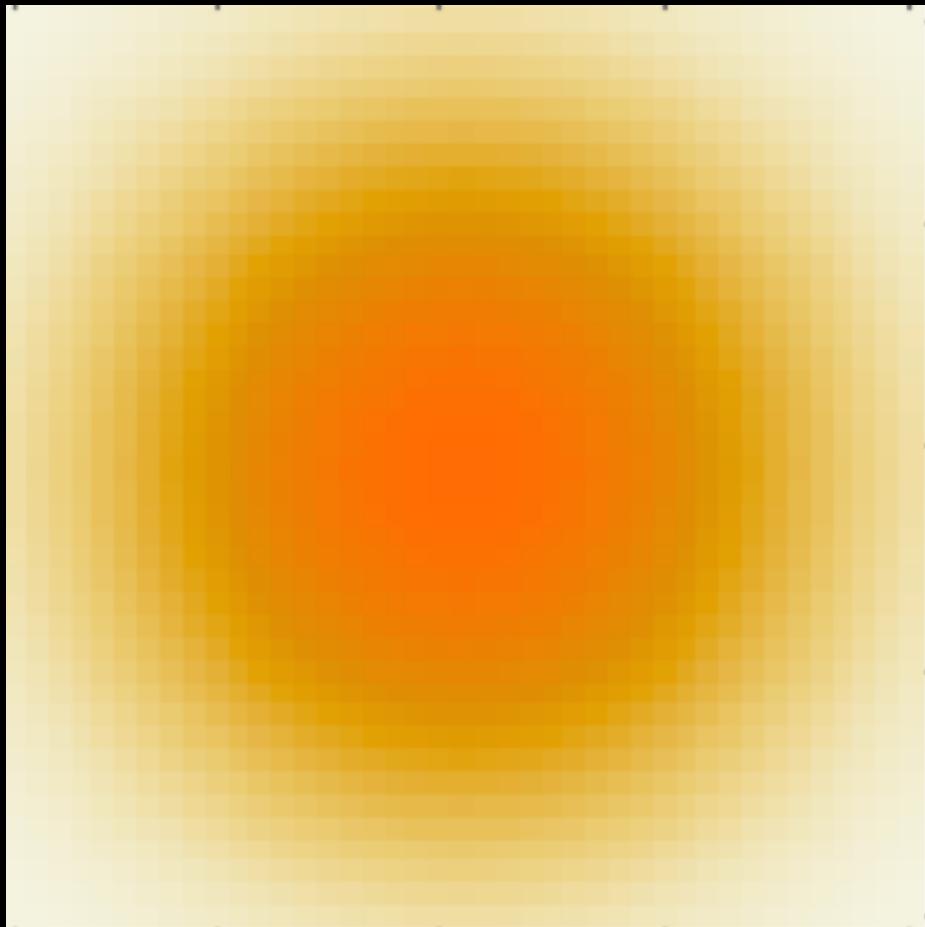
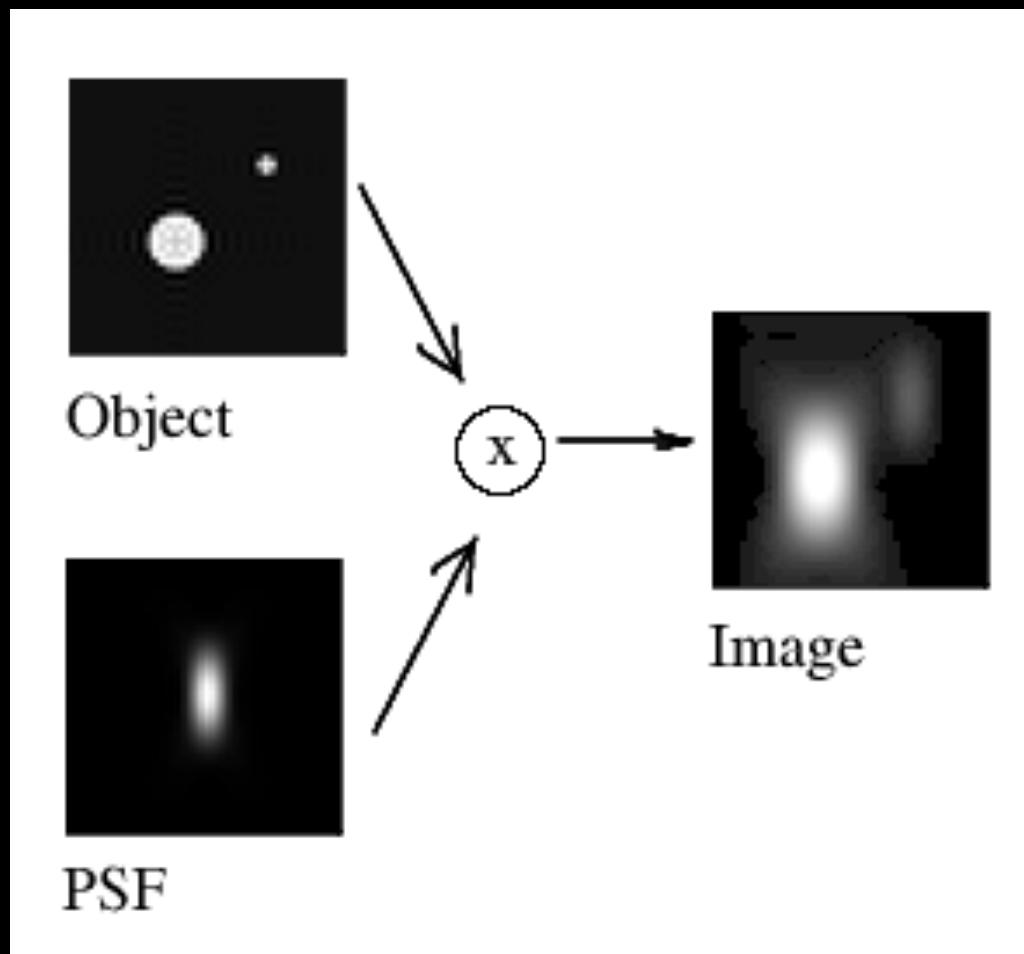


Image Convolution



Bi-lateral Denoise

- **Bilateral:** *Spatial and Pixel value*
- Convolution with a kernel
- Usually Gaussian Kernel

Bi-lateral Denoise Cont.

- Higher weights for spatially close pixel
- Higher weights for pixel value similarity

Method 2: Variation Minimization

Fidelity: $E(x, y) = \frac{1}{2} \sum_n (x_n - y_n)^2$

Variation: $V(y) = \sum_n |y_{n+1} - y_n|$

Minimize: $E(x, y) + \lambda V(y)$

How much to denoise

x_n is the original data

y_n is denoised data

Total Variation Denoise

- Takes much longer if the image is noisy
- Can adjust λ

Edge Detection

- Only works with grayscale images
- **Edge:**
 - ★ Large change of color between neighboring pixels



Canny



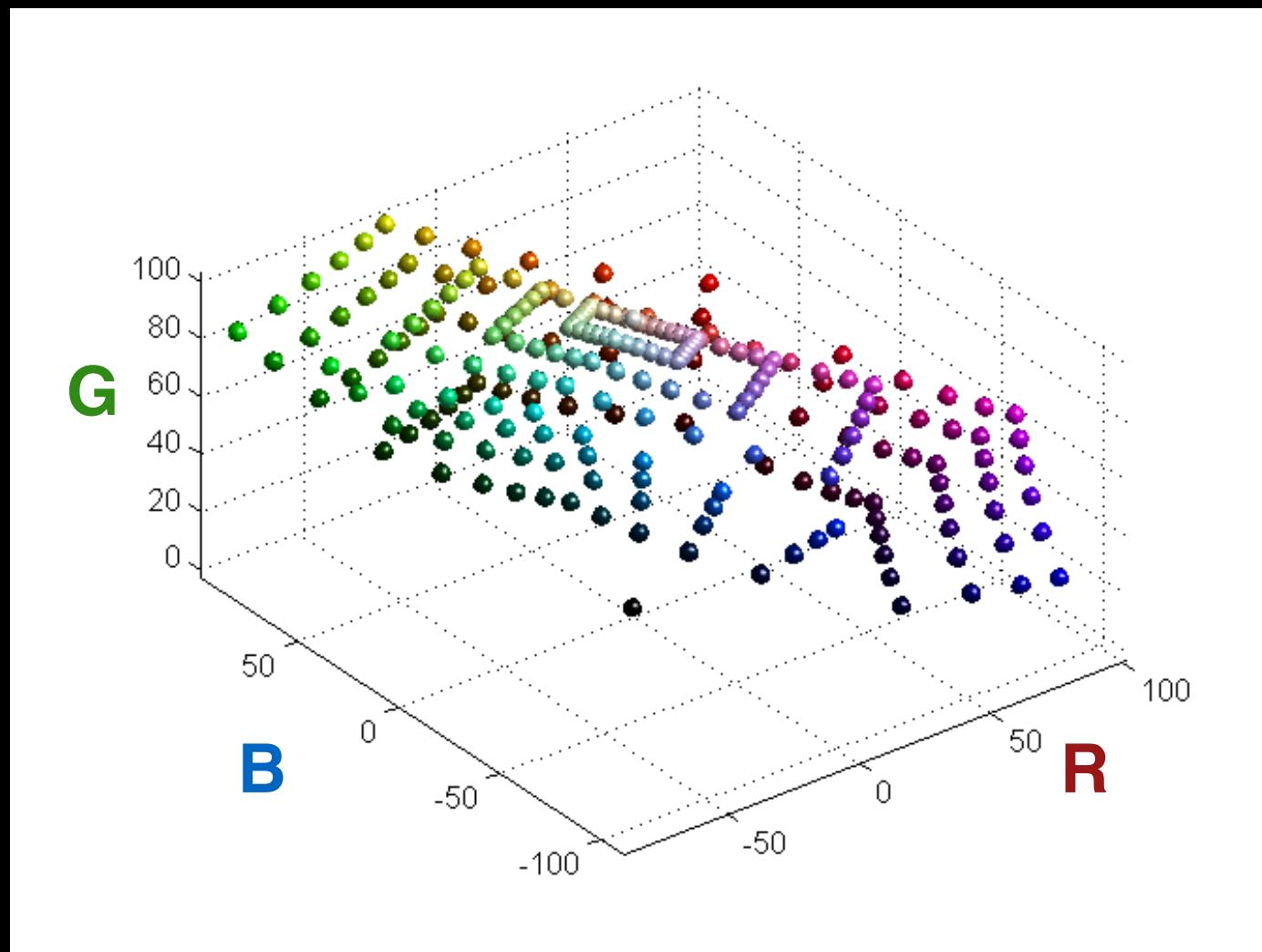
Sobel



Dominant Colors

- Most common colors in the image
- KMeans to cluster the RGB pixels
- Centriods represent the dominant colors

K-Means of RGB pixels





Framework

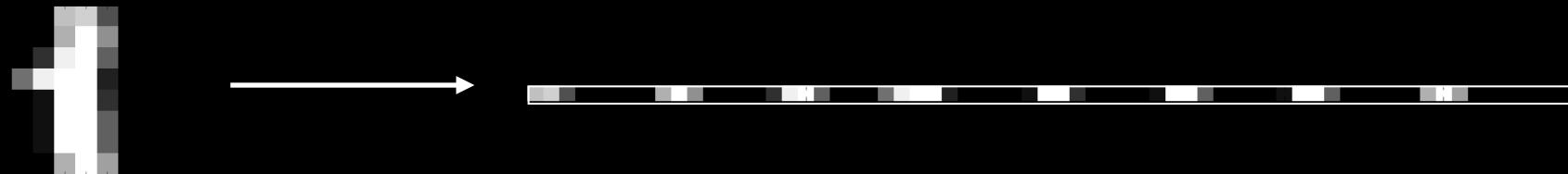
- Read
- Resize
- Transformations
- Featurize

Vectorize Image

`numpy.ravel()`



`numpy.ravel()`



Framework

- Read
- Resize
- Transformations
- Featurize

End of Morning
Afternoon lec @ 2pm