Lecture Notes for
Machine Learning in Python

⟦👨‍🏫👨‍💻🐍👨‍🔬⟧

Professor Eric Larson

Dimensionality and Images

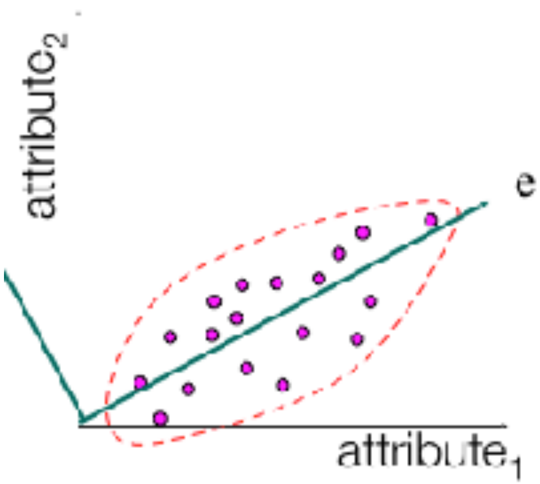
Class Logistics and Agenda

- **Logistics**:
    - Next lab due at the end of the week
    - Text with visuals!
    - Seminar Wednesday on Maneframe
    - Next time: no class if we finish everything today
        - Office Hours Location?
- **Agenda**
    - Kernel Methods
    - Common Feature Extraction Methods for Images

Last time it was so linear…

E1	E2
0.85	0.85
0.52	-0.52

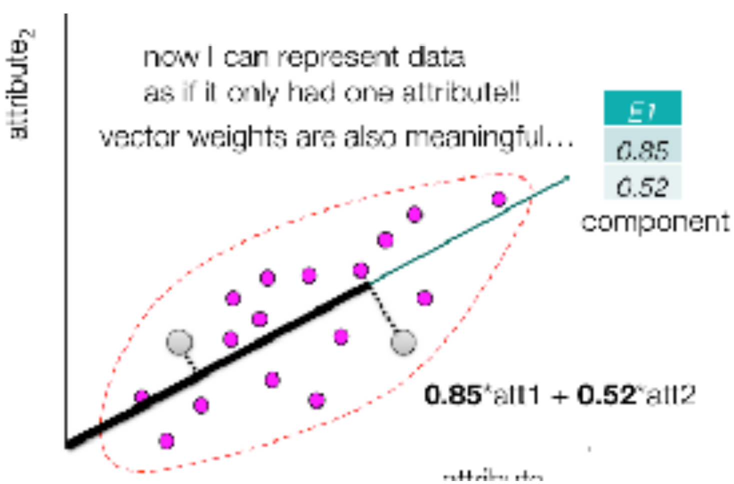
37.1	-6.7
-6.7	43.9



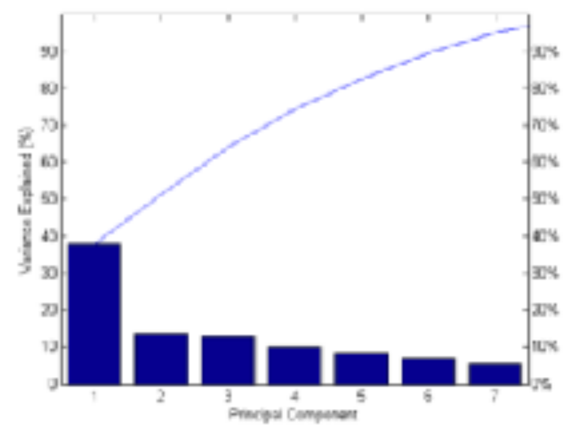
	A1	A2
1	66	33.6
2	54	26.6
3	69	23.3
4	73	28.1
5	61	43.1
6	62	25.6

	A1	A2
1	1.83	3.55
2	-10.1	-3.45
3	4.83	-6.75
4	8.83	-1.95
5	-3.17	13.05
6	-2.17	-4.45

zero mean



$$r_q = \frac{\sum_{j=1}^{q} \lambda_j}{\sum_{j=1}^{p} \lambda_j}$$



Dimensionality Reduction: Randomized PCA

- **Problem**: PCA on all that data can take a while to compute
  - What if the number of dimensions is gigantic?
    - Actually, **that's okay**: there are **iterative** algorithms for finding the **largest eigenvalues** that scales well with the number of data dimensions, but **not** the **number of instances**…
  - What if the number of instances is gigantic?
- What if we construct the covariance matrix with a subsample of the data?
  - By **randomly sampling from the dataset**, we can get something representative of the covariance for the entire dataset (if we sample correctly)

source: Halko, et al., (2009) Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions.

04.Dimension Reduction and Images.ipynb

Dimension Reduction

PCA

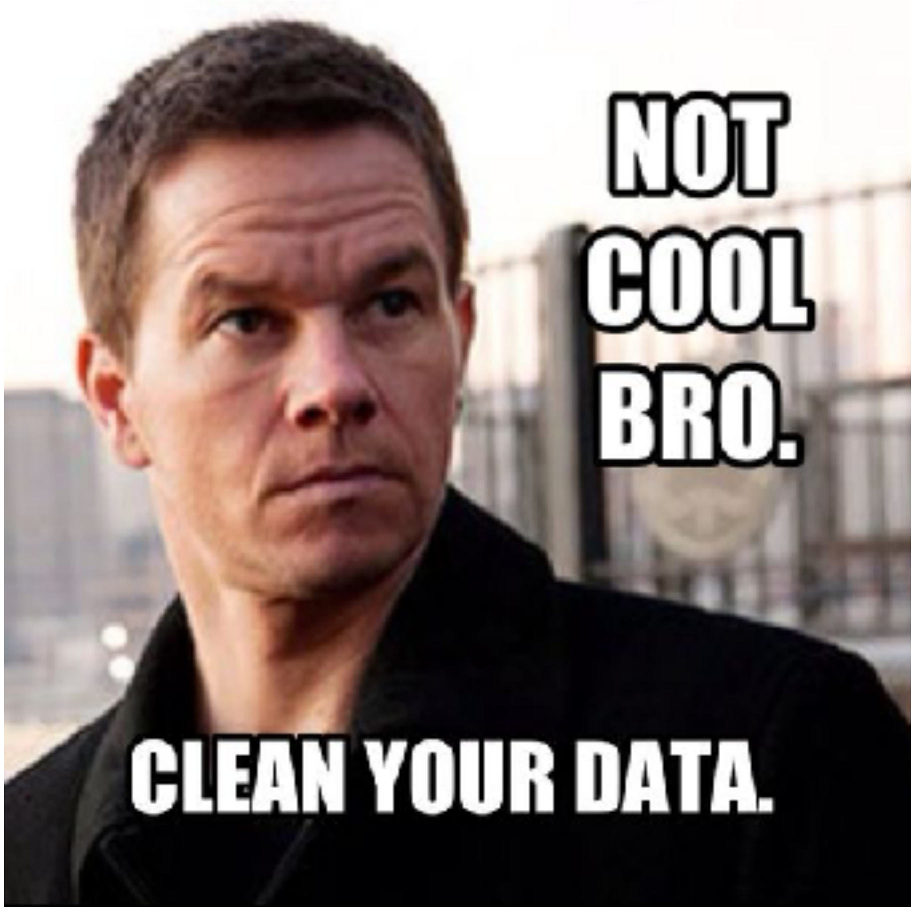
biplots

Other Tutorials:

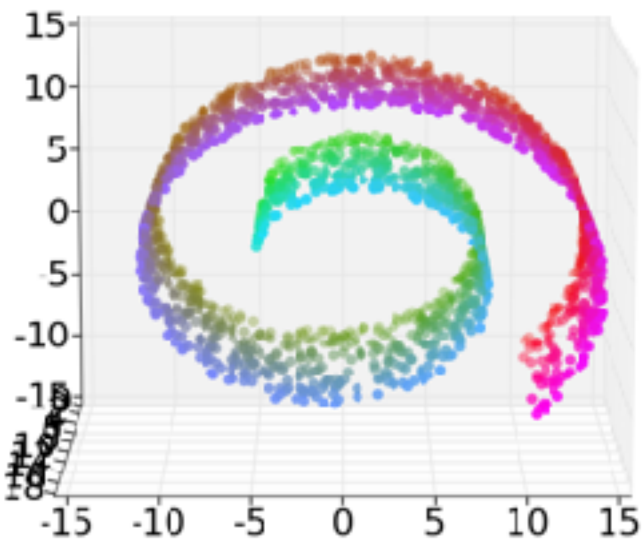
http://scikit-learn.org/stable/auto_examples/decomposition/plot_pca_vs_lda.html#example-decomposition-plot-pca-vs-lda-py

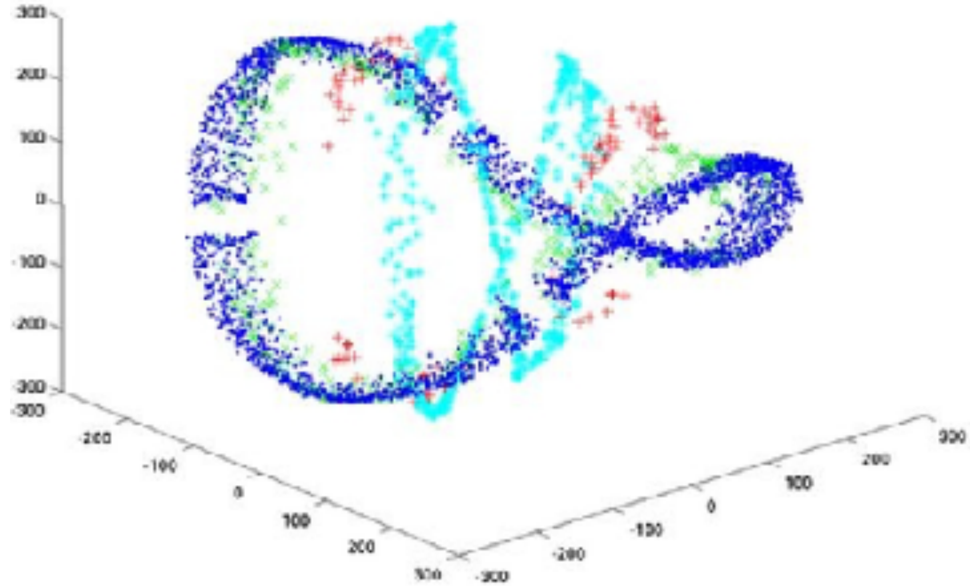
http://nbviewer.ipython.org/github/ogrisel/notebooks/blob/master/Labeled%20Faces%20in%20the%20Wild%20recognition.ipynb

Non-linear Dimensionality Reduction



Dimensionality Reduction: non-linear

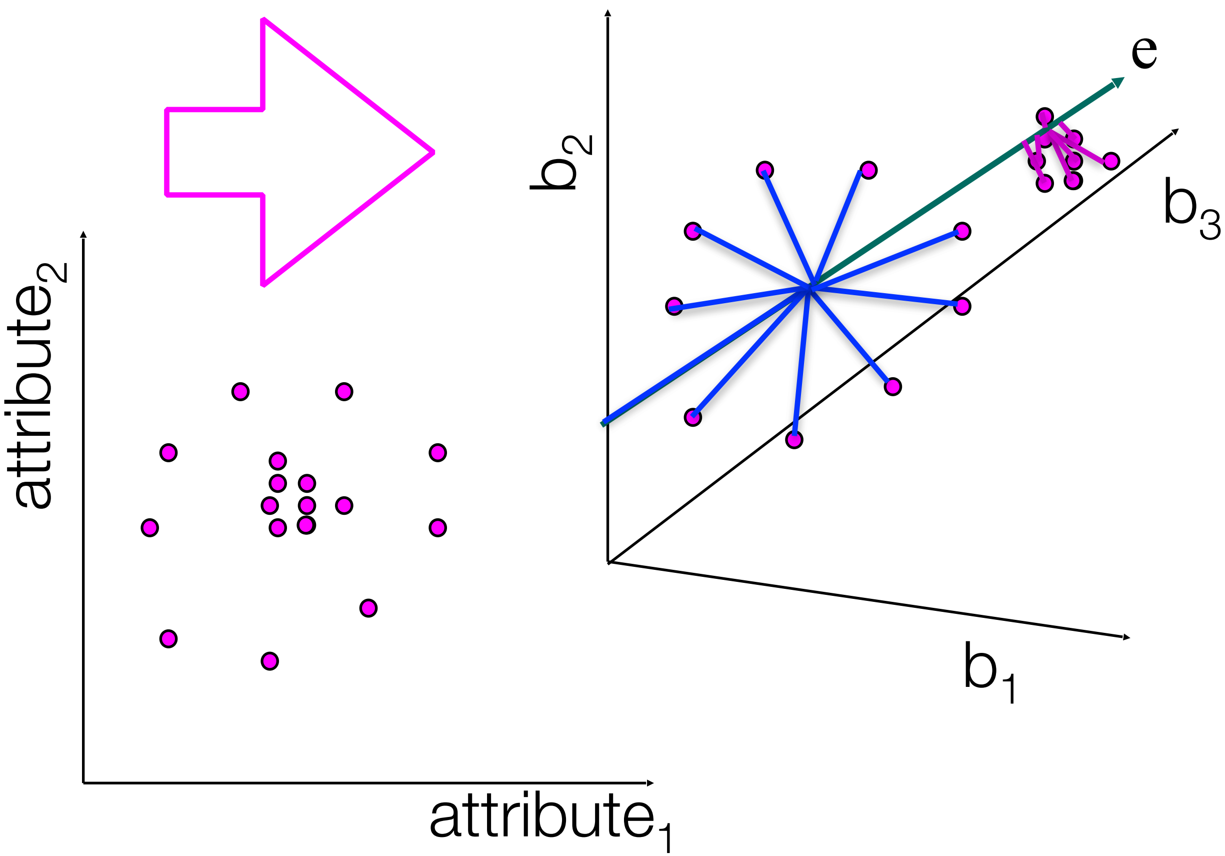




- Sometimes a **linear transform** is not enough
- A powerful non-linear transform has seen a resurgence in past decade: **kernel PCA**

Kernel PCA

- Estimate Covariance in higher dimensional space
- Get eigen vectors from nonlinear dot product
- Projecting onto these can be understood as principle components from a higher dimensional space



$\phi(A1)\cdot\phi(A1)$	$\phi(A2)\cdot\phi(A1)$
$\phi(A2)\cdot\phi(A1)$	$\phi(A2)\cdot\phi(A2)$

	A1	*A2*
1	*66*	*33.6*
2	*54*	*26.6*
3	*69*	*23.3*
4	*73*	*28.1*
5	*61*	*43.1*
6	*62*	*25.6*

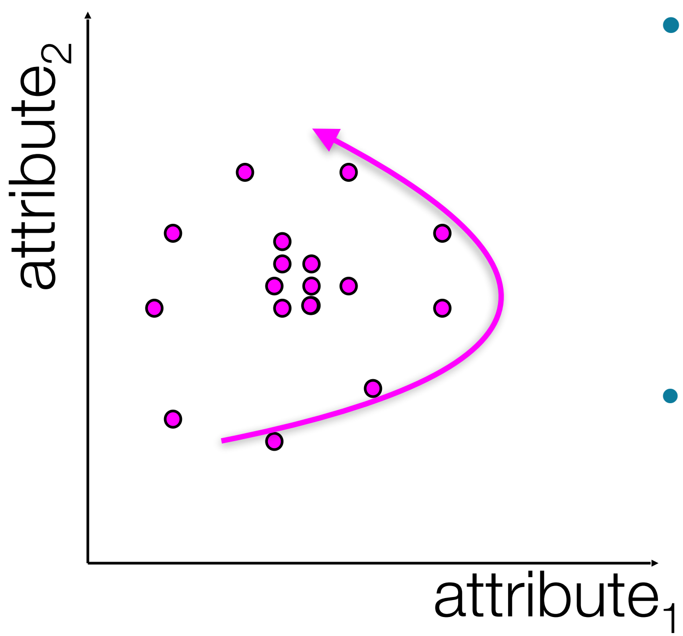
Kernel PCA

kernel: defines what the dot product is in higher dimensional space

$\phi(A1)\cdot\phi(A1)$	$\phi(A2)\cdot\phi(A1)$
$\phi(A2)\cdot\phi(A1)$	$\phi(A2)\cdot\phi(A2)$

some kernels have corresponding transformations with **infinite dimensions**!!

	A1	*A2*
1	*66*	*33.6*
2	*54*	*26.6*
3	*69*	*23.3*
4	*73*	*28.1*
5	*61*	*43.1*
6	*62*	*25.6*



- **Key insight**: don't need to know the actual principle components, just the projections
- **Never need** eigen vectors of full covariance matrix

Kernel PCA

kernel: defines what the dot product is in higher dimensional space

$\phi(A1)\cdot\phi(A1)$	$\phi(A2)\cdot\phi(A1)$
$\phi(A2)\cdot\phi(A1)$	$\phi(A2)\cdot\phi(A2)$

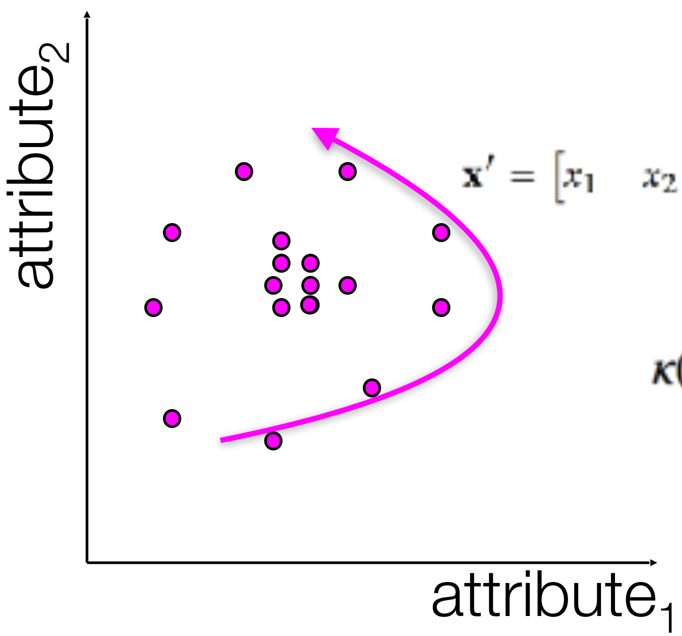
some kernels have corresponding transformations with **infinite dimensions**!!

	A1	A2
1	66	33.6
2	54	26.6
3	69	23.3
4	73	28.1
5	61	43.1
6	62	25.6

$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T \qquad \mathbf{x} \in \mathbb{R}^d$$

$$\Downarrow \phi$$

$$\mathbf{x}' = \begin{bmatrix} x_1 & x_2 & x_1x_2 & x_1^2 & x_1x_2^3 & \dots \end{bmatrix}^T \qquad \mathbf{x} \in \mathbb{R}^k (k >> d)$$

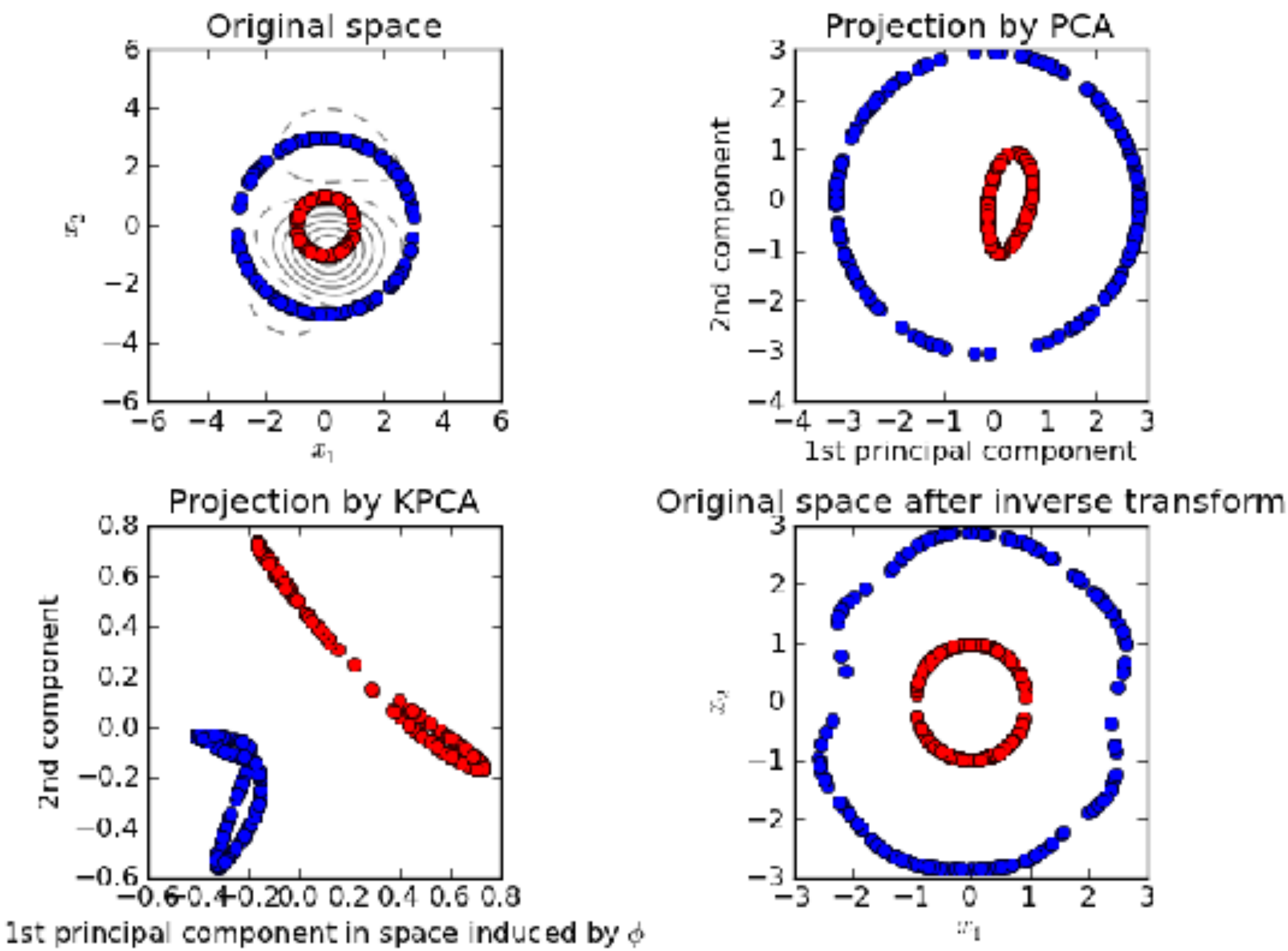


$$\kappa(\mathbf{x_i}, \mathbf{x_j}) = exp\left(-\gamma \; \|\mathbf{x_i} - \mathbf{x_j}\|_2^2 \right)$$

kernel: radial basis function (rbf)

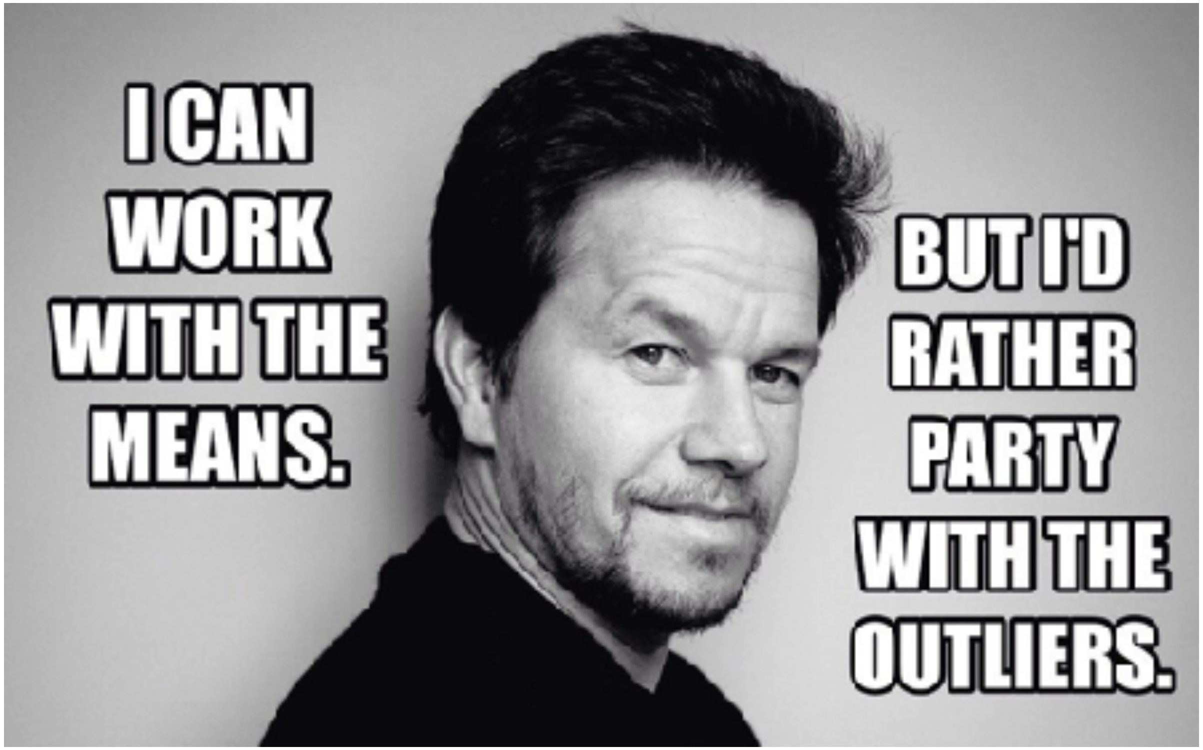
http://sebastianraschka.com/Articles/2014_kernel_pca.html#kernel-functions-and-the-kernel-trick

Kernel PCA



Original space
Projection by PCA
1st principal component
2nd component
Projection by KPCA
1st principal component in space induced by ϕ
2nd component
Original space after inverse transform

Image Processing and Representation

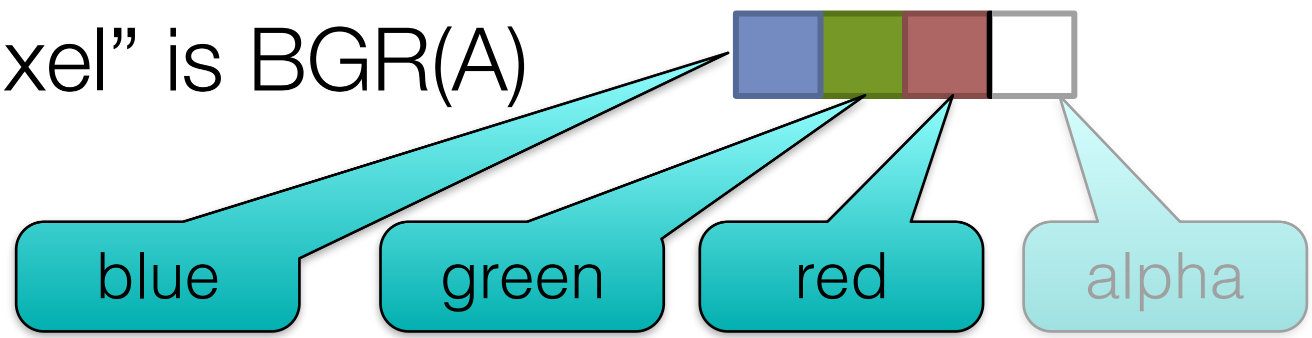


What is image processing

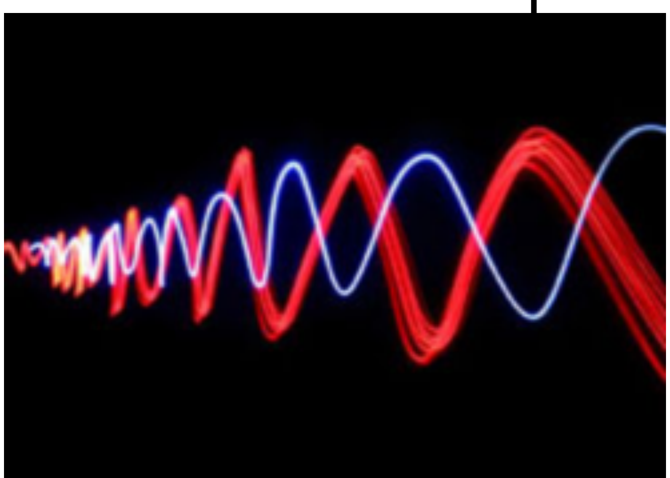
- the **art** and **science** of manipulating pixels
    - combining images (blending or compositing)
    - enhancing edges and lines
    - adjusting contrast, color
    - warping, transformation
    - filtering
    - features extraction

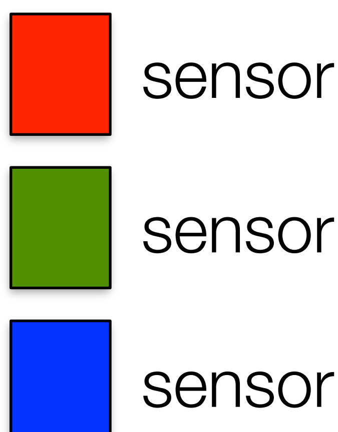
Images as data

- an image can be represented in many ways
- most common format is a matrix of pixels
  - each “pixel” is BGR(A)



- used for capture and display





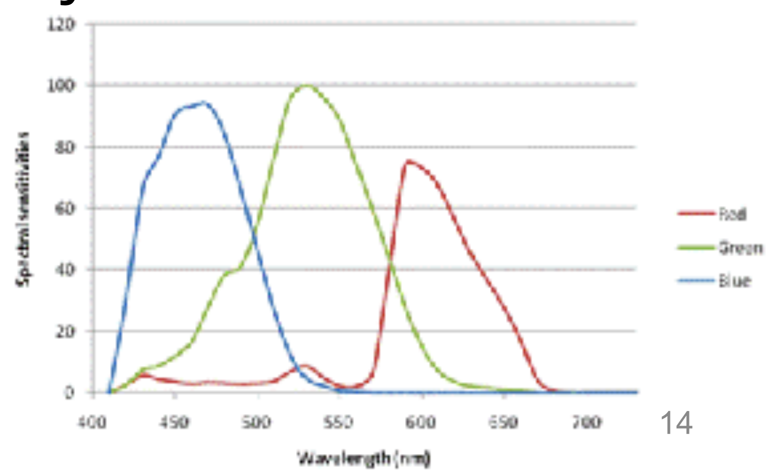


Image Representation

- need a compact representation

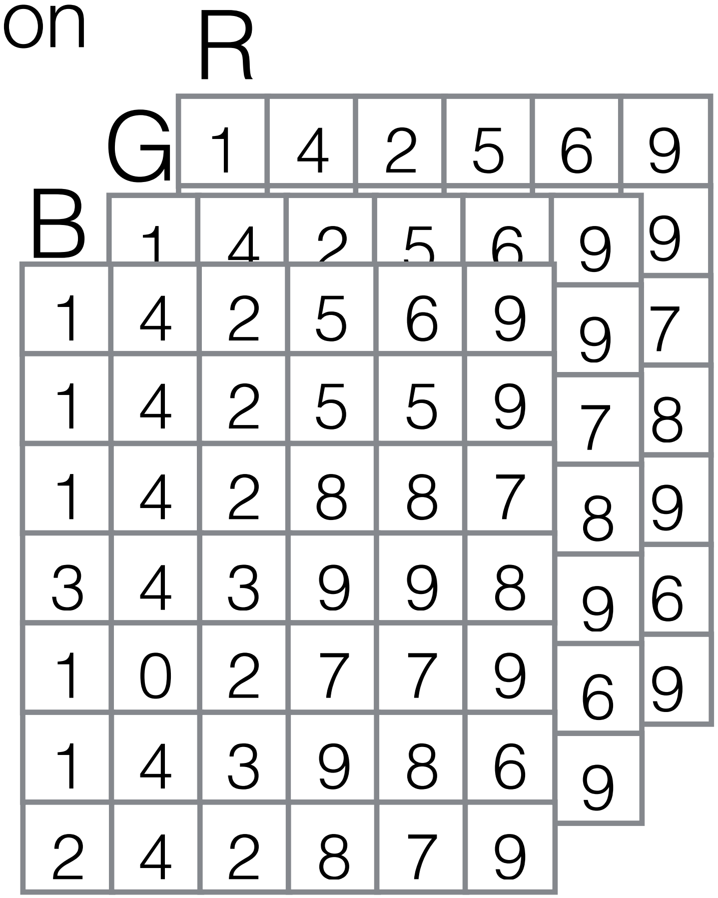
- **grayscale**

  $0.3*R+0.59*G+0.11*B$,
  "luminance"

gray

1	4	2	5	6	9
1	4	2	5	5	9
1	4	2	8	8	7
3	4	3	9	9	8
1	0	2	7	7	9
1	4	3	9	8	6
2	4	2	8	7	9

Numpy Matrix
image[rows, cols]



Numpy Matrix
image[rows, cols, channels]

Image Representation, Features

Problem: need to represent image as table data

1	4	2	5	6	9
1	4	2	5	5	9
1	4	2	8	8	7
3	4	3	9	9	8
1	0	2	7	7	9
1	4	3	9	8	6
2	4	2	8	7	9

Image Representation, Features

Problem: need to represent image as table data

Solution: row concatenation

Row 1	1	4	2	5	6	9	1	4	2	5	5	9	1	4	2	8	8	7	3
Row 2	1	4	2	8	8	7	3	4	3	9	9	8	1	4	2	5	5	9	1

…

Row N	9	4	6	8	8	7	4	1	3	9	2	1	1	5	2	1	5	9	1

Self test: 3a-1

- When vectorizing images into table data, each feature column corresponds to:
  - a. the value (color) of pixel
  - b. the spatial location of a pixel in the image
  - c. the size of the image
  - d. the spatial location and color channel of a pixel in an image

Demo

Dimension Reduction with Images

- Images Representation
- Randomized PCA
- Kernel PCA

`04.Dimension Reduction and Images.ipynb`

Features of Images

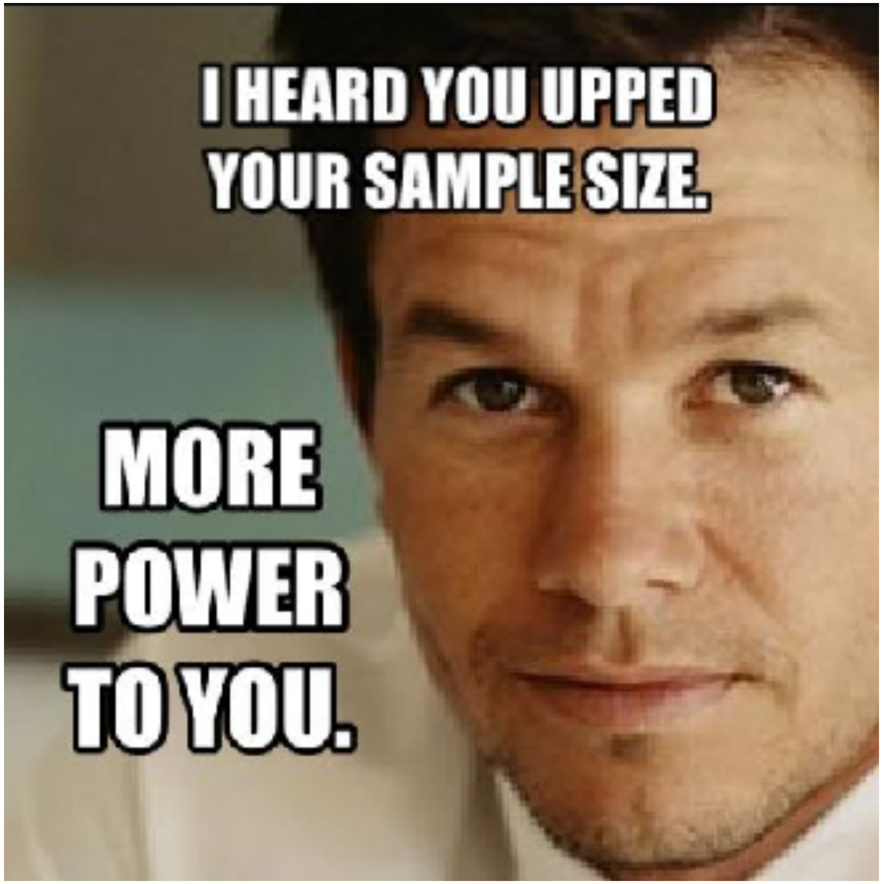
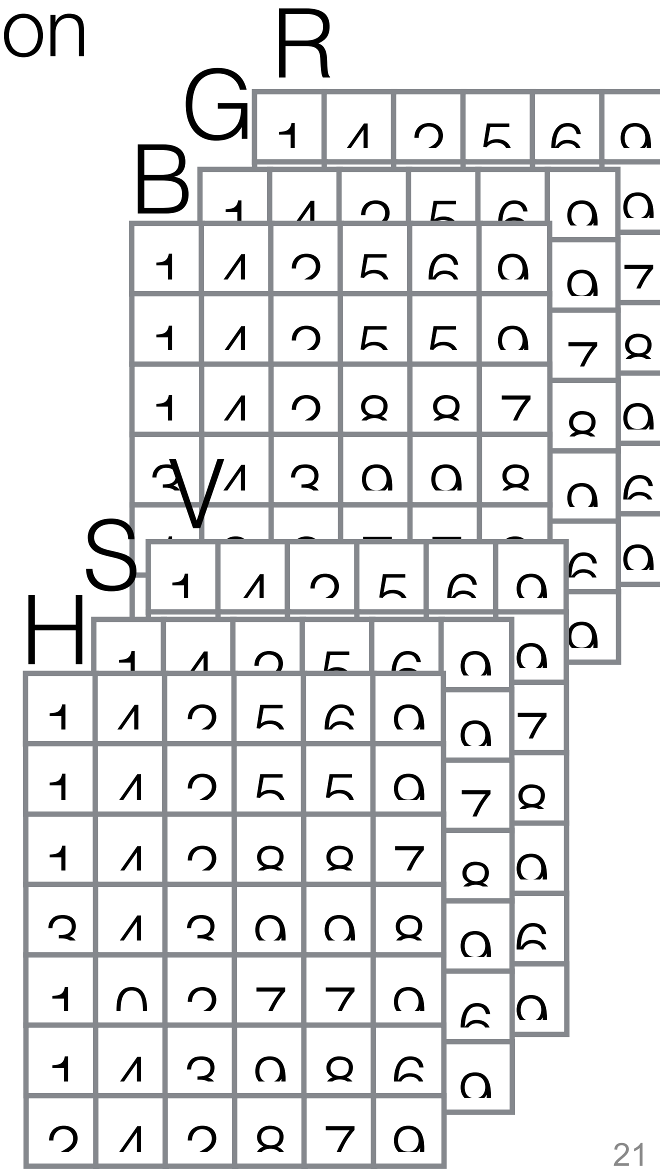
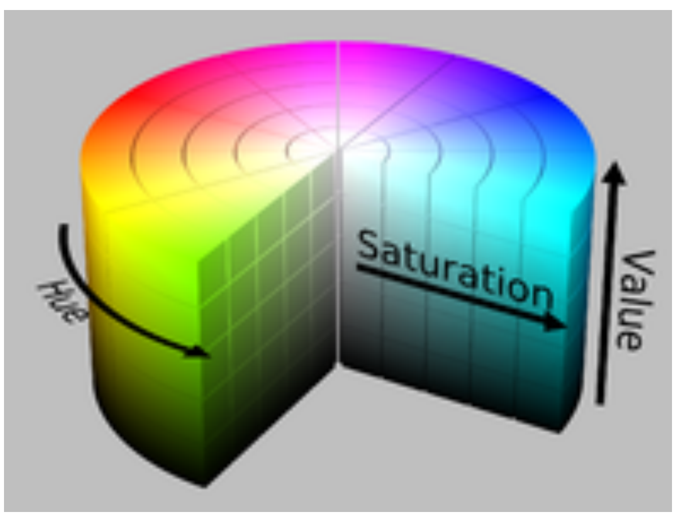


Image Representation

- need a compact representation



- **hsv**
    - what we perceive as color (ish)
        - hue: the color value
        - saturation: the richness of the color relative to brightness
        - value: the gray level intensity

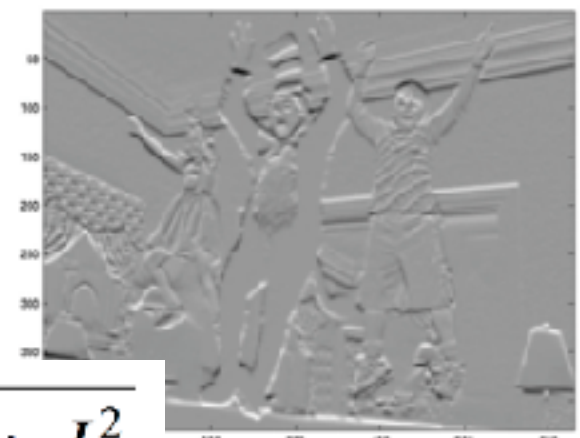


Common operations

- the gradient (2D derivative)

I filter J_x

filter J_y

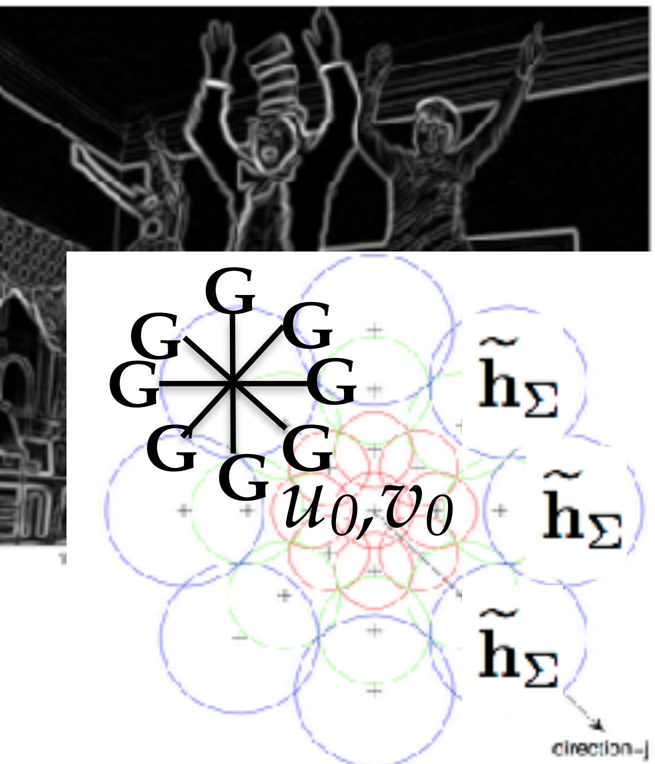


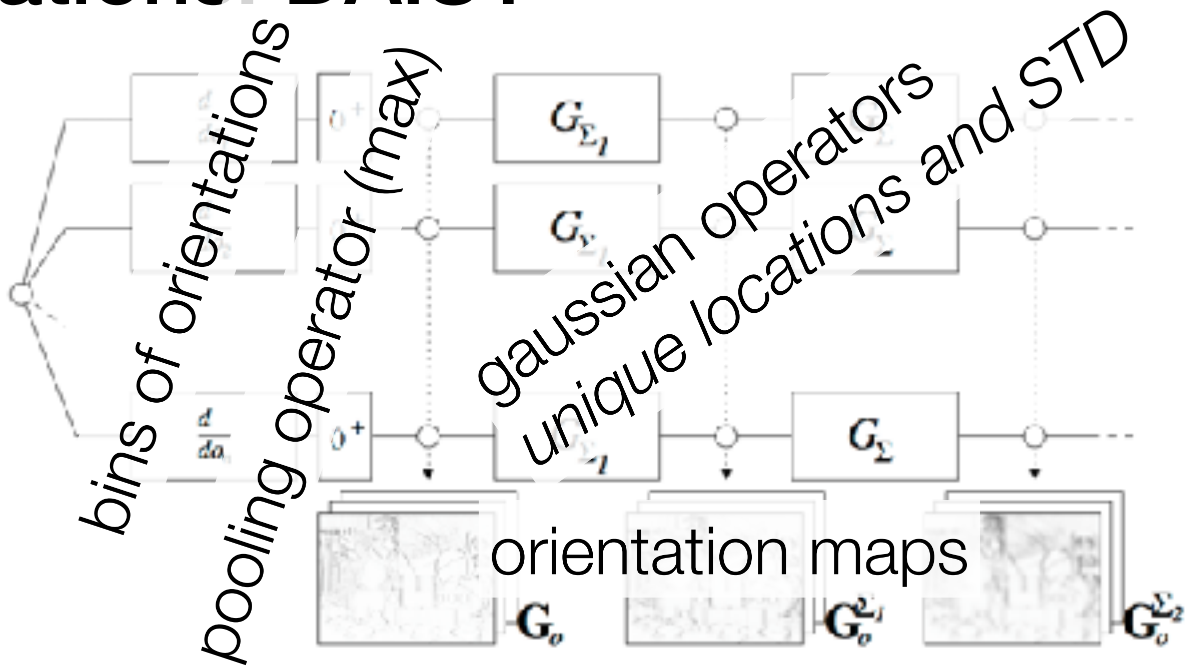
$$\|\nabla J\| = \sqrt{J_x^2 + J_y^2}$$

$$O = \tan^{-1}(Jx / Jy)$$

images: Jianbo Shi, Upenn

Common operations: DAISY



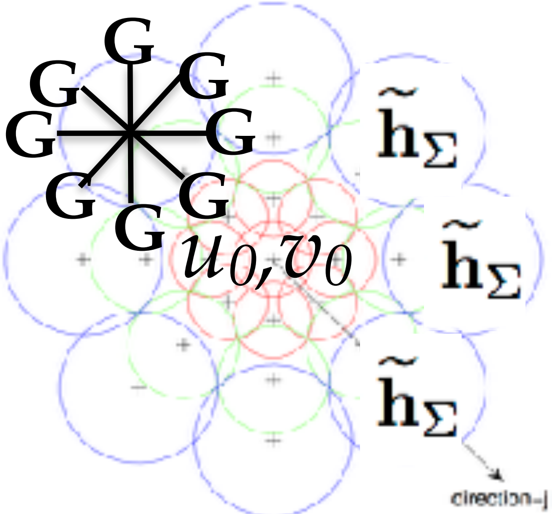


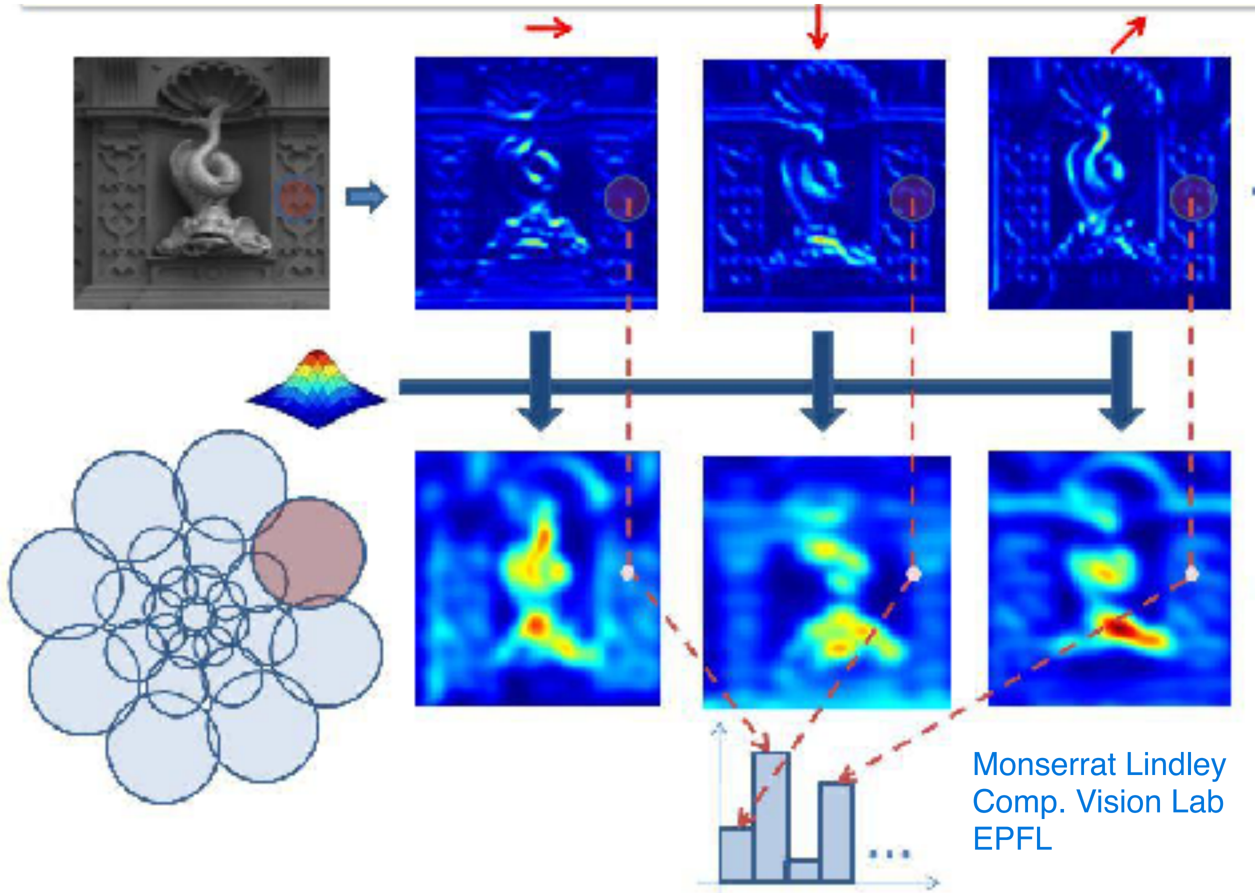
take normalized histogram at point u,v

$$\tilde{\mathbf{h}}_\Sigma(u,v) = \left\| \left[\mathbf{G}_1^\Sigma(u,v), \ldots, \mathbf{G}_H^\Sigma(u,v) \right]^\top \right\|$$

$$\mathcal{D}(u_0,v_0) =$$
$$\Big[\quad \tilde{\mathbf{h}}_{\Sigma_1}^\top(u_0,v_0),$$
$$\tilde{\mathbf{h}}_{\Sigma_1}^\top(\mathbf{l}_1(u_0,v_0,R_1)), \cdots, \tilde{\mathbf{h}}_{\Sigma_1}^\top(\mathbf{l}_T(u_0,v_0,R_1)),$$
$$\tilde{\mathbf{h}}_{\Sigma_2}^\top(\mathbf{l}_1(u_0,v_0,R_2)), \cdots, \tilde{\mathbf{h}}_{\Sigma_2}^\top(\mathbf{l}_T(u_0,v_0,R_2)),$$
$$\ldots$$

Tola et al. "Daisy: An efficient dense descriptor applied to wide-baseline stereo." Pattern Analysis and Machine Intelligence, IEEE Transactions





Monserrat Lindley
Comp. Vision Lab
EPFL

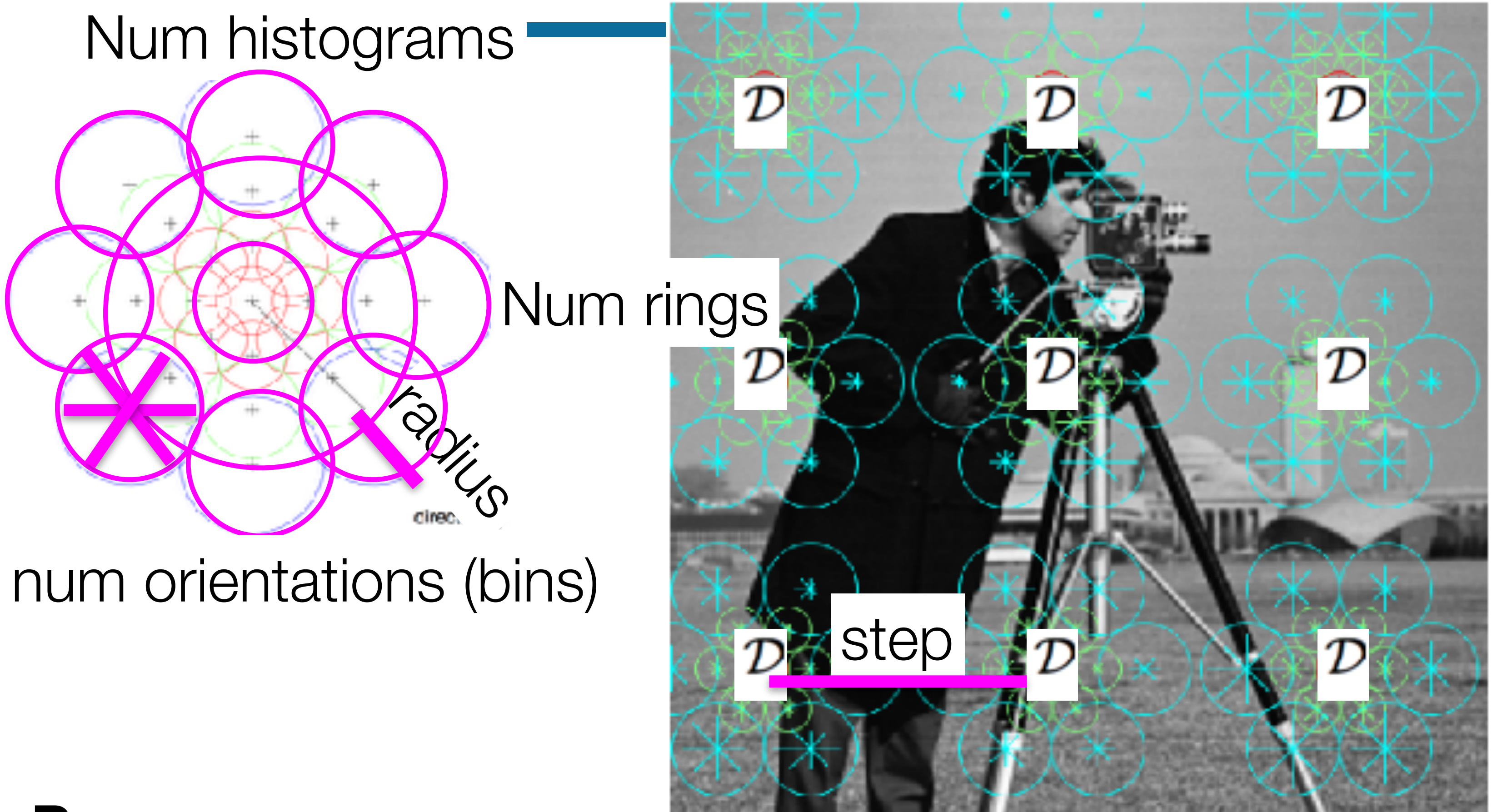
take normalized histogram at point u,v

$$\tilde{\mathbf{h}}_\Sigma(u,v) = \left\| \left[\mathbf{G}_1^\Sigma(u,v),\ldots,\mathbf{G}_H^\Sigma(u,v)\right]^\top \right\|$$

$$\begin{aligned}
\mathcal{D}(u_0,v_0) = \\
[\ \ & \tilde{\mathbf{h}}_{\Sigma_1}^\top(u_0,v_0), \\
& \tilde{\mathbf{h}}_{\Sigma_1}^\top(\mathbf{l}_1(u_0,v_0,R_1)),\cdots,\tilde{\mathbf{h}}_{\Sigma_1}^\top(\mathbf{l}_T(u_0,v_0,R_1)), \\
& \tilde{\mathbf{h}}_{\Sigma_2}^\top(\mathbf{l}_1(u_0,v_0,R_2)),\cdots,\tilde{\mathbf{h}}_{\Sigma_2}^\top(\mathbf{l}_T(u_0,v_0,R_2)),
\end{aligned}$$

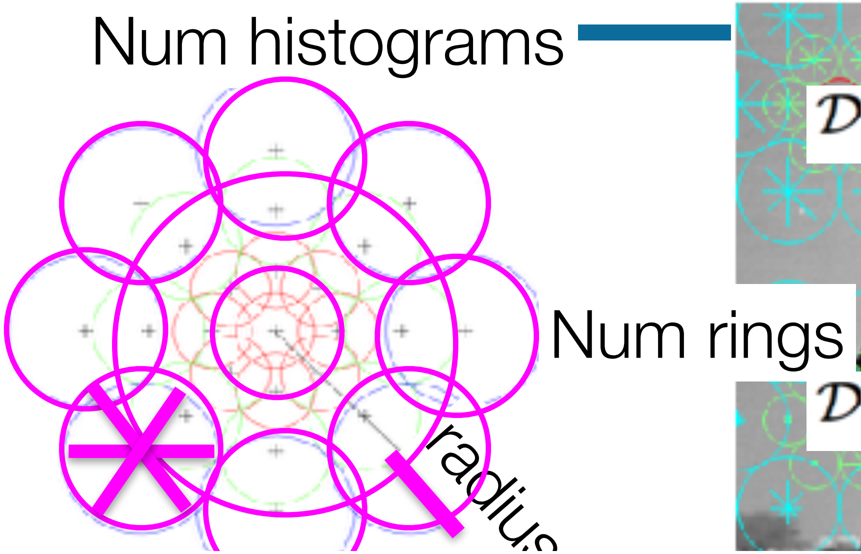
Tola et al. "Daisy: An efficient dense descriptor applied to wide-baseline stereo." Pattern Analysis and Machine Intelligence, IEEE Transactions

Common operations: DAISY

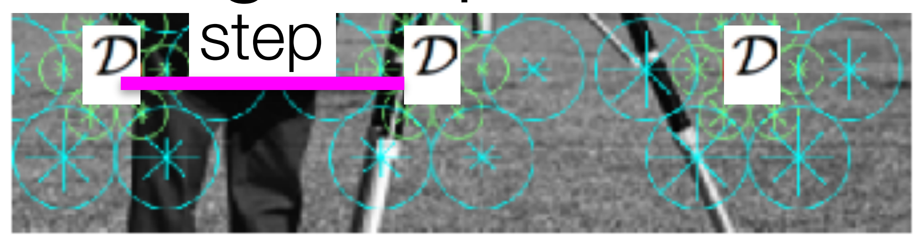


Params:
step, radius, num rings,
num histograms per ring,
orientations per histogram

Common operations: DAISY



Bag of Features Image Representation

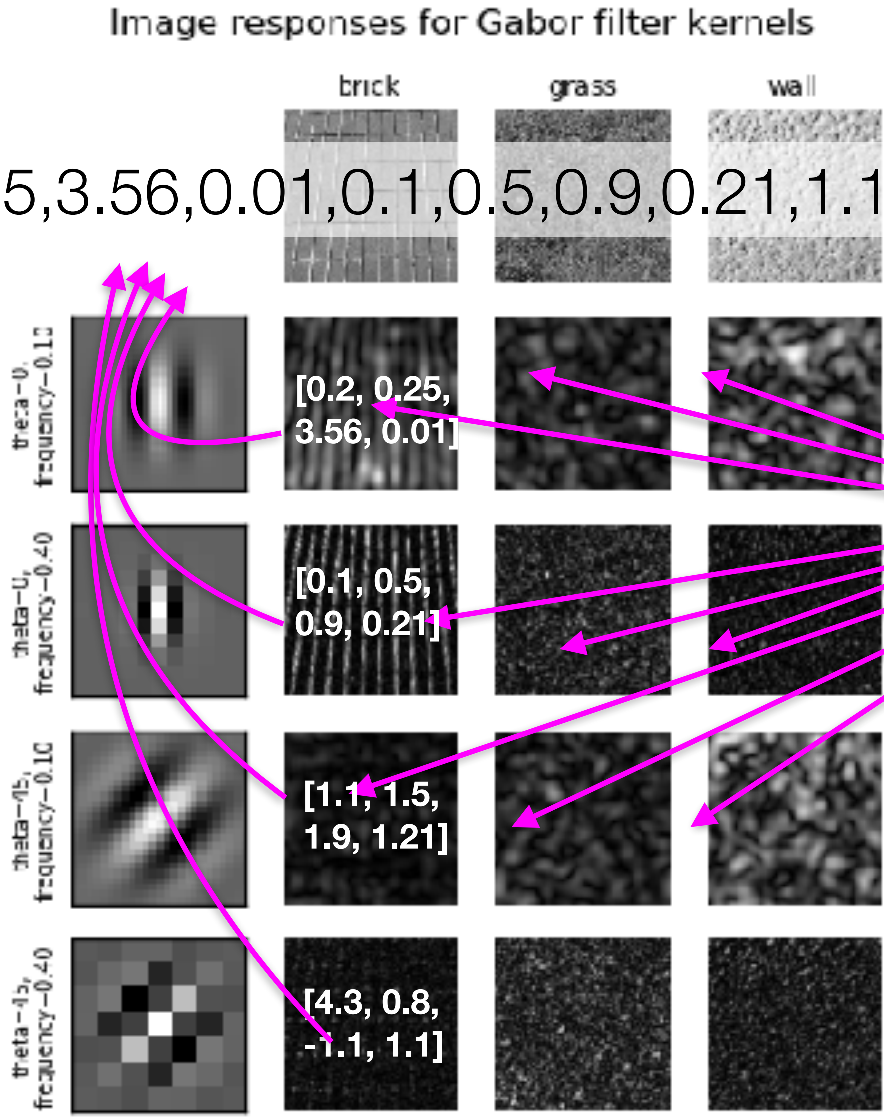


Params:
step, radius, num rings,
num histograms per ring,
orientations per histogram

Common operations: Gabor filter Banks (if time)

- common used for texture classification

[0.2,0.25,3.56,0.01,0.1,0.5,0.9,0.21,1.1,1.5,1.9,1.21,4.3,0.8,-1.1,1.1]



mean
std
skew
kurtosis

Concatenate statistics of different filters into one feature vector

Demo

More Image Processing

Gradients

DAISY

Gabor Filter Banks

Other Tutorials:

http://scikit-image.org/docs/dev/auto_examples/

For Next Lecture

- Work on your text datasets!
- **Next Time**: No Class, Project work day
- **Next Week**: In-Class Assignment One!!!