MULTIMEDIA CONTENT ANALYSIS

Kevin McGuinness

Overview of Material

- Introduction to multimedia content analysis
- Applications
- Describing image content
  - Global features
  - Local features and interest points
  - Aggregating local features
- Matching and distance measures
- Learning and classification

Textbooks and Resources

- Textbooks
  - » Computer vision: models, inference, and learning, by Simon Prince
    - http://www.computervisionmodels.com
  - » Computer vision: algorithms and applications, by Richard Szeliski
    - http://szeliski.org/Book/
  - » The elements of statistical learning, by Trevor Hastie et al.
    - http://www-stat.stanford.edu/~tibs/ElemStatLearn/
- Material
  - » Distance functions and metric learning (ECCV 2010 tutorial)
    - http://www.seas.upenn.edu/~ofirpele/DFML_ECCV2010_tutorial/

Multimedia content analysis

INTRODUCTION

INTRODUCTION

- Challenge in image compression
  - » How do we represent images in a way that amenable to quantization and coding, without compromising reconstruction quality?
  - » Transform coding, energy compaction, quantization
  - » Remove information that doesn't matter to human visual perception
  - » Retain information that does

INTRODUCTION

- Challenge in image matching and classification
  - » How do we represent multimedia in a way that allows us to determine if two multimedia objects are similar?
  - » Remove information that doesn't matter when determining if objects are similar
  - » Retain information that does
- But:
  - » What information matters?
  - » What do we mean by similar?

INTRODUCTION

- Need to define two things:
  - » **Descriptors**: vector of numbers that describe content
  - » **Similarity criteria**: how to compare two descriptors to see if describe similar content

- If we want to classify content, we also need a **learning algorithm**
  - » Too difficult to say by hand what makes visual attributes are important
  - » Learning algorithm allows you to give examples and have machine automatically determine what is important

Multimedia content analysis

APPLICATIONS

APPLICATIONS

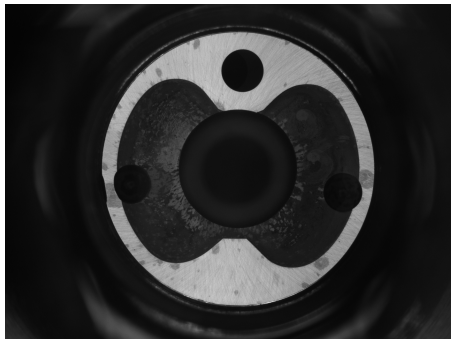
- Before we go into feature extraction and machine learning, let's look at some applications
- Image and video matching and classification have many applications
- All of the following are still active research areas
- Many are now mature enough for practical use

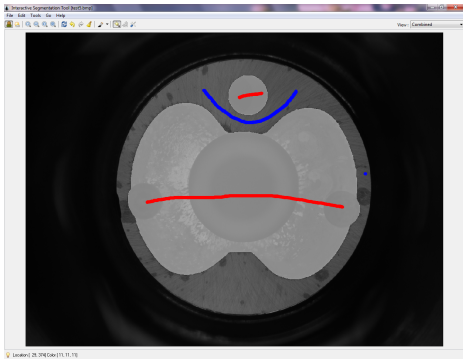
Content Based Information Retrieval

- Given descriptors and a suitable similarity measure, we can perform content based multimedia IR
- Find similar images:
  - » Extract descriptors for all images in collection (can be done in advance, offline)
  - » Extract descriptor for the query image
  - » Compute similarity between query image and all images in collection by comparing their descriptors using the similarity measure
  - » Sort images by their similarity
- Can be done for images or video

Machine Vision

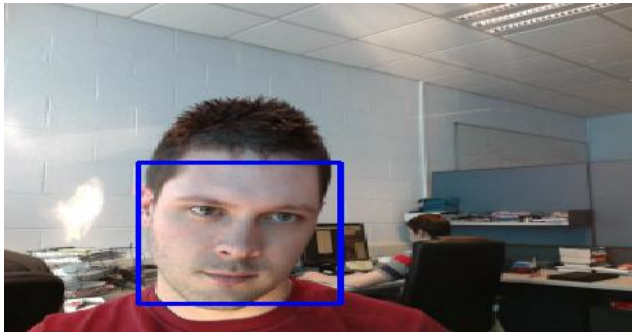
- Machine vision and automated visual inspection of parts
  - » Detecting flaws in manufactured hardware
  - » Capture images using camera
  - » Extract features, compute descriptors
  - » Given examples, use machine learning to infer a function that discriminates between good and bad parts





Detection and Classification

- People and faces
  - » Pass a fixed sized window over the image
  - » Extract features for each window
  - » Classify as face or non-face based on learned model
  - » Similar for people
- Viola-Jones type face detectors included in many modern cameras
  - » Allows auto focus
- Social media sites
  - » Facebook, etc.
  - » User tagging

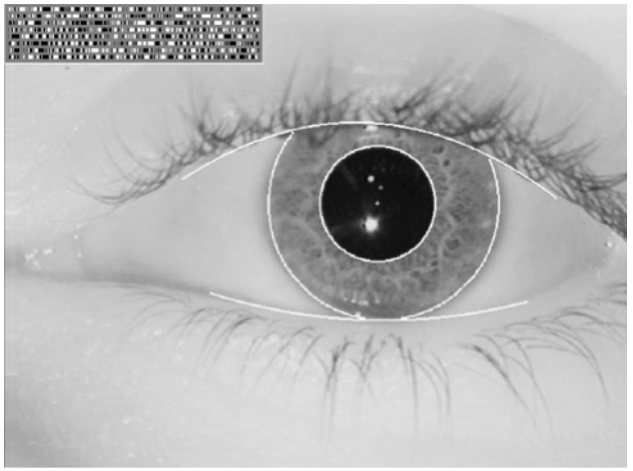


Surveillance Event Detection

- Lots of CCTV cameras
- Use feature extraction and descriptors to detect unusual events
- Machine learning and classification algorithms can be used to learn from examples

BIOMETRIC IDENTIFICATION

- Iris recognition
- Fingerprints
- Palm prints
- Face recognition
- Recognition based on ear characteristics!



Detection of Copyright Infringement

- Find videos containing similar content to copyrighted content
- Must be robust to different transforms
  - » Rotation, scaling, picture-in-picture
- Should not be overly computationally expensive

Multimedia content analysis

FEATURE EXTRACTION

Describing Image Content

feature extraction

N-dimensional descriptors

1.20	0.24	7.22	3.21	…	1.28

0.28	1.23	8.32	1.32	…	5.21

feature extraction

feature values

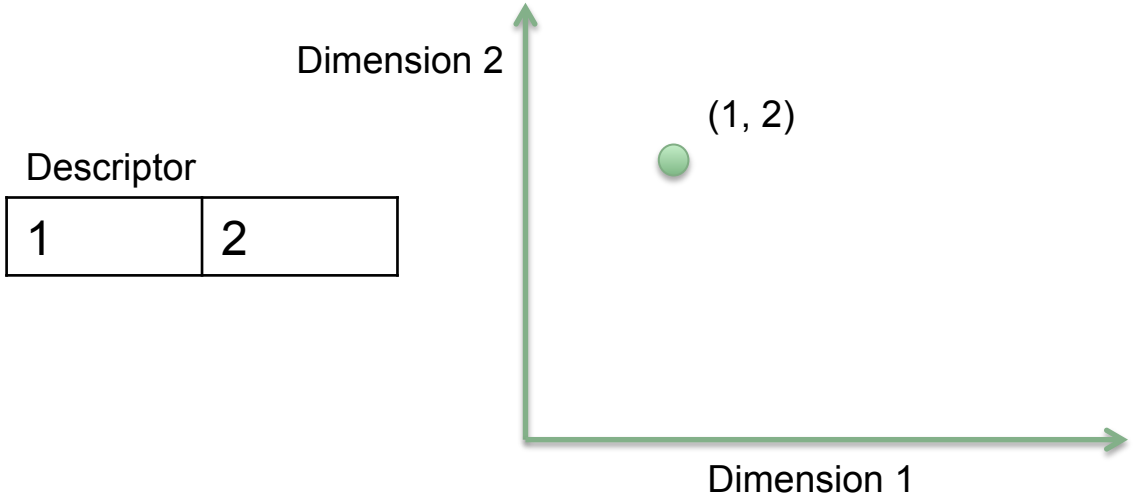
DESCRIBING IMAGE CONTENT

1.20	0.24	7.22	3.21	…	1.28

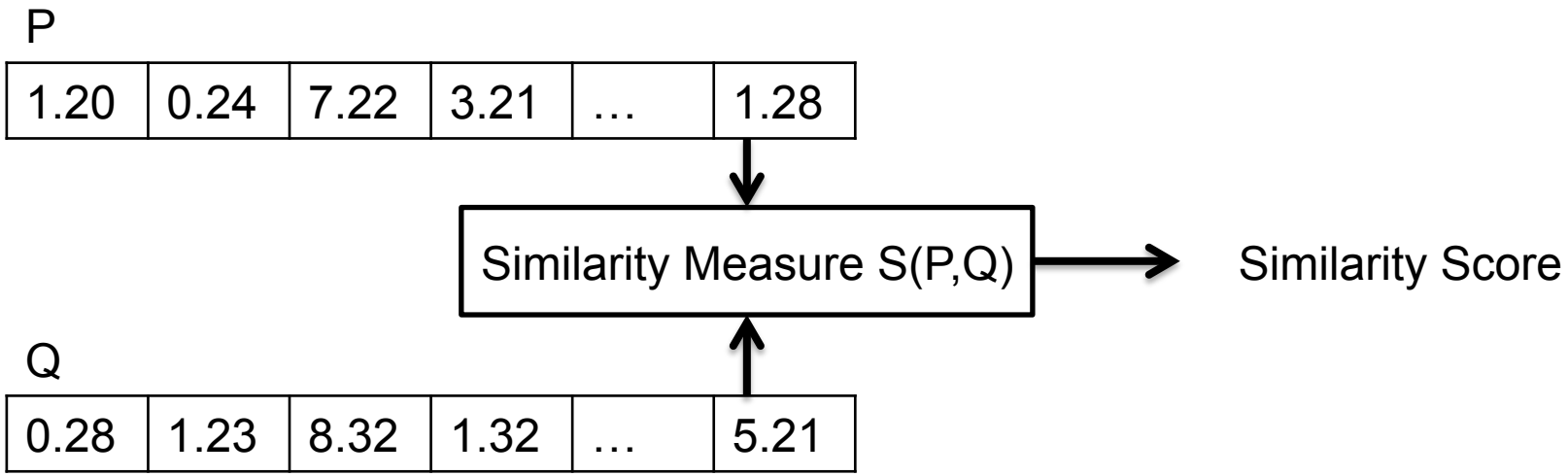
- What is a descriptor?
  - » Just a fixed length array of numbers
  - » Describes some aspect of content
  - » Number of values = **dimension**
  - » Also known as **feature vectors**
  - » Variable length descriptors are called “signatures”

Descriptors as Points in N-D Space

- A descriptor of length N can be thought of as a point or vector in N dimensional space
  - » Length 2 descriptor is a point in 2D space
  - » Length 3 descriptor is a point in 3D space
- Important for machine learning applications



COMPARING DESCRIPTORS



For example:

Euclidean (L2) distance

$$D_{L1}(P,Q) = \sqrt{\sum_{i=1}^{D}(P_i - Q_i)^2}$$

$$S(P,Q) = \frac{1}{1 + D_{L1}(P,Q)}$$

Distance to similarity

More on distance metrics later…

Feature Extraction

- Need a method derive useful descriptors from image content
  - » Simplest possible approach would be to use the raw pixel values themselves
- But:
  - » Descriptors must be of a fixed size for many distance measures to work
  - » Images may come in different sizes

USING THE RAW PIXEL VALUES

- Strategy:
  - » Resample image to a fixed dimension (eg. 100 x 100 pixels)
    - Using suitable interpolation strategy (bilinear, bicubic)
  - » Unravel pixels in the image to produce a descriptor
    - Just take values row by row or column by column
    - If retaining color, the descriptor dimension would be 100 x 100 x 3 = 30,000
    - If just using gray scale values, descriptor dimension 100 x 100 = 10,000
  - » Use suitable distance measure to compare images

PROBLEMS WITH RAW IMAGE PIXELS

- Descriptor is of high dimension
  - » Expensive to store without compression
  - » Expensive to compute distance or similarity between descriptors
- Descriptor not invariant to many types of changes that we don’t care about
  - » Not transform invariant

EXAMPLES

Brighten + 0.08

Euclidean distance between descriptors: 21.38

(descriptor 10,000 grayscale intensity [0..1])

EXAMPLES

Translate horizontal + 10 pixels

Euclidean distance between descriptors: 18.91

(descriptor 10,000 grayscale intensity [0..1])

EXAMPLES

Flip left and right

Euclidean distance between descriptors: 19.51

(descriptor 10,000 grayscale intensity [0..1])

TRANSFORMS AND INVARIANCE

- Transforms:
  - » Geometric:
    - Translation
    - Scale
    - Sheer
  - » Photometric:
    - Global illumination changes
    - Local illumination changes
- Geometric transforms often occur due to camera position changes
- Photometric changes due to lighting conditions
  - » Day and night
  - » Inside and outside
  - » Lamps and spotlights

TRANSFORMS AND INVARIANCE

- Invariance:
  - » Robustness to transforms
  - » Descriptor stays the same (or almost the same) even when the image is transformed
- Types of invariance:
  - » Translation invariance (geometric)
  - » Scale invariance (geometric)
  - » Affine invariance (geometric)
  - » Local/global illumination invariance (photometric)

HISTOGRAMS

- Capture global distribution statistics
- Invariant to small geometric changes

Generic Histogram algorithm:

```
Quantize values to integers from 0..N
Initialize histogram H[i] = 0 for i from 0..N
For each value v:
    H[v] += 1
```

INTENSITY HISTOGRAMS

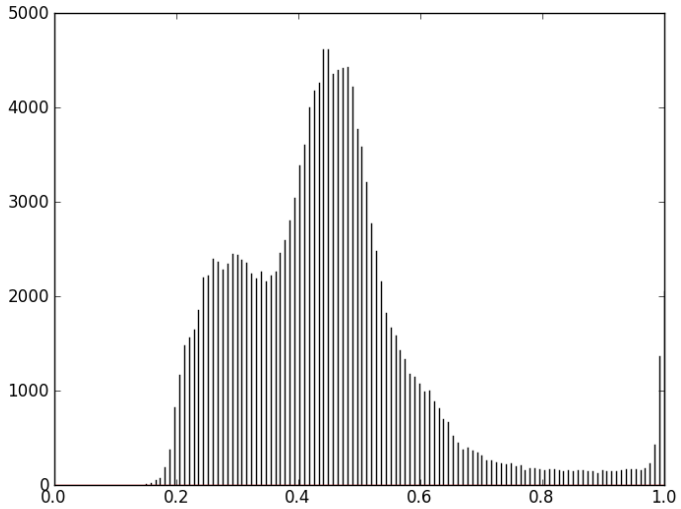
- Compute histograms directly on pixel intensity values
  - » I.e. their grayscale values

Intensity Histogram algorithm:

```
Quantize intensity values to integers from 0..N
Initialize histogram H[i] = 0 for i from 0..N
For each pixel intensity value v:
    H[v] += 1
```

INTENSITY HISTOGRAMS

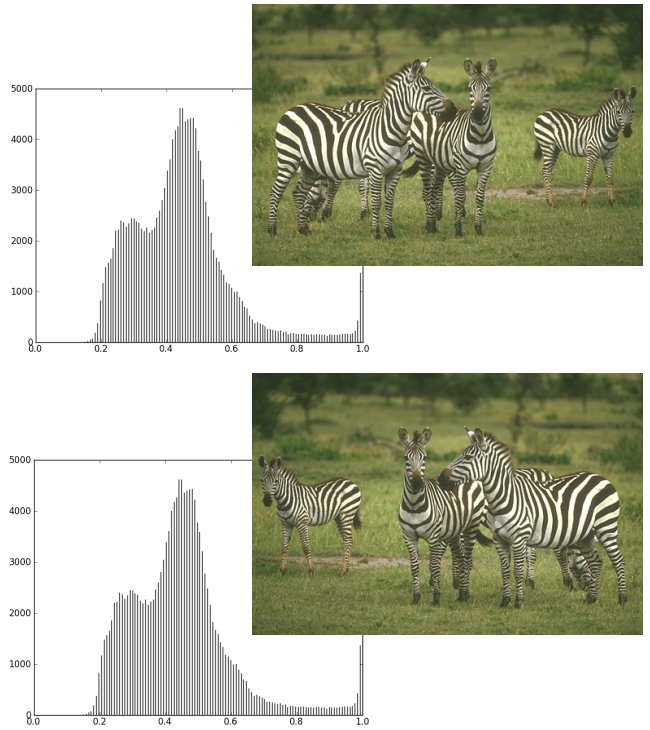
- E.g. transform to grayscale, quantize between 0 and 255, compute histogram



0	0	…	200	…	100

INTENSITY HISTOGRAMS

- Invariances:
  - » Several geometric transforms
    - translate,
    - flip,
    - small rotations



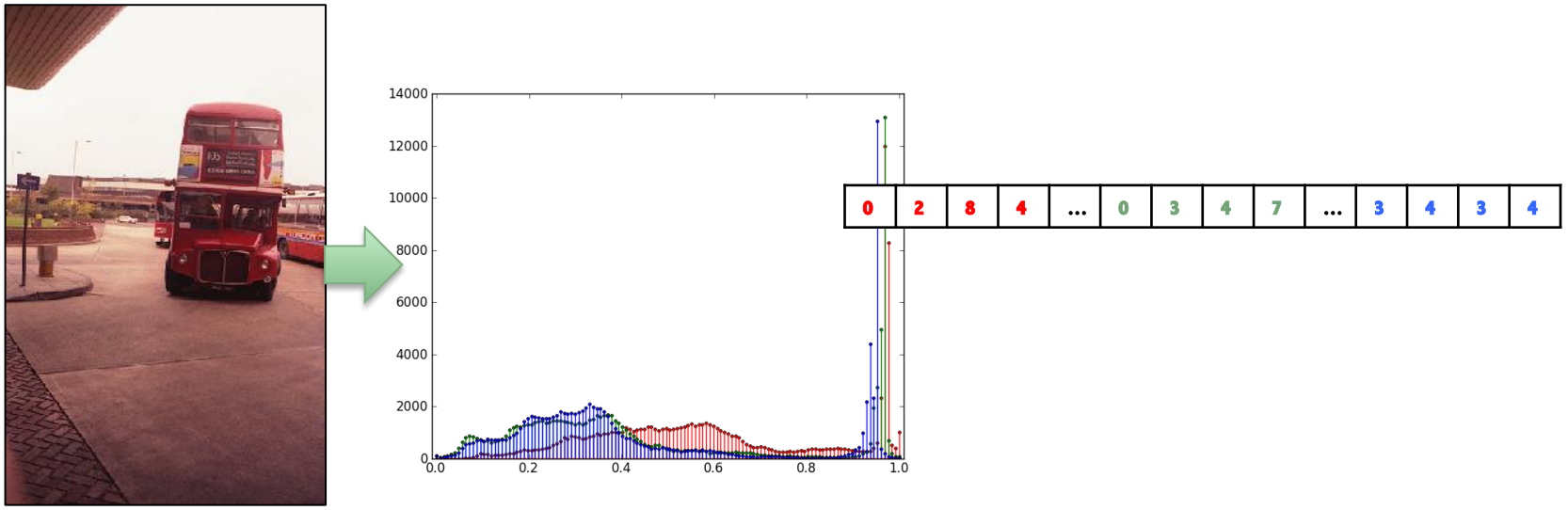
- Generally not invariant to photometric transforms or camera angle changes (zooms, pans, etc.)

COLOR HISTOGRAMS

- Intensity histograms discard all color information
  - » Color can be important when looking for images that are visually similar
  - » For intensity histograms: bright red = bright blue = bright green
- Color histograms:
  - » 2 kinds
    - Marginal color histograms
    - Joint color histograms

MARGINAL COLOR HISTOGRAMS

- Compute histograms on R, G, and B channels independently
- Concatenate resulting 3 histograms to form descriptor



JOINT COLOR HISTOGRAM

- Treat color triples (R,G,B) as a 3-D space
- Quantize each dimension to N bins
- Form N x N x N histogram
  - » H[R, G, B] += 1
- Advantages
  - » Captures correlations in RGB components
- Disadvantages
  - » Descriptor can be very large unless heavy quantization is used:
    - 255^3 = 16,581,375 !
    - 8^3 = 512
  - » Most of the RGB space will be empty

Edges and Gradients

- Color and intensity still sensitive to photometric transforms
  - » Can alleviate this somewhat by normalizing image range
- Another approach is to use image gradients
  - » Derivatives of the image in the horizontal and vertical directions
  - » Correspond to **edges** in the image
  - » Studies have shown that edges are very important for human perception

Image Gradients

- If image were continuous real valued function…

$$\frac{\partial I}{\partial x} = \partial_x(I)$$

Horizontal partial derivative:
How image changes as we move from left to right

$$\frac{\partial I}{\partial y} = \partial_y(I)$$

Vertical partial derivative:
How image changes as we move from top to bottom

Image Gradients

These are all functions that can be evaluated at any (x,y)

Gradient Vector

$$\nabla(I) = \begin{bmatrix} \partial_x(I) \\ \partial_y(I) \end{bmatrix}$$

Gradient Direction

$$\theta = \arctan\left(\frac{\partial_y}{\partial_x}\right)$$

Gradient Magnitude

$$\alpha = \sqrt{\partial_x^2 + \partial_y^2}$$

Discrete Case

- Images are **not** continuous functions
- Need discrete approximation of gradient
  - » First order approximation

$$\partial_x (I)_{xy} \approx I(x, y) - I(x-1, y)$$
$$\partial_y (I)_{xy} \approx I(x, y) - I(x, y-1)$$

  - » Can be implemented via a convolution
    - Horizontal kernel [1,-1]
    - Vertical kernel $[1, -1]^T$

CONVOLUTION

- Briefly:
  - » Pass a rectangular window over each pixel in the image
  - » Set output pixel to linear combination of neighbors with weights given by mask
  - » Example: box filter (blurs an image)

kernel

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

For each pixel
Output pixel =
    1/9 * pixel at top left +
    1/9 * pixel above +
    1/9 * pixel at top right +
    …
    1/9 * pixel at bottom right

Gradients Using Convolution

First order approximation kernels

-1	1

-1
1

Second order approximation kernels

-1	0	1

-1
0
1

- Can derive from Taylor series expansion
- Symmetric around pixel
- More accurate in practice

Sobel kernel

-1	0	1
-2	0	2
-1	0	1

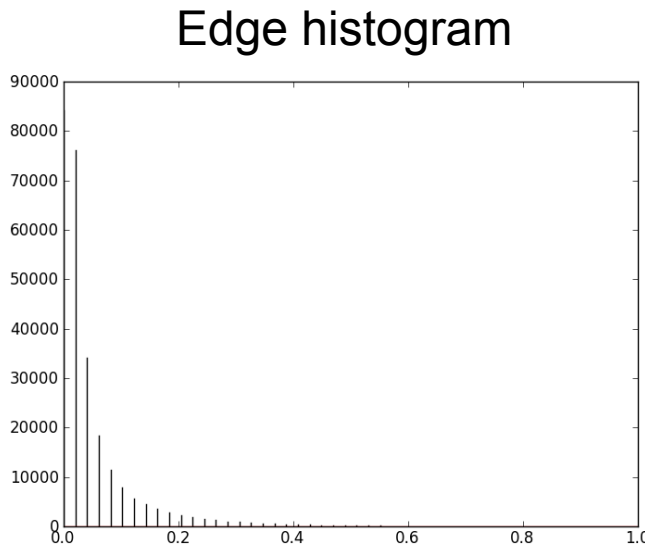
- Vertical version is transpose
- Symmetric around pixel
- Incorporates a little blurring
  - less sensitive to noise

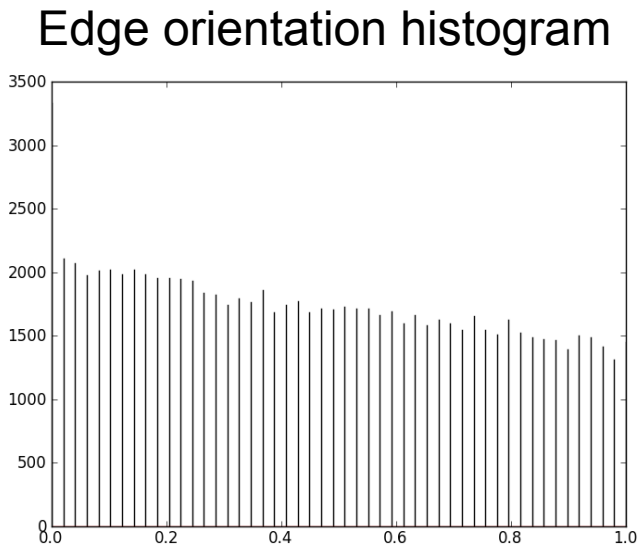
Determining The Gradient

- Convolve image with horizontal and vertical kernels to approximate gradient vector
  - » Magnitude and direction given by previous formulas
  - » Usually performed on intensity (luminance) values

EDGE HISTOGRAMS

- Compute histograms of the gradient
  - » On the gradient magnitude (edge histogram)
  - » On the gradient orientation (edge orientation)





SPATIALLY LOCALIZED HISTOGRAMS

- Global histograms throw away lots of positional information
- Block based histograms divide image up into blocks and compute histograms on each block
- Resulting block histograms are concatenated to form the final descriptor

LIMITATIONS OF GLOBAL FEATURES

- Discriminability
  - » Only account for global characteristics, not local
  - » Very different images can present similar histograms
- Affine invariance
  - » Zoom, camera angle
- Objects
  - » Objects in different backgrounds
  - » Clutter

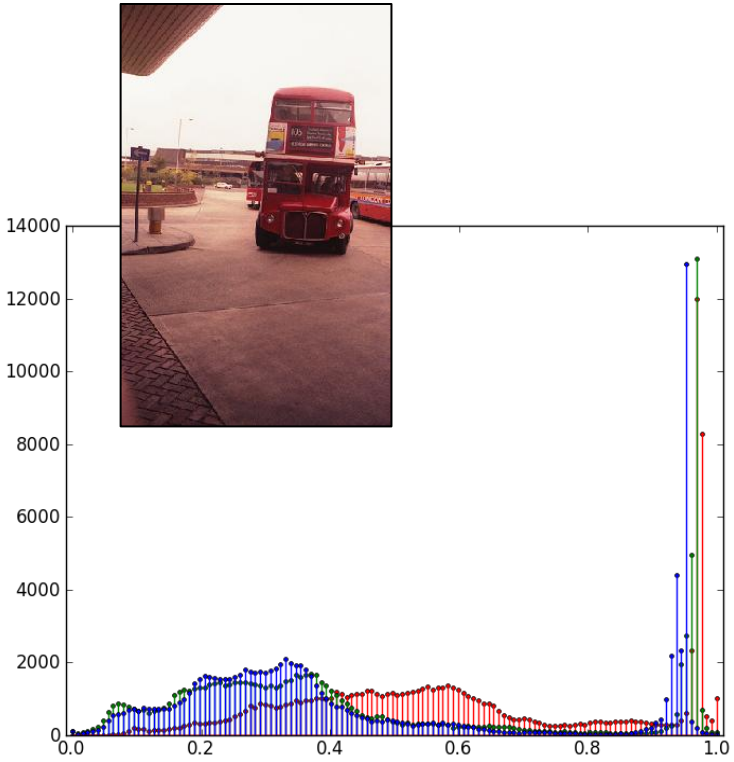
Multimedia content analysis

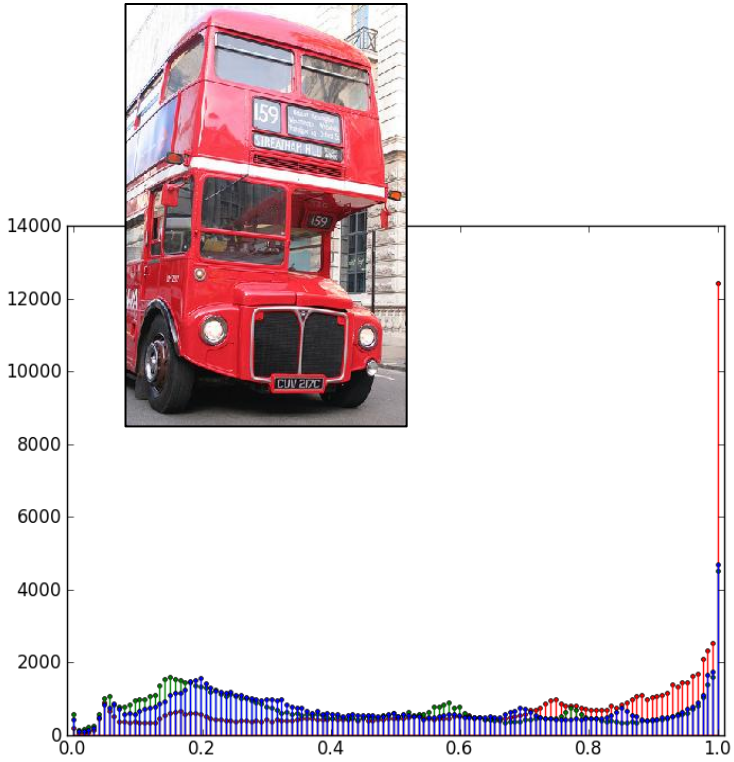
LOCAL FEATURES AND INTEREST POINTS

Local Features and Interest points

- Global image characteristics may change a lot, but still contain the same, or similar objects

LOCAL FEATURES AND INTEREST POINTS





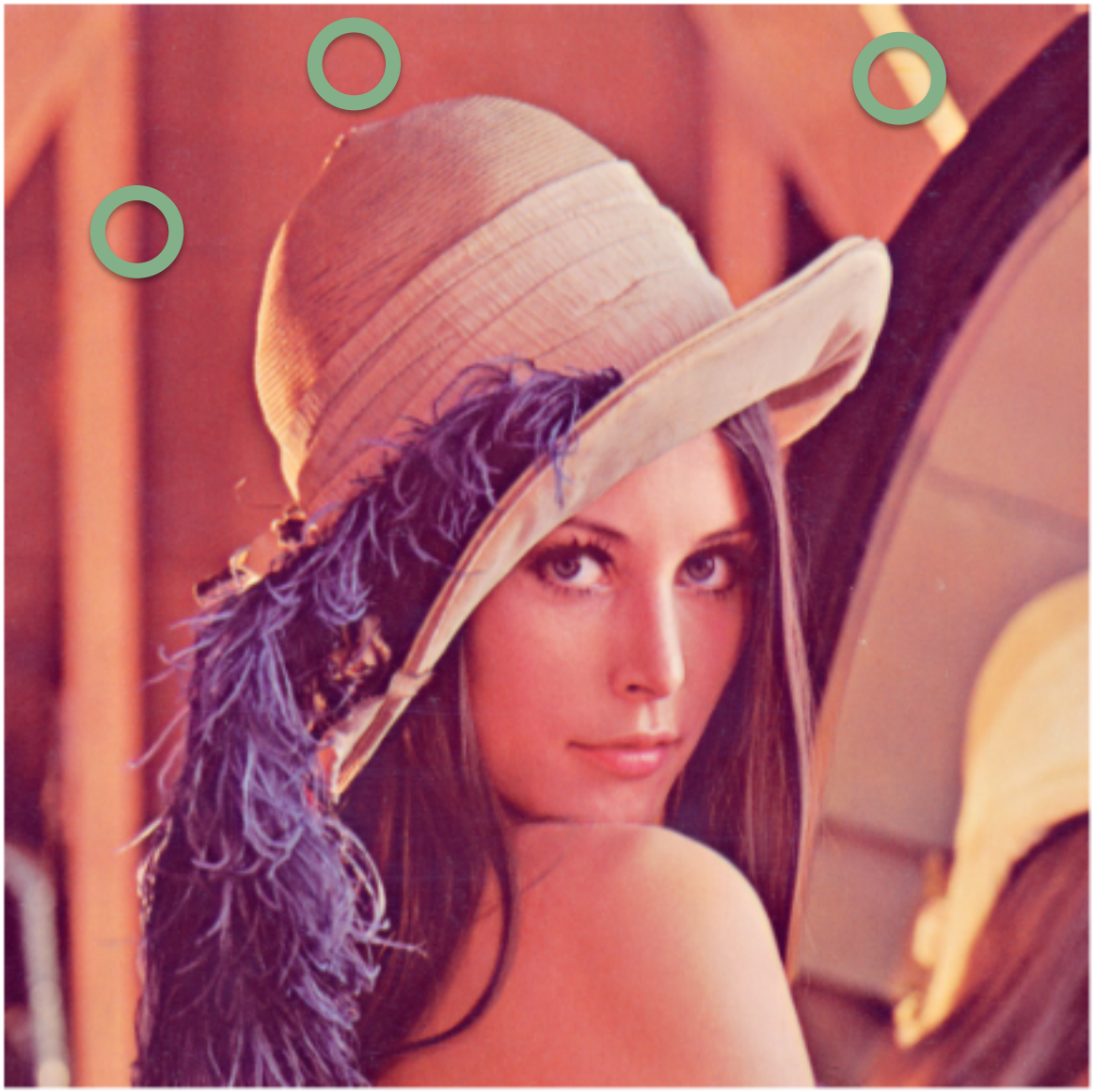
Images contain similar objects, but global descriptors very different

Local Features and Interest points

- Characteristics of some local regions in one image may still be similar to characteristics of some local regions in another

INTEREST POINTS

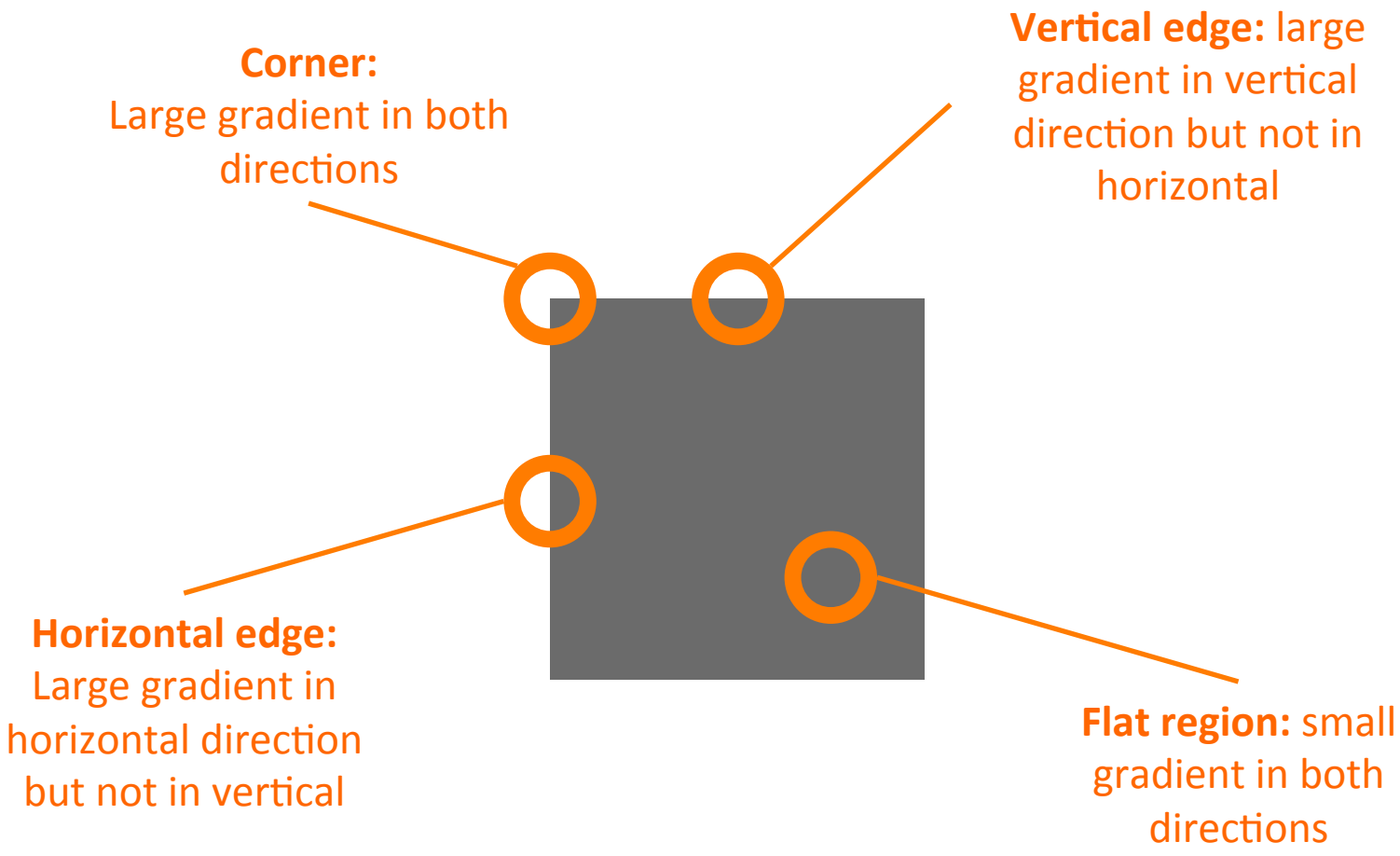
- If we had an algorithm that could produce a set of points from *interesting* regions in an image, then we could describe these regions using invariant descriptors and match them against other images
- What do we mean by interesting?
  - » Sparse
  - » Reproducible



THE HARRIS CORNER DETECTOR

- Detects corner points
  - » Sparse
    - far less corner points than image pixels
  - » Reproducible
    - relatively insensitive to geometric and photometric transformations
- Based on directional gradients
  - » Corner points are points in the image where the **is large gradient** magnitude in **two orthogonal directions**

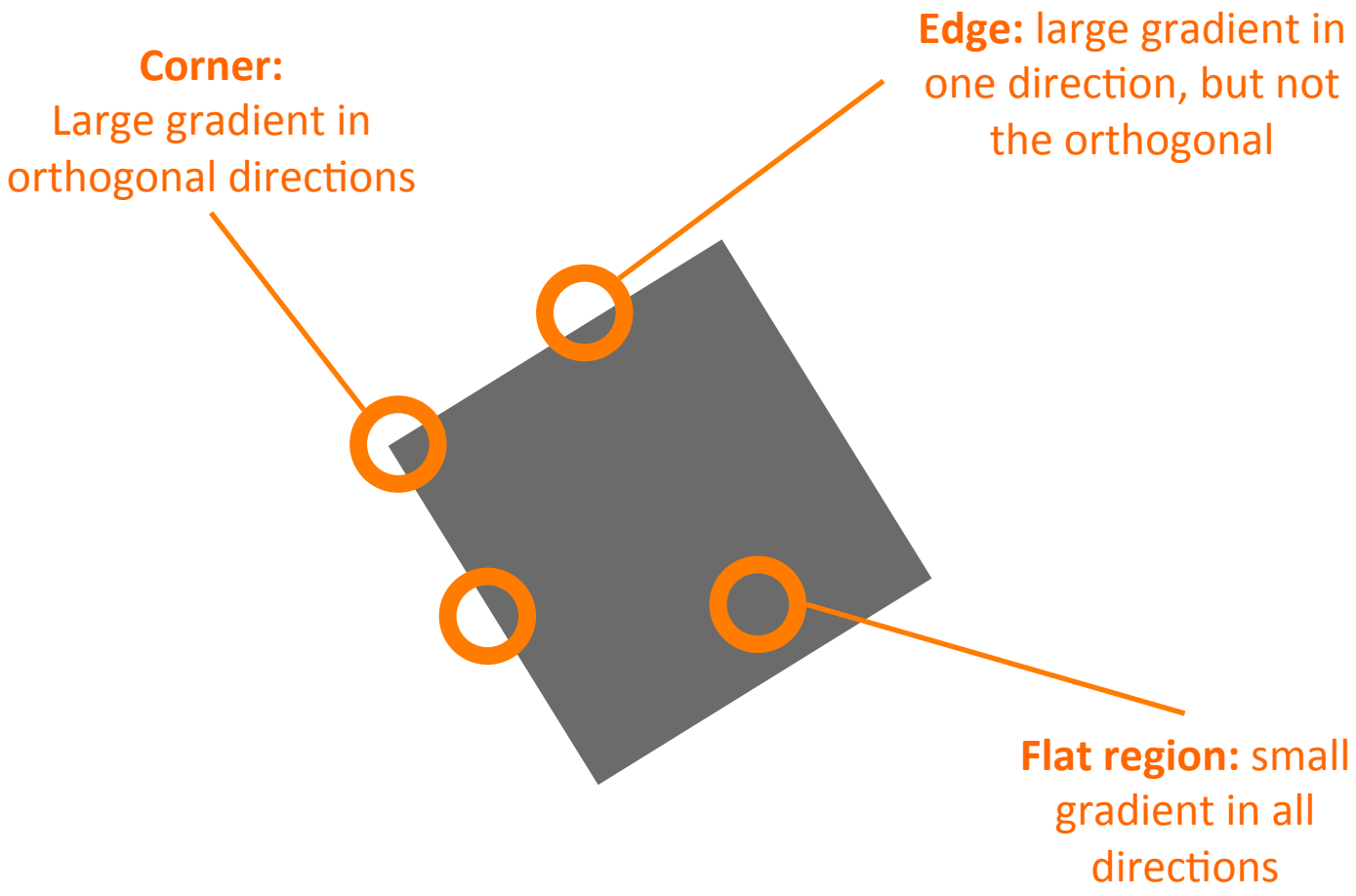
THE HARRIS CORNER DETECTOR



THE HARRIS CORNER DETECTOR

- So, we could detect corners by examining gradient (horizontal and vertical derivatives)
  - » Easily estimated using convolution as we seen earlier
  - » If the horizontal and vertical gradient magnitude are large, declare it a corner point
- But:
  - » There is no reason that the edges would be aligned with the horizontal and vertical directions in the image…

THE HARRIS CORNER DETECTOR



THE HARRIS CORNER DETECTOR

- So we need to look for high gradient magnitude in arbitrary orthogonal (perpendicular) directions
- Image structure tensor:

$$S_{ij} = \sum_{m=i-D}^{i+D} \sum_{n=j-D}^{j+D} w_{mn} \begin{bmatrix} h_{mn}^2 & h_{mn}v_{mn} \\ h_{mn}v_{mn} & v_{mn}^2 \end{bmatrix}$$

- Called a tensor because it's a 2x2 matrix for every (i, j) in the image (MxNx2x2)

THE HARRIS CORNER DETECTOR

- So we need to look for high gradient magnitude in arbitrary orthogonal (perpendicular) directions
- Image structure tensor:

Horizontal gradient at (m, n)

$$S_{ij} = \sum_{m=i-D}^{i+D} \sum_{n=j-D}^{j+D} w_{mn} \begin{bmatrix} h_{mn}^2 & h_{mn}v_{mn} \\ h_{mn}v_{mn} & v_{mn}^2 \end{bmatrix}$$

- Called a tensor because it's a 2x2 matrix for every (i, j) in the image (MxNx2x2)

THE HARRIS CORNER DETECTOR

- So we need to look for high gradient magnitude in arbitrary orthogonal (perpendicular) directions
- Image structure tensor:

Vertical gradient at (m, n)

$$S_{ij} = \sum_{m=i-D}^{i+D} \sum_{n=j-D}^{j+D} w_{mn} \begin{bmatrix} h_{mn}^2 & h_{mn} v_{mn} \\ h_{mn} v_{mn} & v_{mn}^2 \end{bmatrix}$$

- Called a tensor because it's a 2x2 matrix for every (i, j) in the image (MxNx2x2)

THE HARRIS CORNER DETECTOR

- So we need to look for high gradient magnitude in arbitrary orthogonal (perpendicular) directions
- Image structure tensor:

Local gradient covariance matrix

$$S_{ij} = \sum_{m=i-D}^{i+D} \sum_{n=j-D}^{j+D} w_{mn} \begin{bmatrix} h_{mn}^2 & h_{mn}v_{mn} \\ h_{mn}v_{mn} & v_{mn}^2 \end{bmatrix}$$

- Called a tensor because it's a 2x2 matrix for every (i, j) in the image (MxNx2x2)

THE HARRIS CORNER DETECTOR

- So we need to look for high gradient magnitude in arbitrary orthogonal (perpendicular) directions
- Image structure tensor:

$$S_{ij} = \sum_{m=i-D}^{i+D} \sum_{n=j-D}^{j+D} w_{mn} \begin{bmatrix} h_{mn}^2 & h_{mn}v_{mn} \\ h_{mn}v_{mn} & v_{mn}^2 \end{bmatrix}$$

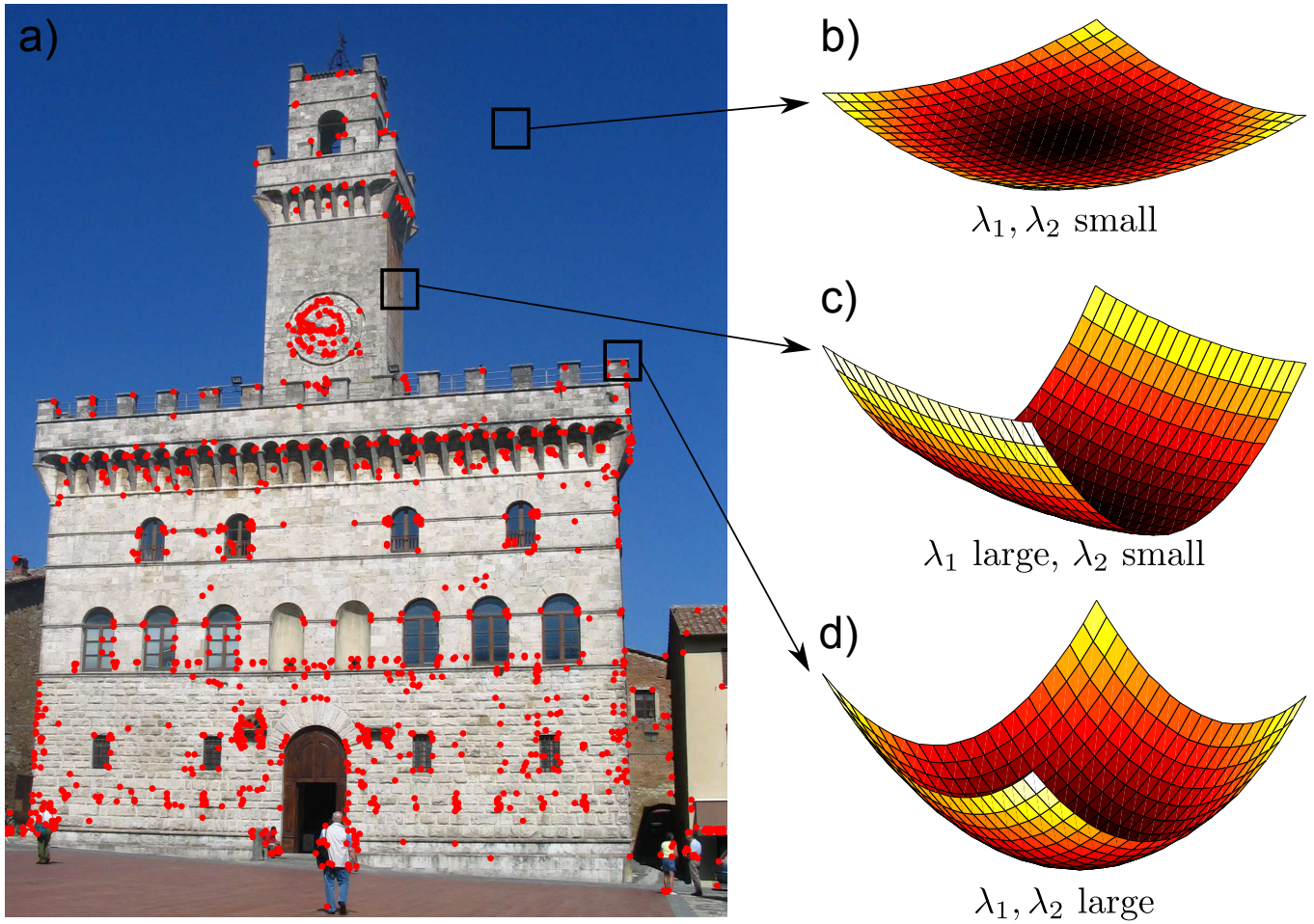
Weighted sum a square window (reduces effect of noise)

- Called a tensor because it's a 2x2 matrix for every (i, j) in the image (MxNx2x2)

THE HARRIS CORNER DETECTOR

- The image structure tensor captures the covariance matrix for the gradient magnitude at each image location
  - » Equal to the outer product of the gradient vector with itself
  - » Averaged over a window to reduce noise
- Covariance matrices are positive semidefinite
  - » Can be **diagonalized** to decompose it into a rotation matrix and a magnitude matrix ($S = QDQ^{-1}$)
  - » D is the diagonal matrix of eigenvalues which equals the magnitude of the gradients in orthogonal directions
    - First eigenvalue λ_1 is magnitude of gradient in direction where it changes most
    - Second eigenvalue λ_2 is magnitude of gradient orthogonal to this
  - » This is exactly what we want ☺

The Harris Corner Detector



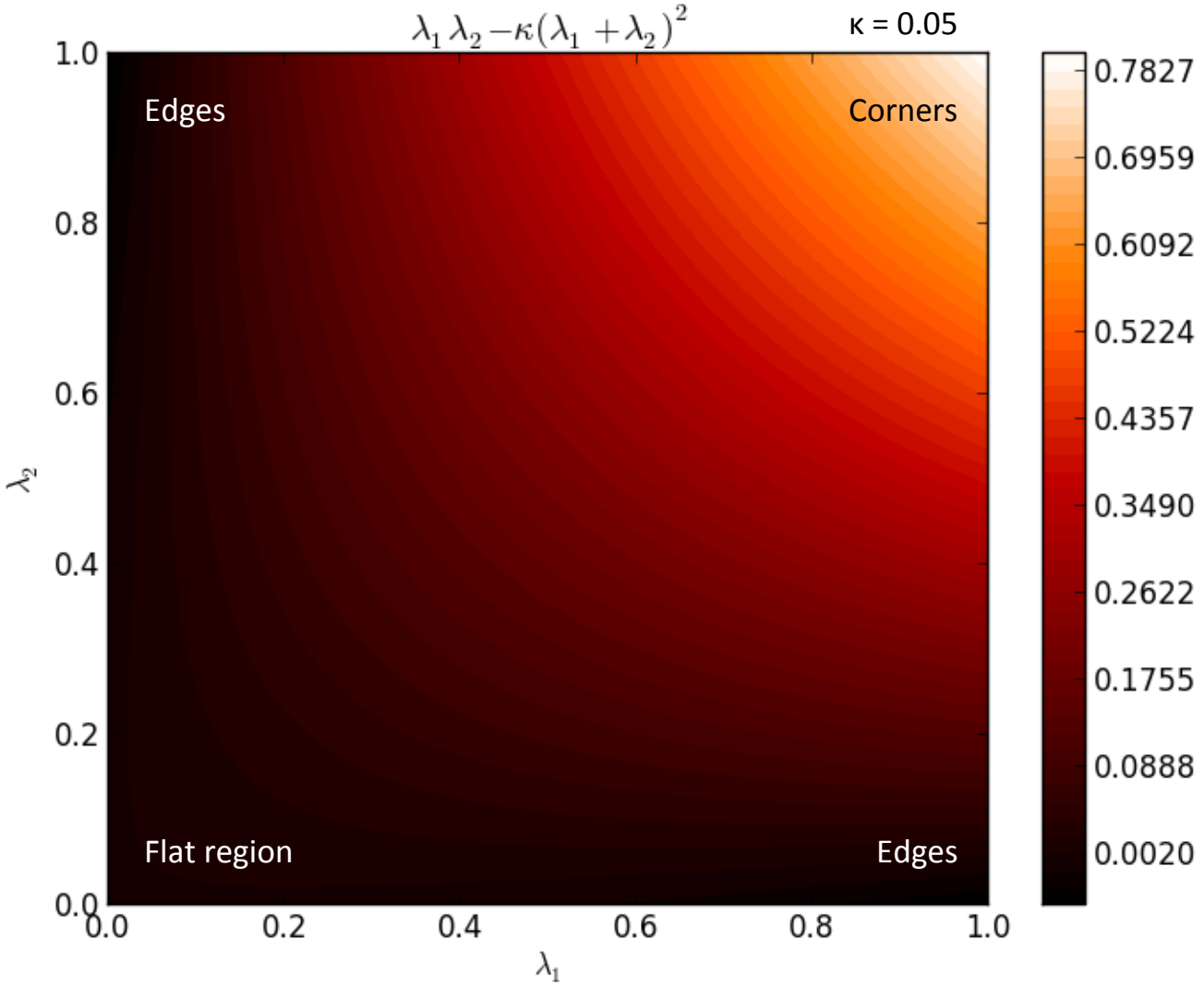
THE HARRIS CORNER DETECTOR

- Harris criteria [Harris & Stephens, 1988]:

$$C_{ij} = \lambda_1 \lambda_2 - \kappa(\lambda_1 + \lambda_2)^2$$

  - » κ is a small positive constant
    - usually between 0.04 and 0.15
  - » Large when both λ_1 and λ_2 are large
  - » Small when one is large but not the other
  - » Small when both are small

THE HARRIS CRITERIA



THE HARRIS CORNER DETECTOR

- Trick:

$$C_{ij} = \lambda_1\lambda_2 - \kappa(\lambda_1 + \lambda_2)^2$$
$$= \mathbf{det}(S_{ij}) - \kappa\,\mathbf{tr}^2(S_{ij})$$

  - » We don't need to do the eigenvalue decomposition (diagonalization) at all!
  - » Since S_{ij} is a 2x2 matrix, we have exact formulas for the trace and the determinant

$$X = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$
$$\mathbf{det}(X) = ad - bc$$
$$\mathbf{tr}(X) = a + d$$

THE HARRIS CORNER DETECTOR

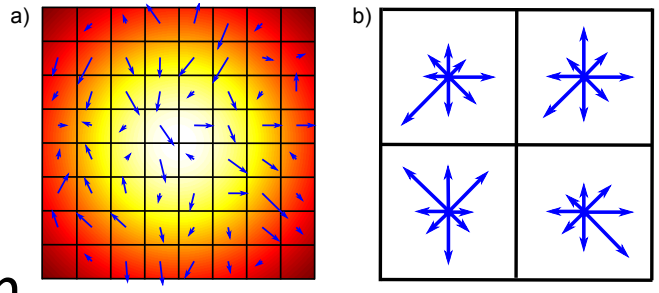
- Algorithm:
  - » Approximate the horizontal h_{ij} and vertical v_{ij} gradients at each position *(i,j)* in the image by convolving the image with appropriate filters.
  - » For each image location *(i,j)*:
    - Compute the image structure tensor S_{ij}
    - Compute the Harris criteria C_{ij}
  - » Perform non-maximal suppression and thresholding of C_{ij}
    - Simple non-maxima suppression: remove all points that are not the largest in a fixed sized window
  - » Non-zero points in *C* are the corner points.

Describing the Local Area

- We can now extract sparse, repeatable, interest points
- How do we describe the local region around these interest points
  - » Compute local descriptors
- Usually very similar to global descriptors but computed over a local window around the interest point

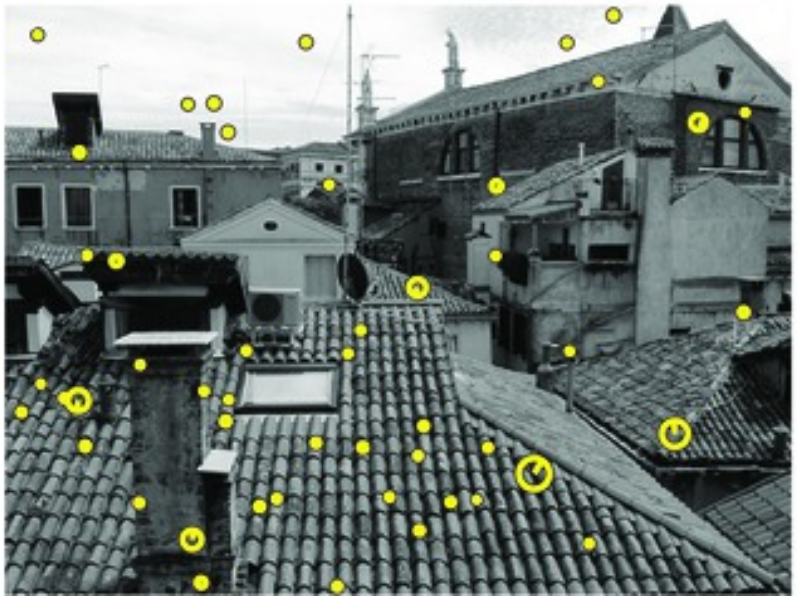
Local Descriptors

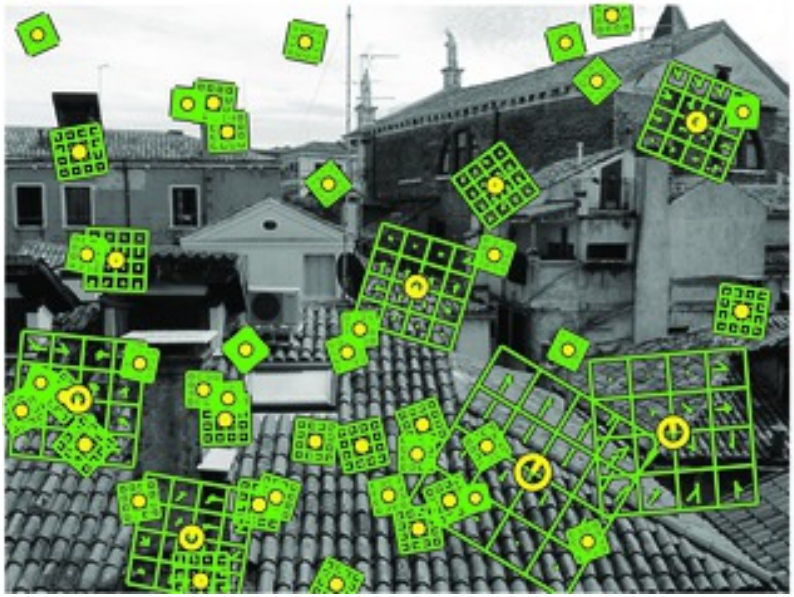
SIFT [Lowe, 1999]



- SIFT: Scale Invariant Feature Transform
- Extract interest points
- Place 16x16 window around each interest point
- Divide into 4x4 cells
- Compute edge orientation histogram for each cell
  - » 8 orientation bins
  - » Weight contributions by gradient magnitude and distance from interest point
- Concatenate all 8 bins x 4 x 4 values to a form a 128D descriptor
- Usually window is also rotated and scaled to match direction of maximum gradient to make rotation invariant

SIFT [Lowe, 1999]



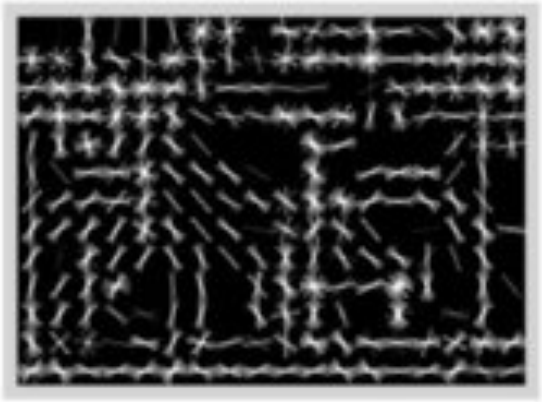


SURF [Bay, et al., 2006]

- SURF: Speeded up robust features
- Similar to SIFT
- Based on responses of Haar like filters on cells
- Optimized for speed
  - » Uses integral images (a.k.a. summed area tables)

HOG [Dalal & Triggs, 2005]

- HOG: Histogram of oriented gradients
- Very similar to SIFT
  - » Block based
  - » Weighted



Multimedia content analysis

AGGREGATING LOCAL FEATURES

MATCHING LOCAL FEATURES

- We have a variable number of local features extracted at interest points in the image
- How do we use these for matching images?
- One approach:
  - » Find best matching pairs of descriptors in 2 images
  - » Estimate an affine transform using subsets (RANSAC)
  - » Prune outliers
  - » Compute a measure of similarity between all remaining matches
- Difficult and computationally expensive!
- Works well for some applications
  - » Standard approach for stitching panoramas

Aggregating Local Features

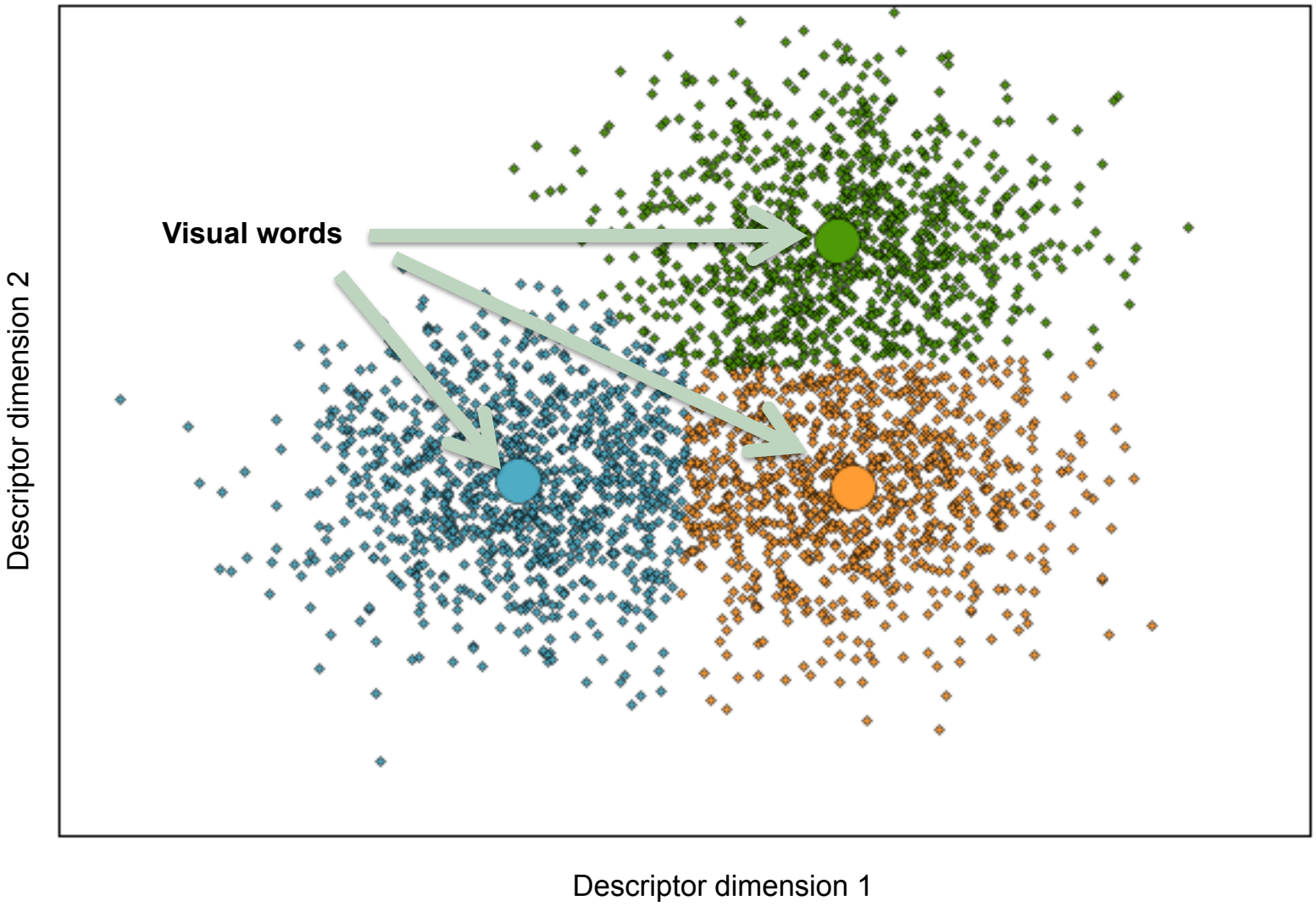
- Matching is greatly simplified if we have **fixed** length descriptors
  - » We can then use standard distance measures like Euclidean
- How do we turn a variable number of interest point descriptors in an image into a fixed length descriptor?
- Several methods
  - » **Bag of Visual Words [Sivic & Zisserman, 2003]**
  - » Fisher kernels [Perronnin & Dance, 2007]
  - » Vector of Locally Aggregated Descriptors (VLAD) [Jegou et al., 2010]

BAG OF VISUAL WORDS

- Idea: local descriptors that are nearby in the feature space represent similar looking regions of the image
- Hypothesis: there is a set of prototype local descriptors that capture interesting low-level visual concepts: **visual words**
- We can build a fixed length codebook of visual words by clustering the extracted descriptors
  - » This codebook is called the **vocabulary**
  - » For example, we could have 500 visual words
- Each image can then be described by the visual words it contains
  - » The **bag of visual words**
  - » This can be converted to a histogram that is the same length as the vocabulary

VOCABULARIES AND CLUSTERING

- What is clustering?
  - » Unsupervised machine learning
  - » Automatically partition data into groups
  - » Usually so that the center of the group is a good representation for the group
- Links with compression:
  - » Vector quantization problem
  - » Find fixed number of representations for values in a high dimensional space


Visual words
Descriptor dimension 2
Descriptor dimension 1

K-Means

- Simple but very popular clustering algorithm
- Works on vectors of fixed but arbitrary dimension
- Need to choose K in advance
  - » K is the number of clusters
- Based on Euclidean distance
  - » Can be adapted for other measures (k-medoids)
- Convergence to optimal clusters is **not** guaranteed
  - » Can get stuck in a local maximum

K-Means Algorithm

Start with K random chosen cluster centers

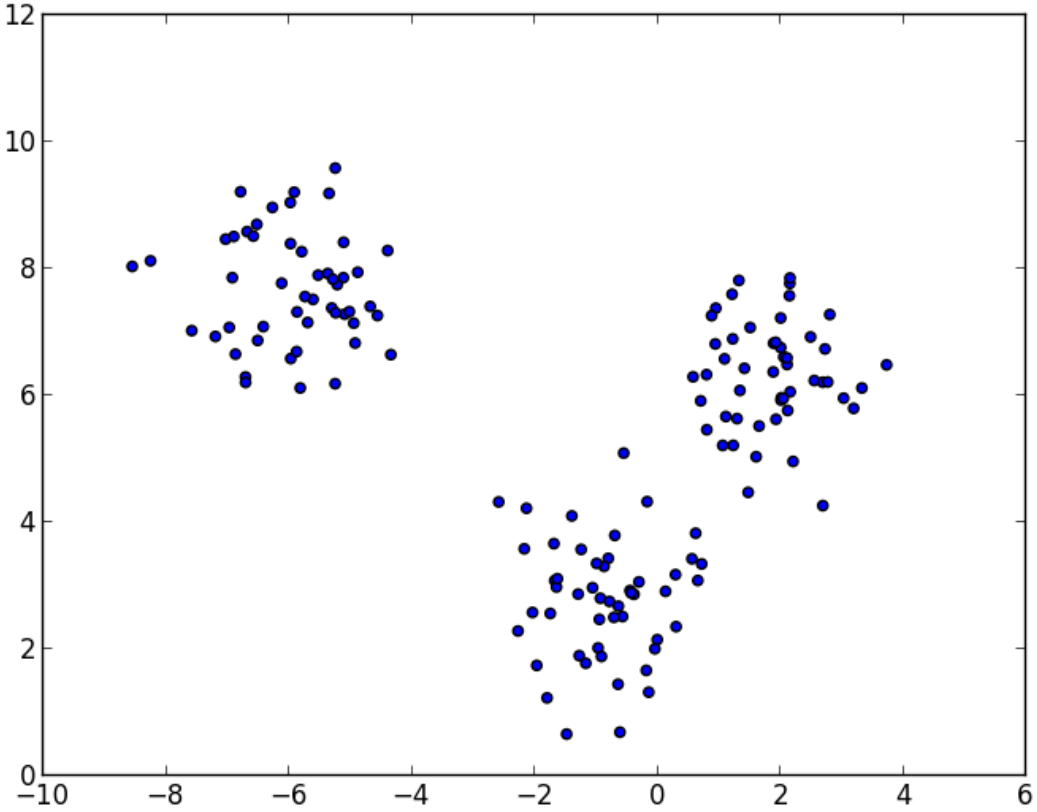
While not converged:

1. Assign each point (descriptor) to nearest cluster center
2. Move cluster center to centroid of all points assigned to it

- Convergence is when change in position of cluster centers is not significant
- An expectation maximization (EM) algorithm

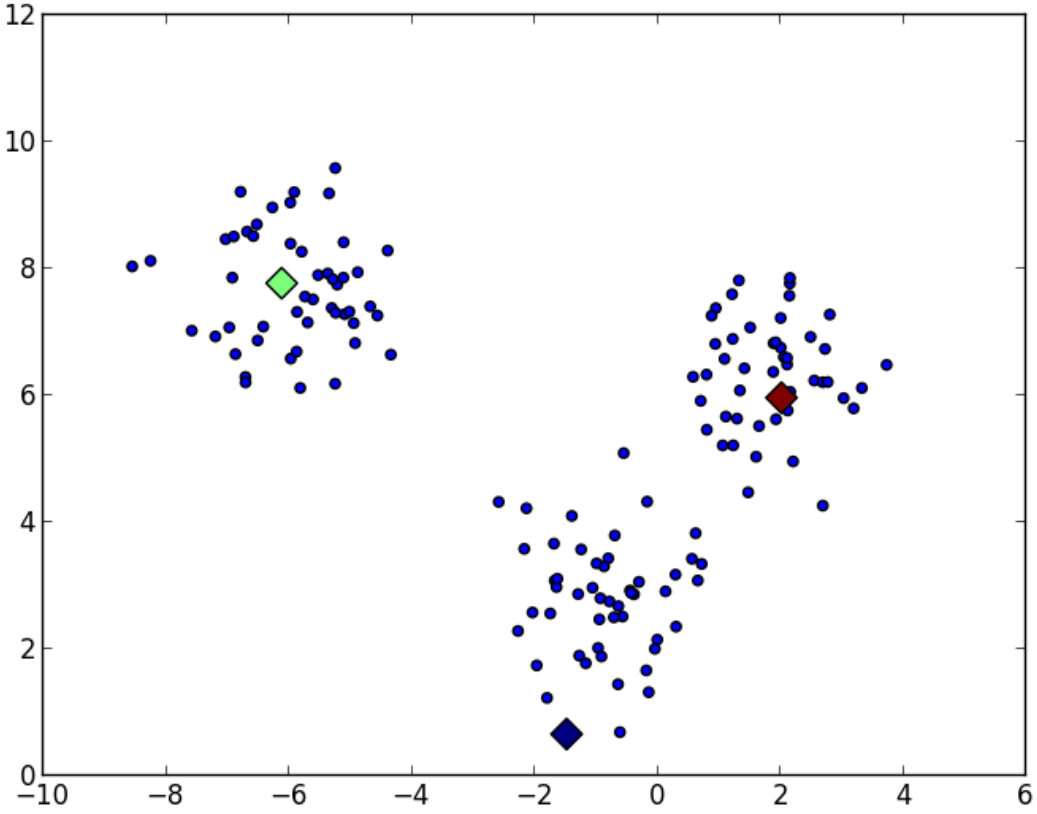
K-Means Example

Start with unlabeled data points in N-dimensions



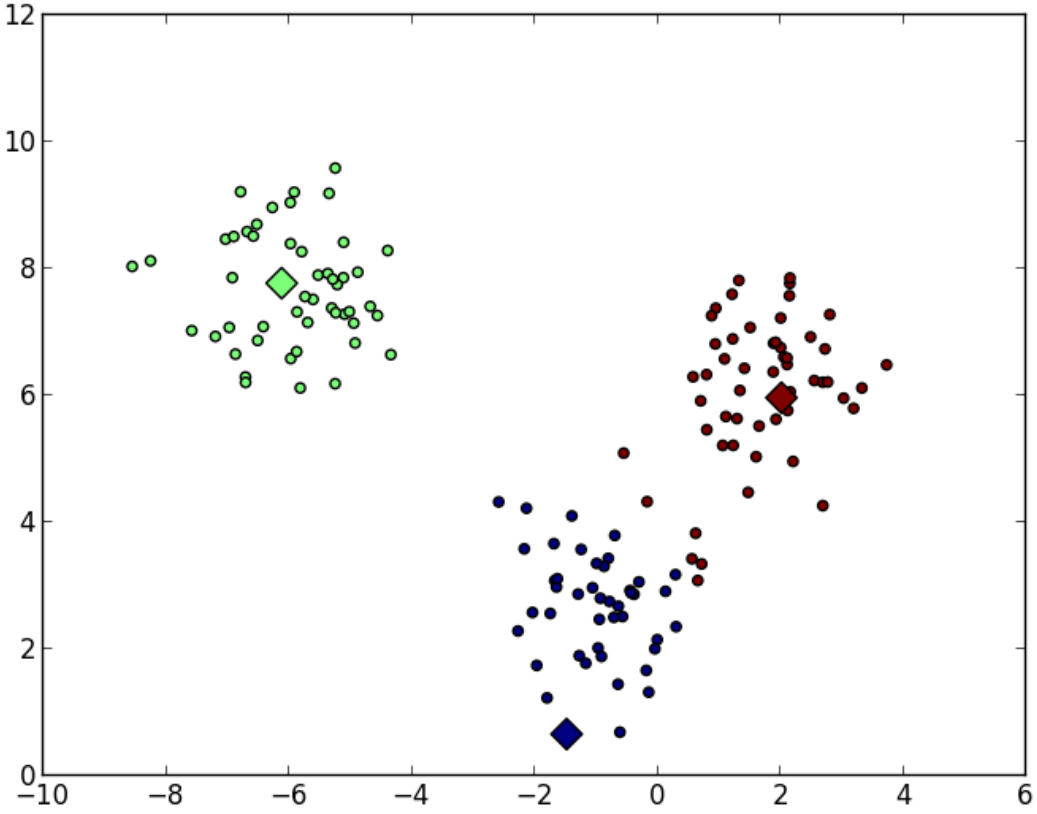
K-Means Example

Initialize clusters centers randomly



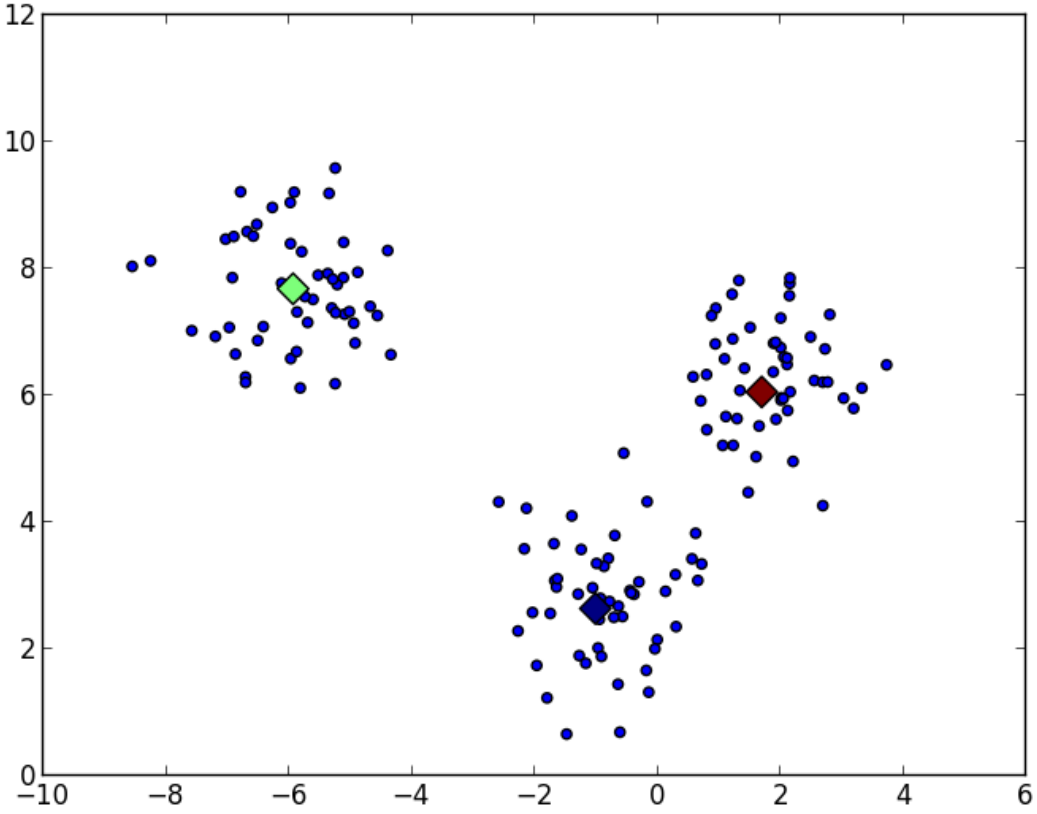
K-Means Example

Assign points to nearest cluster center



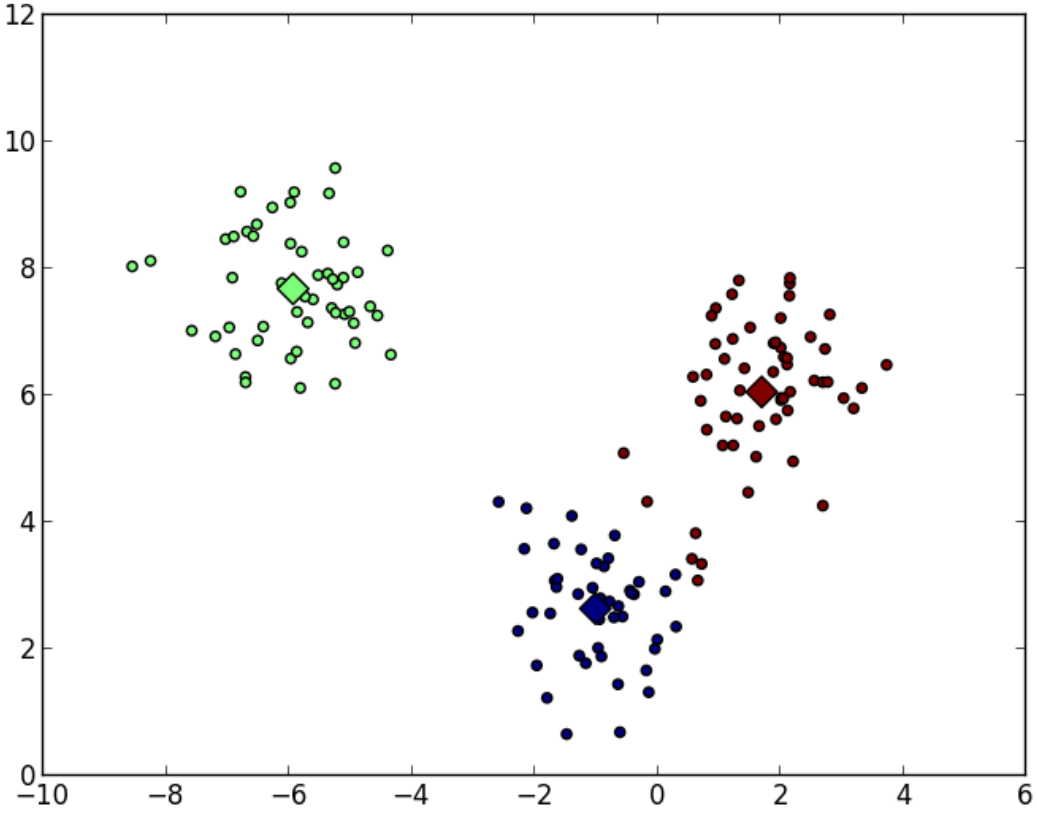
K-Means Example

Move cluster centers to centroid of assigned points



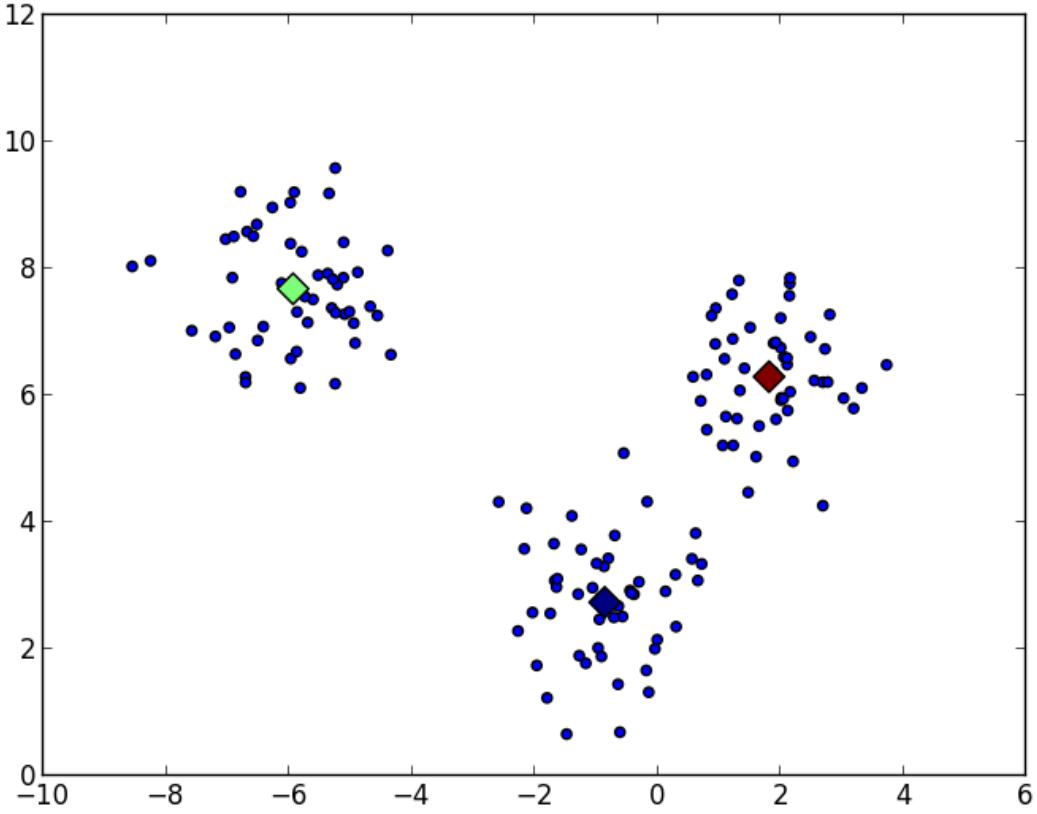
K-Means Example

Assign points to nearest cluster center

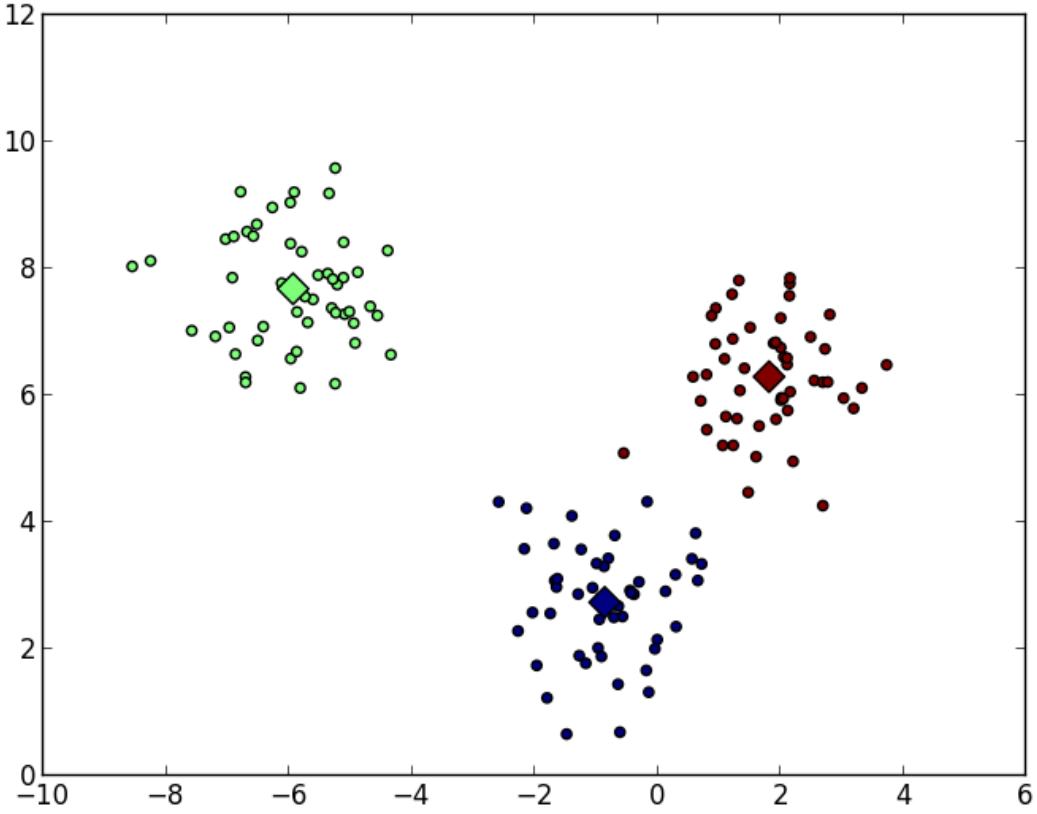


K-Means Example

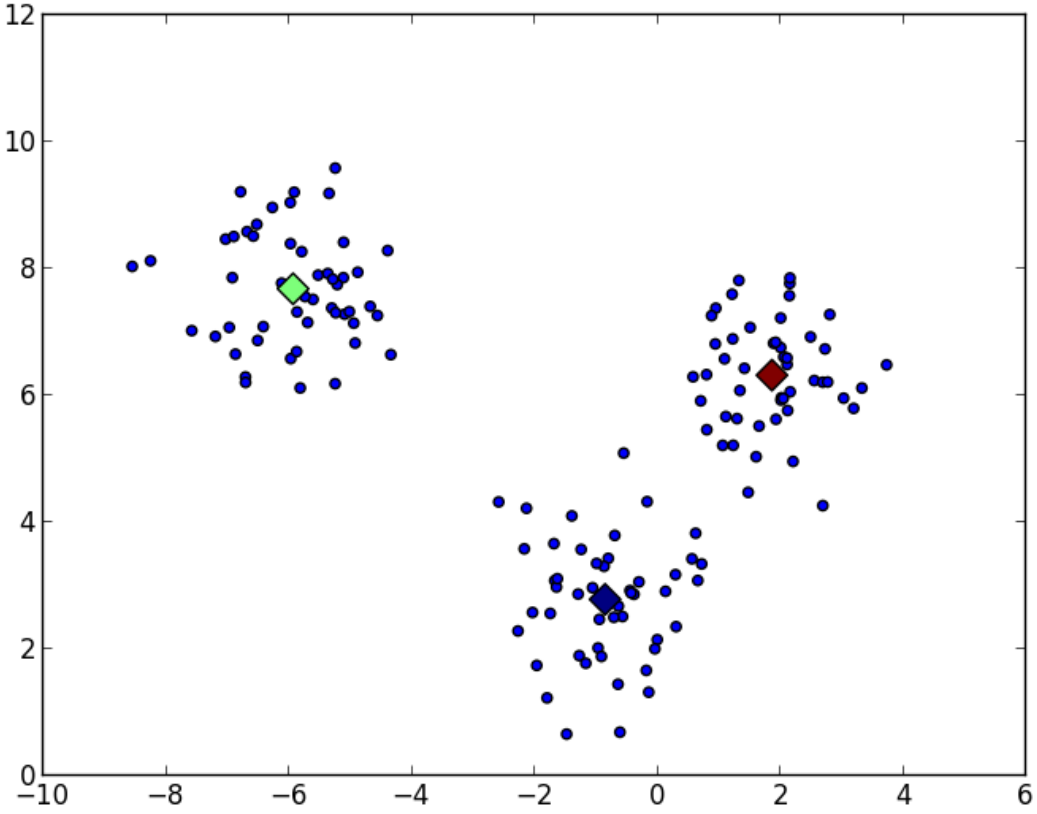
Move cluster centers to centroid of assigned points



K-Means Example

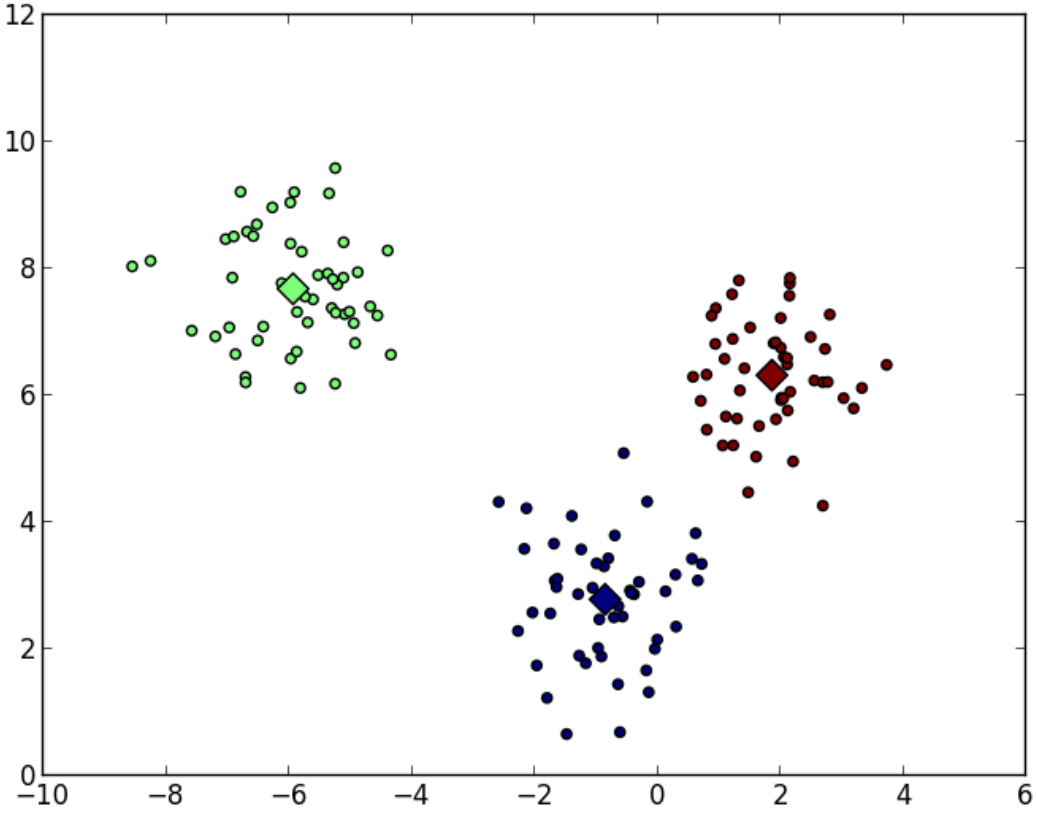


K-Means Example



K-Means Example

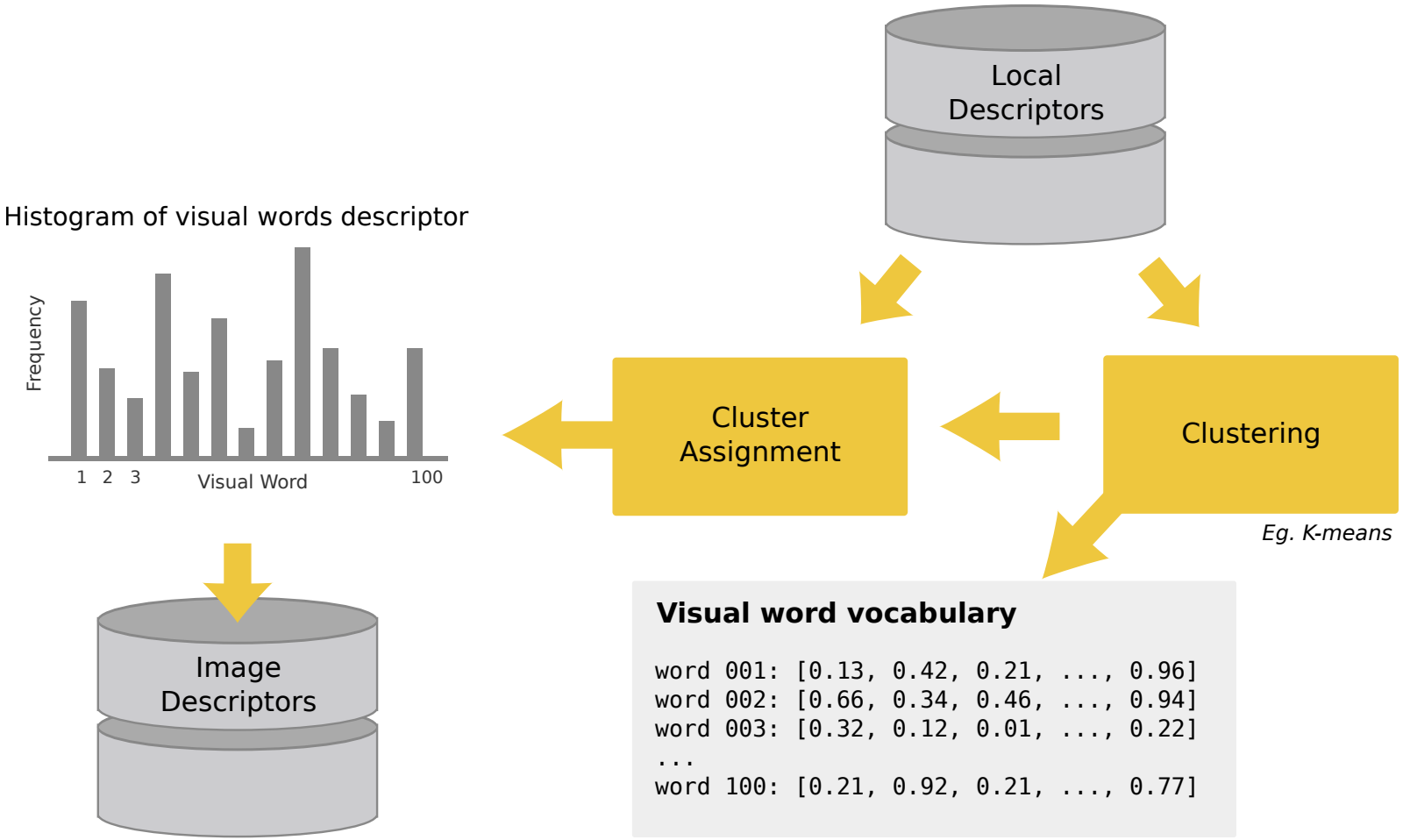
Algorithm converged when centers do not move significantly



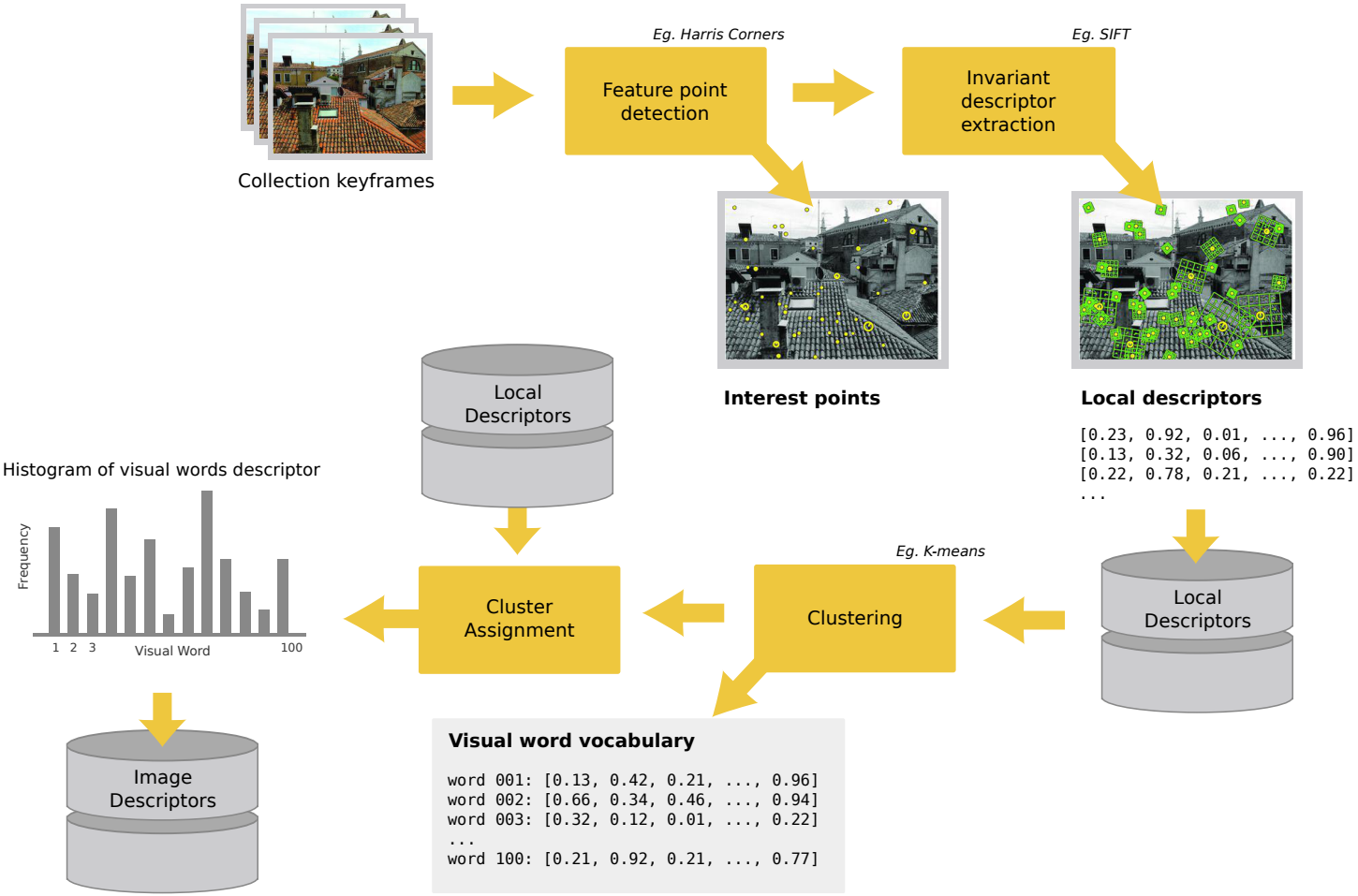
BAG OF VISUAL WORDS ALGORITHM

- Collect all feature points from all images in the collection
- Perform clustering (K-Means) on these to find a predefined number of clusters K
- For each image I:
  - » Initialize a histogram H representation of length K to zero
  - » For each interest point descriptor D in I:
    - Find the index i of the nearest cluster found by K-means
    - Increment H[i]
  - » H is the descriptor for I

Bag of Visual Words Algorithm



Feature Extraction Pipeline



Multimedia content analysis

MATCHING AND DISTANCE MEASURES

Matching and Distance Measures

- To match images based on descriptors, we need a method for computing the distance or similarity between the descriptors
- We've already seen one way of doing this with Euclidean distances
- There are several other useful distance measures
- Vary in level of robustness, discriminability, and efficiency

Simple Distance Measures

- P-Norm based distances

$$\|P - Q\|_p$$

- Give rise to L-n (Minkowski form) distances

$$D_{Ln}(P, Q) = \left[\sum_{i=1}^{N} (P_i - Q_i)^n \right]^{\frac{1}{n}}$$

Simple Distance Measures

- L-1 distance

$$D_{L1}(P,Q) = \sum_{i=1}^{N} |P_i - Q_i|$$

- L-2 distance

$$D_{L2}(P,Q) = \sqrt{\sum_{i=1}^{N} (P_i - Q_i)^2}$$

Simple Distance Measures

- Advantages:
  - » Easy to implement
  - » Can be computed very fast
- Disadvantages:
  - » Sensitive to bin quantization
  - » Sensitive to unequal bin masses
  - » Sensitive to un-normalized histograms

Dealing with Unequal Bin Masses

- χ^2-squared distance

$$D_{\chi^2}(P,Q) = \frac{1}{2}\sum_{i=1}^{N}\frac{(P_i - Q_i)^2}{P_i + Q_i}$$

- Advantages:
  - » cheap to compute,
  - » simple to implement,
  - » reduces the effect of large bins and unequal histogram masses.
  - » Experimentally better than L-1 and L-2 in many applications.
- Disadvantages:
  - » sensitive to bin quantization.

DEALING WITH QUANTIZATION ERROR

- Similar values may fall in different histogram bins due to quantization error
- Simple measures do not account for this
  - » Only compare each bin against the corresponding bin (P_i – Q_i)
- Cross-bin distance measures:
  - » Also compare P_i and Q_j, when $i \neq j$
  - » Quadratic form measures
  - » Earth movers distance

QUADRATIC FORM DISTANCES

$$D_{QF}(P,Q|A) = \sqrt{\sum_{i,j=1}^{N}(P_i - Q_i)(P_j - Q_j)A_{ij}}$$

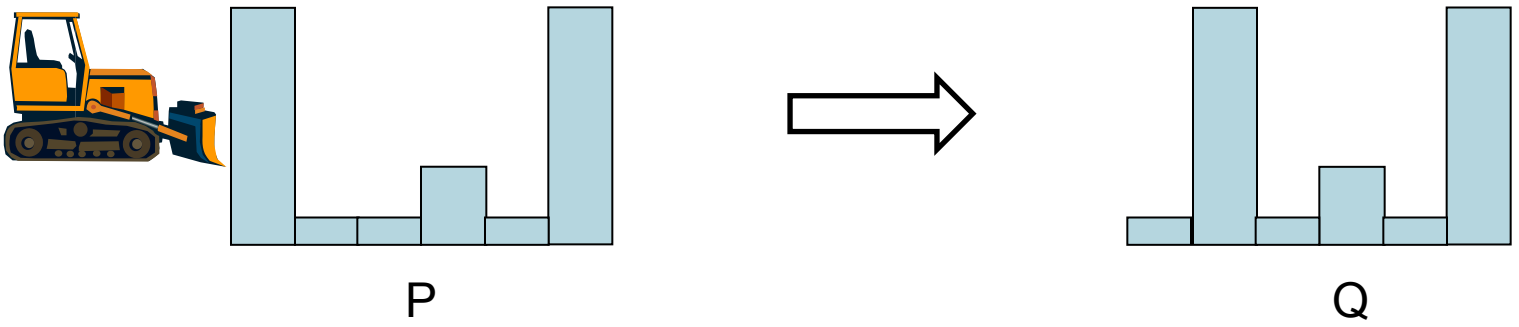
- Define a symmetric (usually pos. def.) matrix A where A_{ij} = similarity between bin *i* and *j*
- When A = identity matrix, $D_{QF} = D_{L2}$
- A is often set to be the inverse of the covariance matrix for the data
  - » Covariance matrix measures how the data varies together
  - » Called the **Mahalanobis** distance when A is the covariance

QUADRATIC FORM DISTANCES

- Advantages:
  - » can be made less sensitive to bin quantization by choosing appropriate matrix A
- Disadvantages:
  - » computationally more expensive,
  - » requires estimation of A

EARTH MOVERS DISTANCE (EMD)

- The minimal cost to transform one histogram into another, given a ground distance between features in the histograms



EARTH MOVERS DISTANCE (EMD)

$$D_{\text{emd}}(P, Q|D) = \min_F \frac{\sum_{i,j} F_{ij} D_{ij}}{\sum_i F_{ij}}$$

$$s.t. \quad \sum_j F_{ij} \leq P_i$$

$$\sum_i F_{ij} \leq Q_j$$

$$\sum_{i,j} F_{ij} = \min\{\sum_i P_i, \sum_j Q_j\}$$

$$F_{ij} \geq 0$$

EARTH MOVERS DISTANCE (EMD)

$$D_{\text{emd}}(P, Q|D) = \min_F \frac{\sum_{i,j} F_{ij} D_{ij}}{\sum_i F_{ij}}$$

$$s.t. \quad \sum_j F_{ij} \leq P_i$$

EMD Conditions

$$\sum_i F_{ij} \leq Q_j$$

$$\sum_{i,j} F_{ij} = \min\{\sum_i P_i, \sum_j Q_j\}$$

$$F_{ij} \geq 0$$

EARTH MOVERS DISTANCE (EMD)

$$D_{\text{emd}}(P, Q|D) = \min_F \frac{\sum_{i,j} F_{ij} D_{ij}}{\sum_i F_{ij}}$$

Ground distance between bin i and j

$$s.t. \quad \sum_j F_{ij} \leq P_i$$

$$\sum_i F_{ij} \leq Q_j$$

$$\sum_{i,j} F_{ij} = \min\{\sum_i P_i, \sum_j Q_j\}$$

$$F_{ij} \geq 0$$

EARTH MOVERS DISTANCE (EMD)

$$D_{\text{emd}}(P, Q|D) = \min_F \frac{\sum_{i,j} F_{ij} D_{ij}}{\sum_i F_{ij}}$$

Flow between bin i and j

$$s.t. \quad \sum_j F_{ij} \leq P_i$$

$$\sum_i F_{ij} \leq Q_j$$

$$\sum_{i,j} F_{ij} = \min\{\sum_i P_i, \sum_j Q_j\}$$

$$F_{ij} \geq 0$$

EARTH MOVERS DISTANCE (EMD)

$$D_{\text{emd}}(P, Q|D) = \min_F \frac{\sum_{i,j} F_{ij} D_{ij}}{\sum_i F_{ij}}$$

$$s.t. \quad \sum_j F_{ij} \leq P_i$$

The total flow out of bin i must not exceed the amount in bin i

$$\sum_i F_{ij} \leq Q_j$$

$$\sum_{i,j} F_{ij} = \min\{\sum_i P_i, \sum_j Q_j\}$$

$$F_{ij} \geq 0$$

EARTH MOVERS DISTANCE (EMD)

$$D_{\text{emd}}(P, Q|D) = \min_F \frac{\sum_{i,j} F_{ij} D_{ij}}{\sum_i F_{ij}}$$

$$s.t. \quad \sum_j F_{ij} \leq P_i$$

$$\sum_i F_{ij} \leq Q_j$$

The total flow must not exceed the mass of the smaller histogram

$$\sum_{i,j} F_{ij} = \min\{\sum_i P_i, \sum_j Q_j\}$$

$$F_{ij} \geq 0$$

EARTH MOVERS DISTANCE (EMD)

$$D_{\text{emd}}(P, Q|D) = \min_F \frac{\sum_{i,j} F_{ij} D_{ij}}{\sum_i F_{ij}}$$

$$s.t. \quad \sum_j F_{ij} \leq P_i$$

$$\sum_i F_{ij} \leq Q_j$$

$$\sum_{i,j} F_{ij} = \min\{\sum_i P_i, \sum_j Q_j\}$$

$$F_{ij} \geq 0$$

Negative flows are not allowed

EARTH MOVERS DISTANCE (EMD)

- Advantages:
  - » Robust to bin quantization
  - » Can be used on signatures (variable length descriptors)
- Disadvantages:
  - » Computation requires solving a linear programming optimization problem
    - Equivalent to the **transportation problem**
  - » Usually computationally expensive
  - » Difficult to implement

TRADEOFFS

- Simple measures
  - » Fast to compute
  - » Easy to implement
  - » Sensitive to quantization error
- Cross-bin
  - » Slower to compute
  - » More difficult to implement (esp. EMD)
  - » Robust to quantization error
  - » Require ground distance or similarity matrix

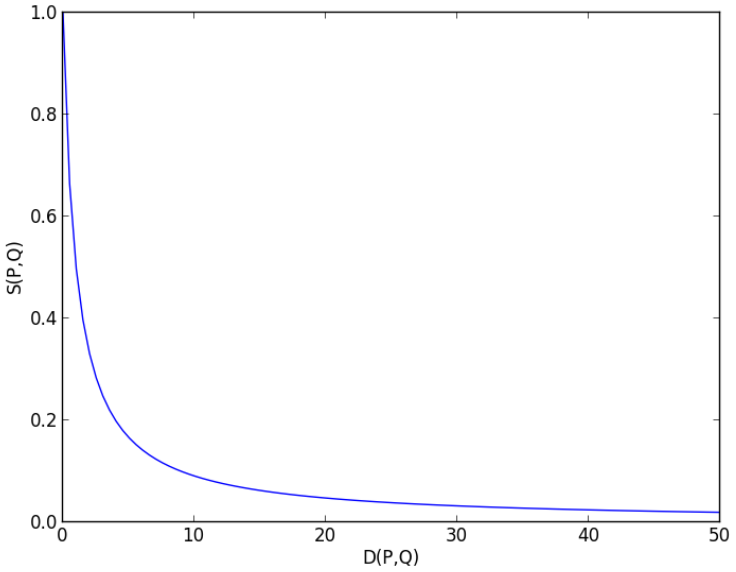
AFFINITY FUNCTIONS

- Some applications require similarity function (affinity function) rather than distance function
  - » Example: Affinity matrix for Quadratic form distances
  - » Affinity high for similar descriptors
  - » Affinity low for different descriptors

- Distance functions can be converted to affinity functions

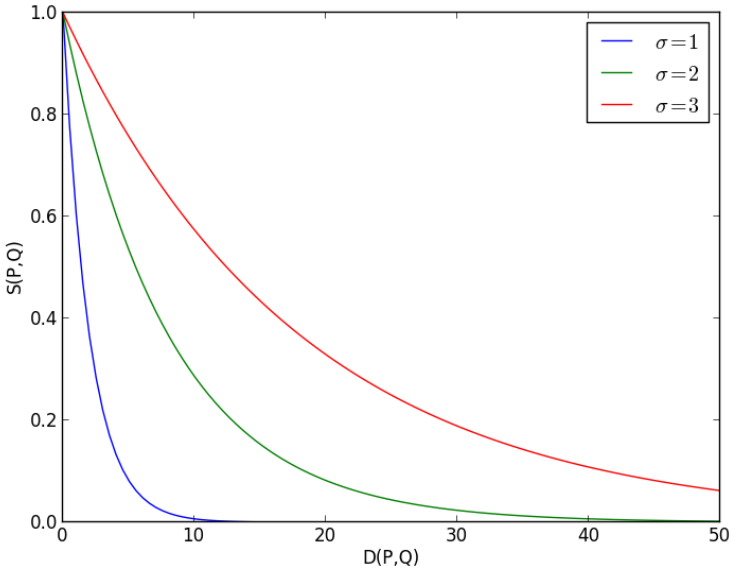
Distance to Affinity

$$S(P,Q) = \frac{1}{1 + D(P,Q)}$$



Distance to Affinity

$$S(P,Q) = \exp\left(-\frac{D(P,Q)}{\sigma^2}\right)$$



Multimedia content analysis

LEARNING AND CLASSIFICATION

Learning and Classification

- How to build a classifiers that can, for example:
  - » Tell faces from non-faces?
  - » Tell persons from cars?
  - » Tell if this video contains cats?
  - » Tell if a part is OK or defective?
  - » Tell if a scene is indoors or outdoors?
  - » Tell if there is an intruder present?
- It is **very** difficult to write classifiers like this by hand
  - » What features make a cat a cat?
- Usually we gather examples and use **machine learning**

Machine Learning

Two main types of machine learning

- **Unsupervised**
  - » Start with unlabeled data
  - » Discover structure from data (e.g. clusters)
  - » K-means is an example
- **Supervised**
  - » Start with labeled/annotated data
    - Examples of input-output pairs
  - » Infer a function from the data
  - » Function range is continuous: **regression**
  - » Function range is discrete: **classification**

SUPERVISED CLASSIFICATION

Problem: Given a **training set** of K examples $x_{1...K}$ and corresponding labels $y_{1...K}$ infer a function $f : R^D \rightarrow C$

- » $x_{1...K}$ are D-dimensional descriptors
- » $y_{1...K}$ are **class labels**
- » **f(x)** is a function that maps a descriptor to a class label

Goal:

- » Use training data to automatically create f
- » Minimize **generalization error**
  - Error on unseen **test** data

Binary classification: special case when y_i is 0 or 1

SUPERVISED CLASSIFICATION

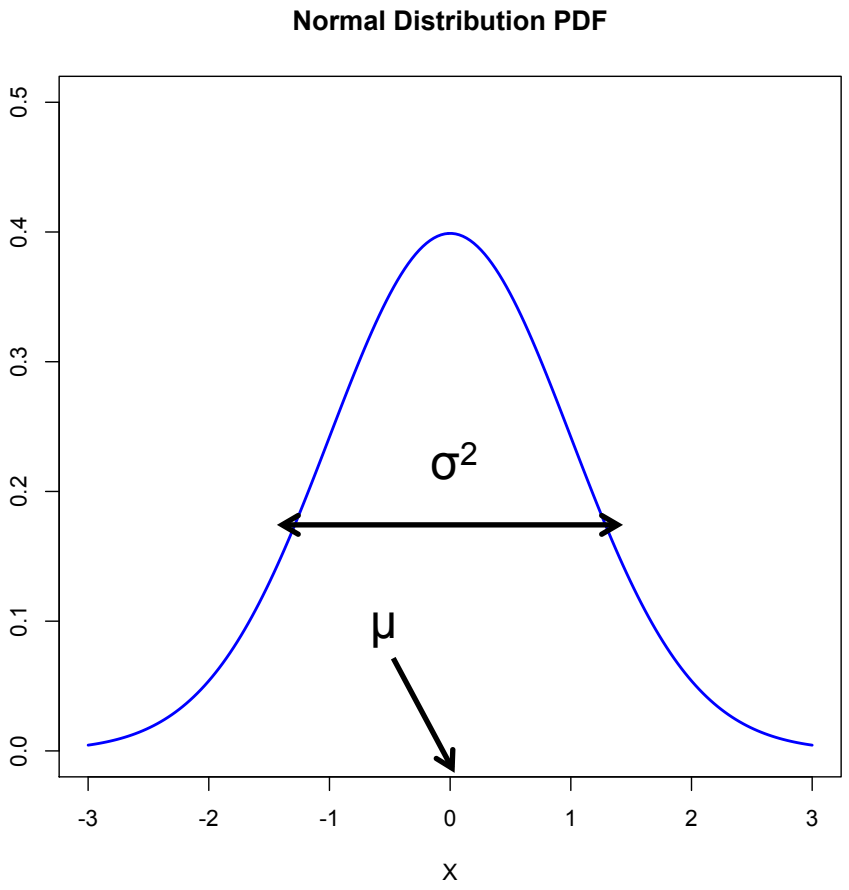
- Problem is **ill-posed**
  - » It is necessary to make some assumptions about the nature of the data to do inference
- Different assumptions lead to different machine learning algorithms
- Typical kinds of assumptions:
  - » Training examples are drawn from particular parametric distribution
  - » Decision boundary between classes is a hyperplane

QUADRATIC DISCRIMINANT ANALYSIS

- a.k.a. Gaussian Discriminant Analysis
- Simple ML algorithm to understand and implement
- Assumption:
  - » Training examples $x_{1...K}$ drawn from two separate multivariate Gaussian (Normal) distributions
- Algorithm overview:
  - » Training: Estimate parameters of Gaussians from training examples
  - » Inference: Use fitted distributions to classify unseen examples

The Gaussian Distribution

- a.k.a. Normal distribution
- Distribution over continuous random variables
- Uni-modal
  - » Single peak
- Occurs a lot in nature
  - » Often due to the central limit theorem
- Two parameters
  - » Mean (μ)
  - » Variance (σ^2)



The Gaussian Distribution

$$P(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

- As x gets farther from μ, probability decreases exponentially
- Decrease is faster with smaller σ^2

The Gaussian Distribution

$$P(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Squared distance from the mean

- As x gets farther from μ, probability decreases exponentially
- Decrease is faster with smaller σ^2

The Gaussian Distribution

$$P(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Variance

- As x gets farther from μ, probability decreases exponentially
- Decrease is faster with smaller σ^2

The Gaussian Distribution

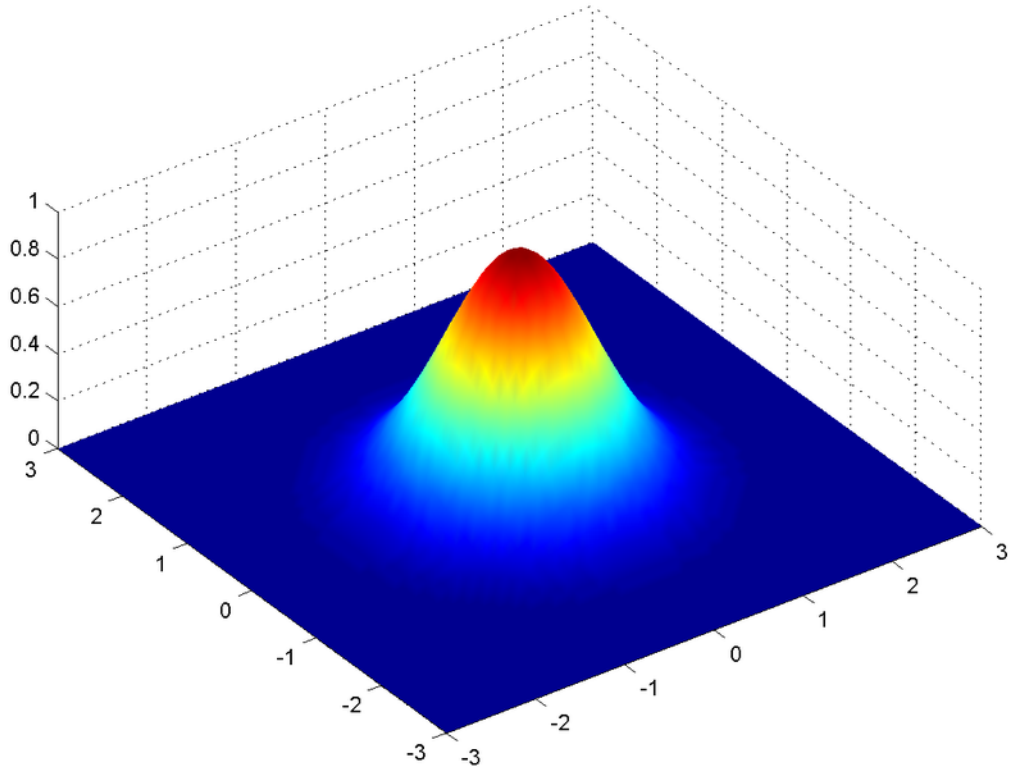
$$P(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Normalizing constant

- As x gets farther from μ, probability decreases exponentially
- Decrease is faster with smaller σ^2

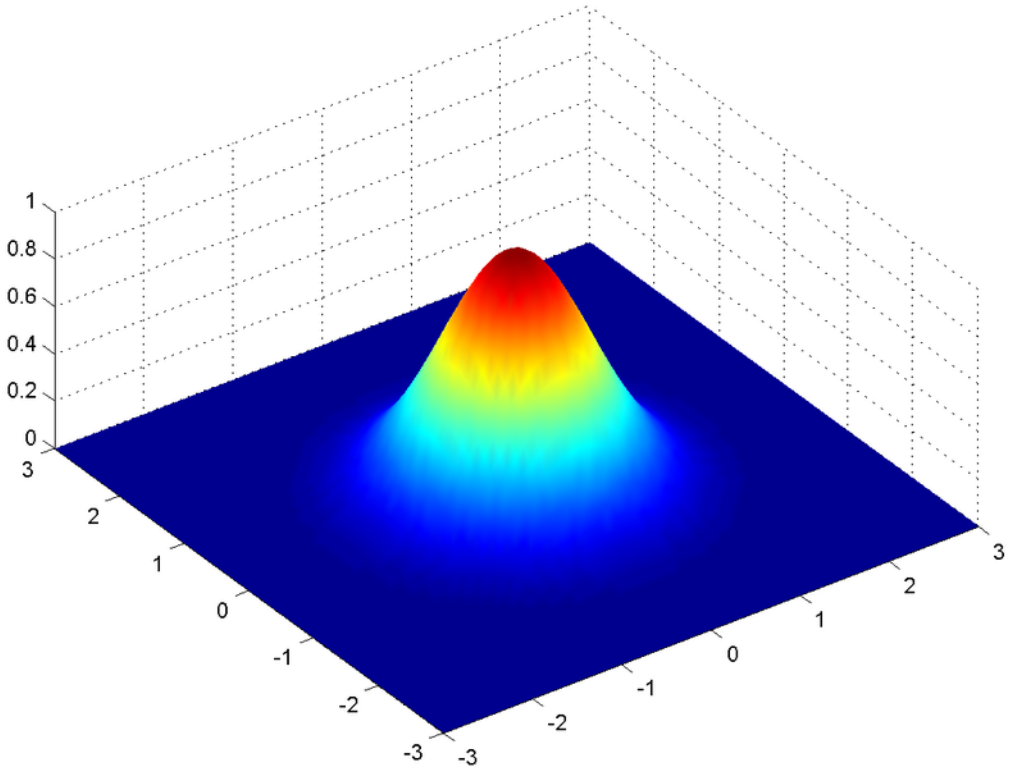
MULTIVARIATE GAUSSIAN DISTRIBUTION

- Extension of the Gaussian to multiple dimensions
- Can be used to describe the probability of a D-dimensional feature vector



Multivariate Gaussian Distribution

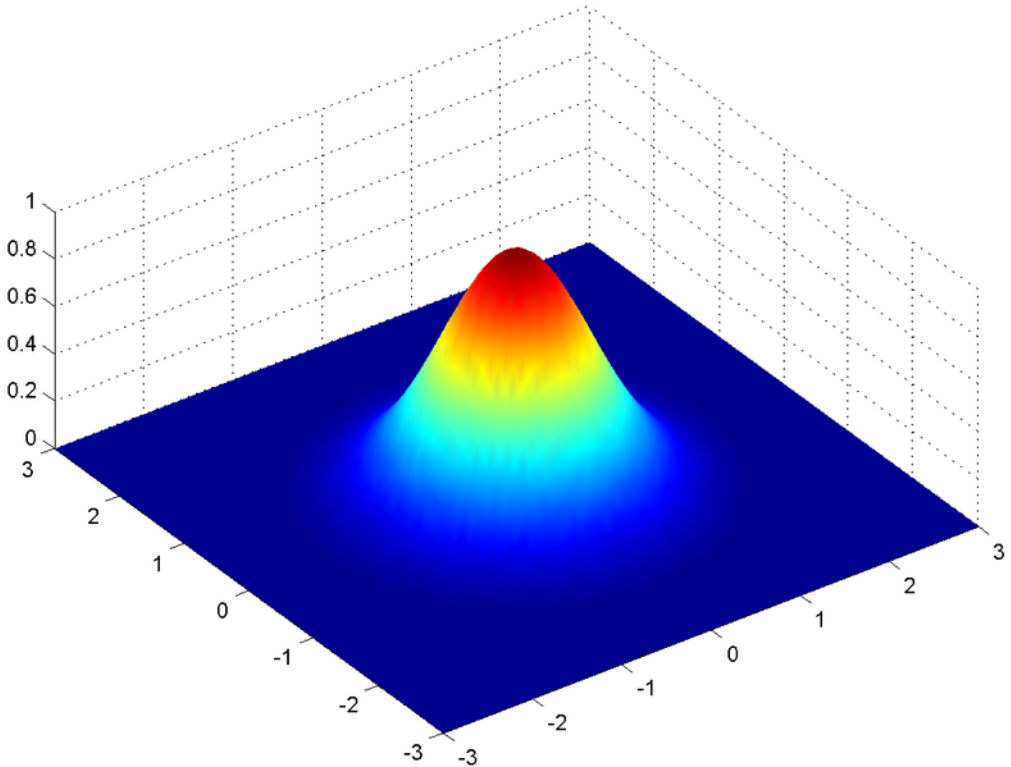
$$P(\mathbf{x}|\mu,\Sigma) = \frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma|^{\frac{1}{2}}}\exp(-0.5(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu))$$



Multivariate Gaussian Distribution

$$P(\mathbf{x}|\mu,\Sigma) = \frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma|^{\frac{1}{2}}}\exp(-0.5(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu))$$

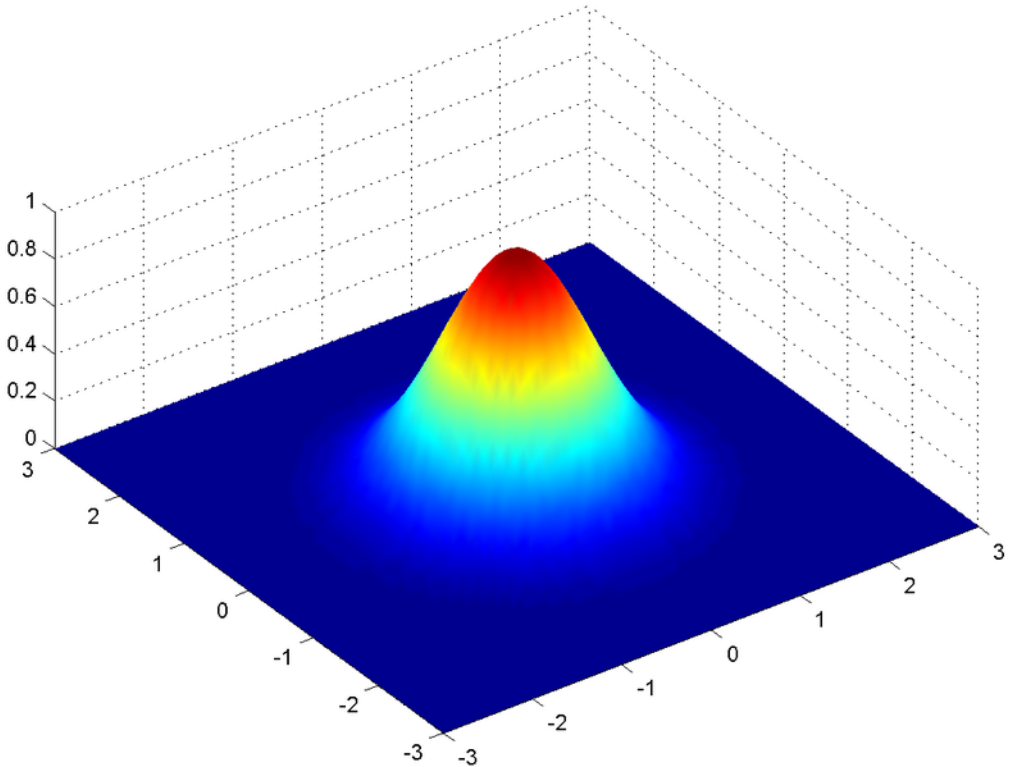
Covariance matrix



Multivariate Gaussian Distribution

$$P(\mathbf{x}|\mu,\Sigma) = \frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma|^{\frac{1}{2}}}\exp(-0.5(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu))$$

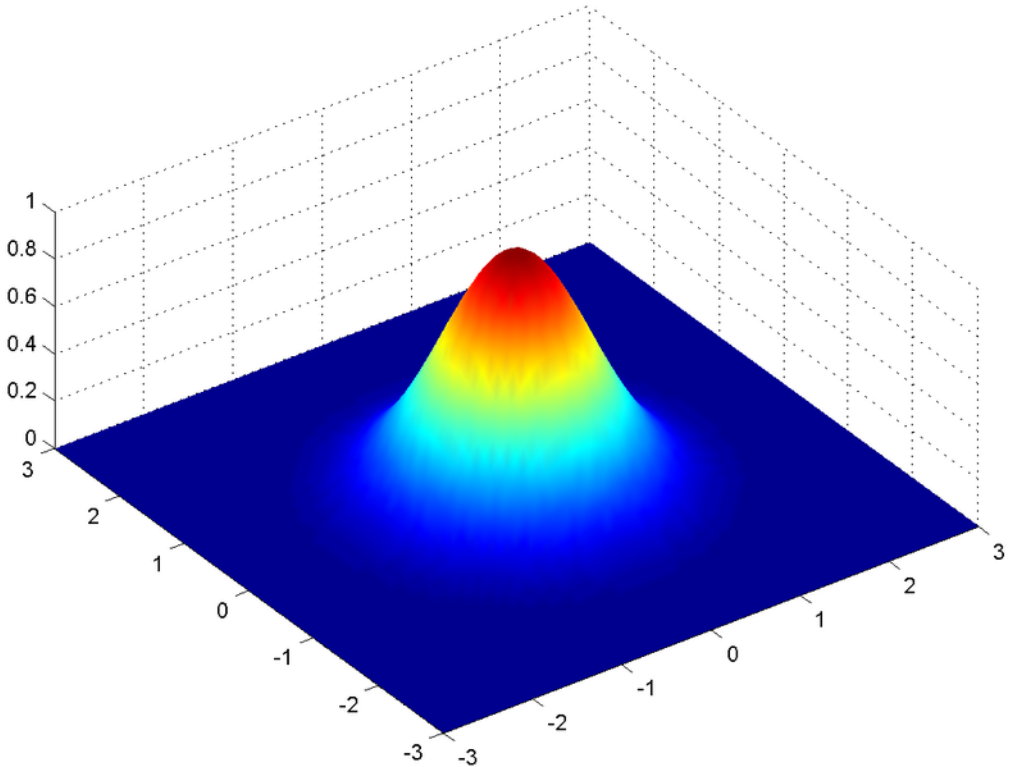
Normalizing constant



Multivariate Gaussian Distribution

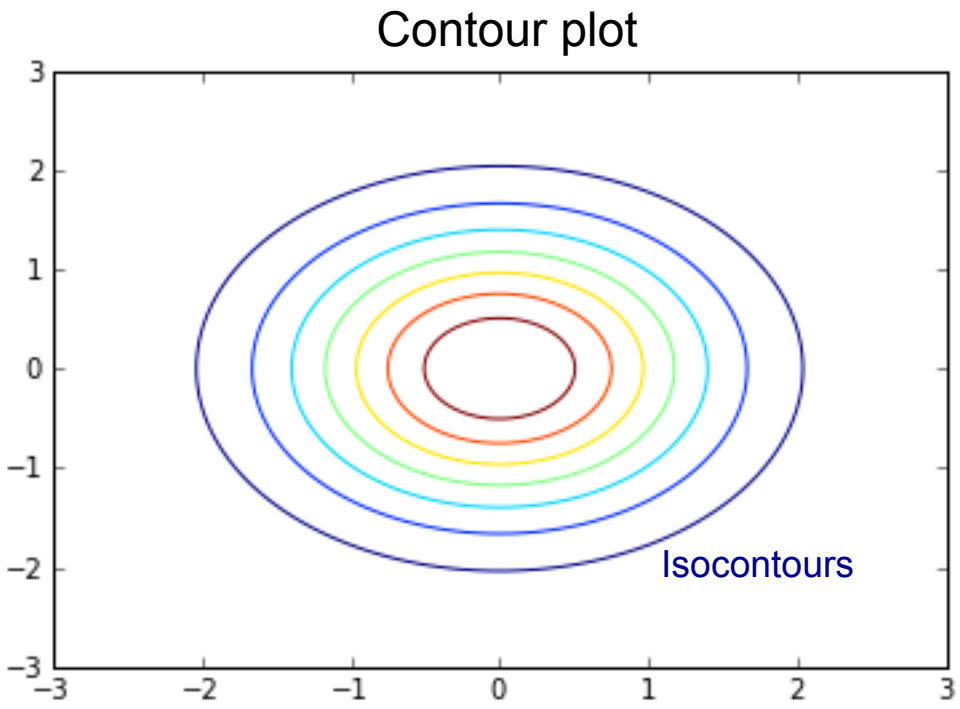
$$P(\mathbf{x}|\mu,\Sigma) = \frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma|^{\frac{1}{2}}}\exp(-0.5(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu))$$

Mahalanobis distance to mean



Visualization with Contour Plots

$$P(\mathbf{x}|\mu,\Sigma) = \frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma|^{\frac{1}{2}}}\exp(-0.5(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu))$$



Maximum Likelihood Estimation

- How to estimate the parameters of the Gaussian from data?
- Several reasonable ways
- Maximum likelihood principle (ML)
  - » Choose parameters that make our observations most likely

ML FOR THE MEAN OF THE GAUSSIAN

$$\mu_{ML} = \arg\max_{\mu} P(\mathbf{x}_{1..K}|\mu, \Sigma)$$

assume i.i.d. data

$$= \arg\max_{\mu} \prod_{i=1}^{K} P(\mathbf{x}_i|\mu, \Sigma)$$

$$= \arg\max_{\mu} \sum_{i=1}^{K} \log P(\mathbf{x}_i|\mu, \Sigma)$$

$$= \arg\max_{\mu} \left[-0.5 \sum_{i=1}^{K} (\mathbf{x_i} - \mu)^T \Sigma^{-1} (\mathbf{x_i} - \mu) \right]$$

$$= \arg\min_{\mu} \sum_{i=1}^{K} (\mathbf{x_i} - \mu)^T \Sigma^{-1} (\mathbf{x_i} - \mu)$$

ML FOR THE MEAN OF THE GAUSSIAN

- Standard optimization problem
  - » Take derivatives w.r.t μ and set equal to zero

$$\frac{\partial}{\partial \mu} \sum_{i=1}^{K} (\mathbf{x_i} - \mu)^T \Sigma^{-1} (\mathbf{x_i} - \mu) = 0$$

$$\sum_{i=1}^{K} \frac{\partial}{\partial \mu} (\mathbf{x_i} - \mu)^T \Sigma^{-1} (\mathbf{x_i} - \mu) = 0$$

$$\sum_{i=1}^{K} \frac{\partial}{\partial \mu} \left(\mathbf{x_i}^T \Sigma^{-1} \mathbf{x_i} - 2\mathbf{x_i}^T \Sigma^{-1} \mu + \mu^T \Sigma^{-1} \mu \right) = 0$$

$$\sum_{i=1}^{K} \left(2\Sigma^{-1} \mathbf{x}_i - 2\Sigma^{-1} \mu \right) = 0$$

$$\sum_{i=1}^{K} (\mathbf{x}_i - \mu) = 0$$

ML FOR THE MEAN OF THE GAUSSIAN

- Rearranging,

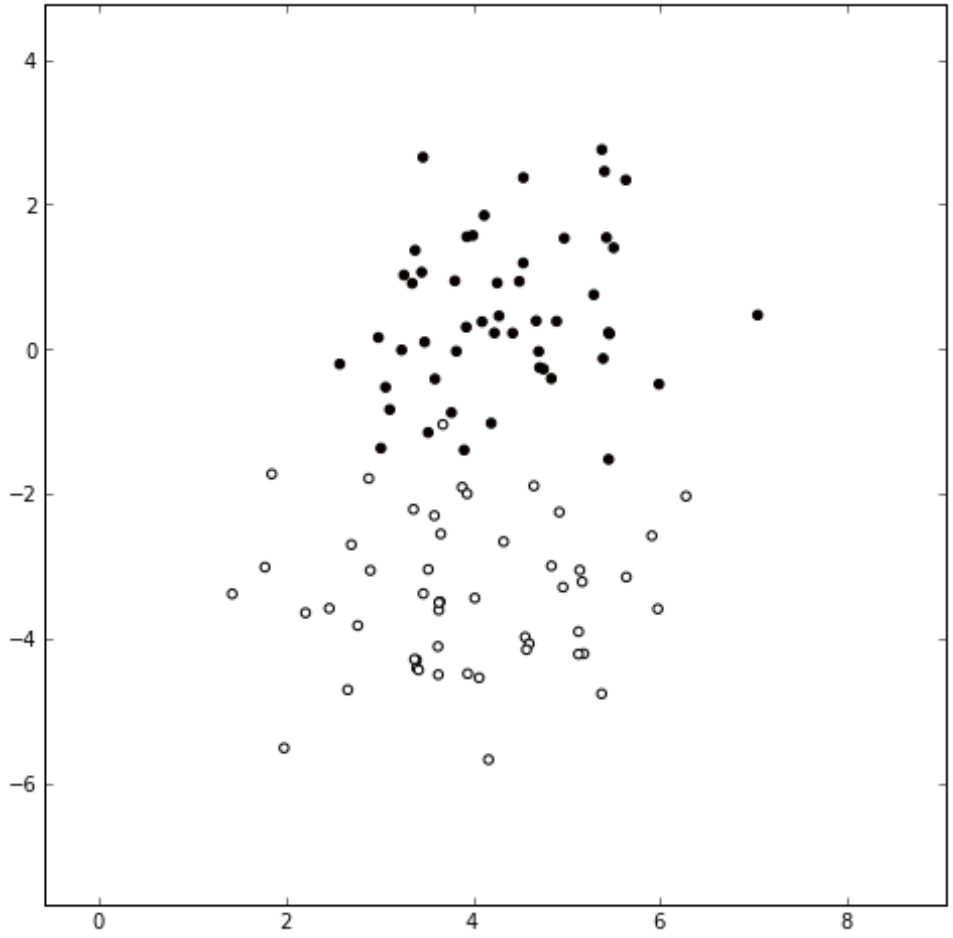
$$\mu = \frac{1}{N} \sum_{i=1}^{K} \mathbf{x}_i$$

- So the ML estimate for the mean of the Gaussian is just the empirical mean of the data points!
- Can do similar for the covariance matrix
  - » We find that the ML estimate is just the empirical covariance matrix for the data
  - » (Calculations are a little more tricky)

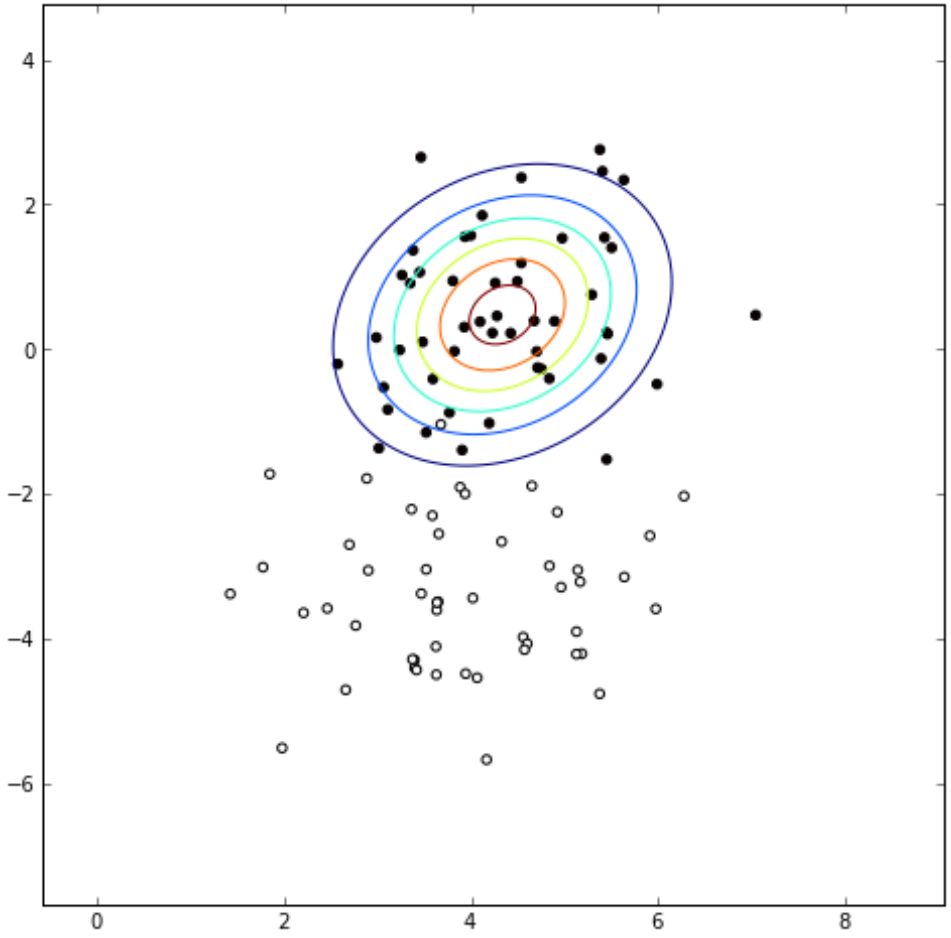
BINARY CLASSIFICATION

- Start with a set of K examples $\{(x_i, y_i) \mid 0 \leq i < K\}$
  - Remember, the x_i's are just feature vectors: points in D-dimensional space
- Separate into:
  - negative training examples $X_{neg} = \{x_i \mid y_i = 0\}$
  - positive examples $X_{pos} = \{x_j \mid y_j = 1\}$
- Training phase:
  - Fit a multivariate Gaussian to X_{neg}
  - Fit a multivariate Gaussian to X_{pos}
- Inference phase:
  - Given an unseen example x
  - Find $P(x \mid y = 0)$ using Gaussian for X_{neg}
  - Find $P(x \mid y = 1)$ using Gaussian for X_{pos}
  - $f(x) = 0$ if $P(x \mid y = 0) > P(x \mid y = 1)$ else 1

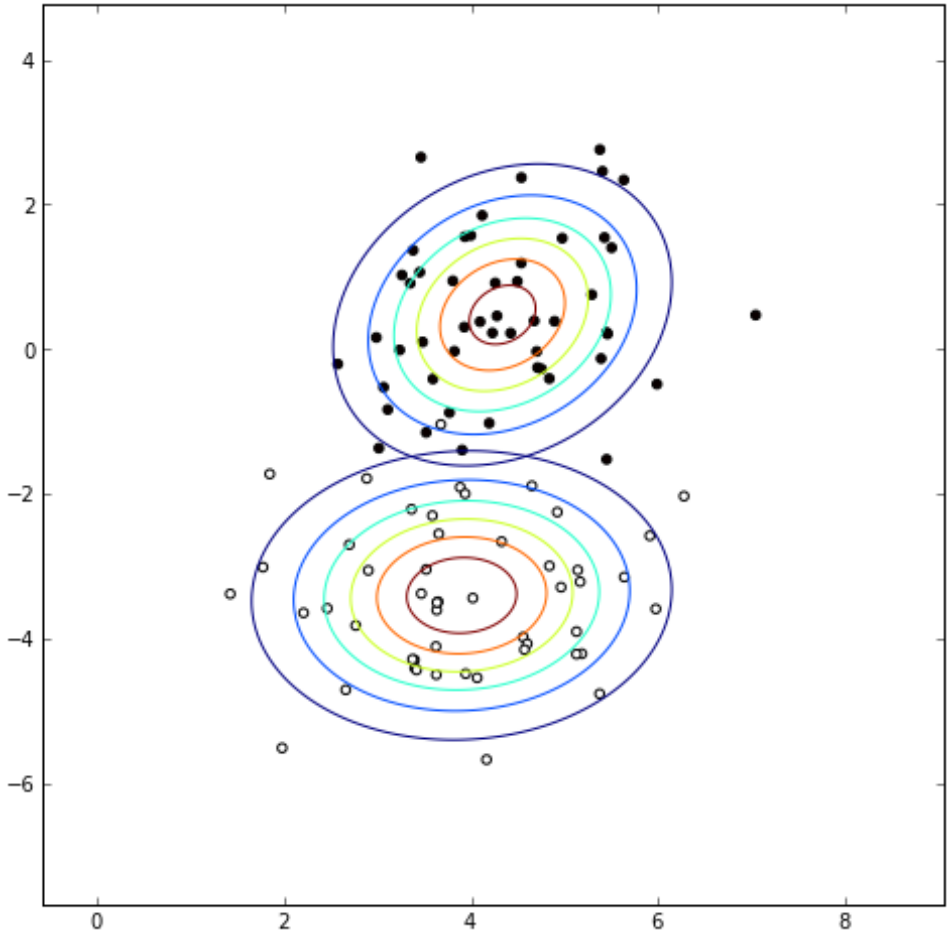
QDA Example



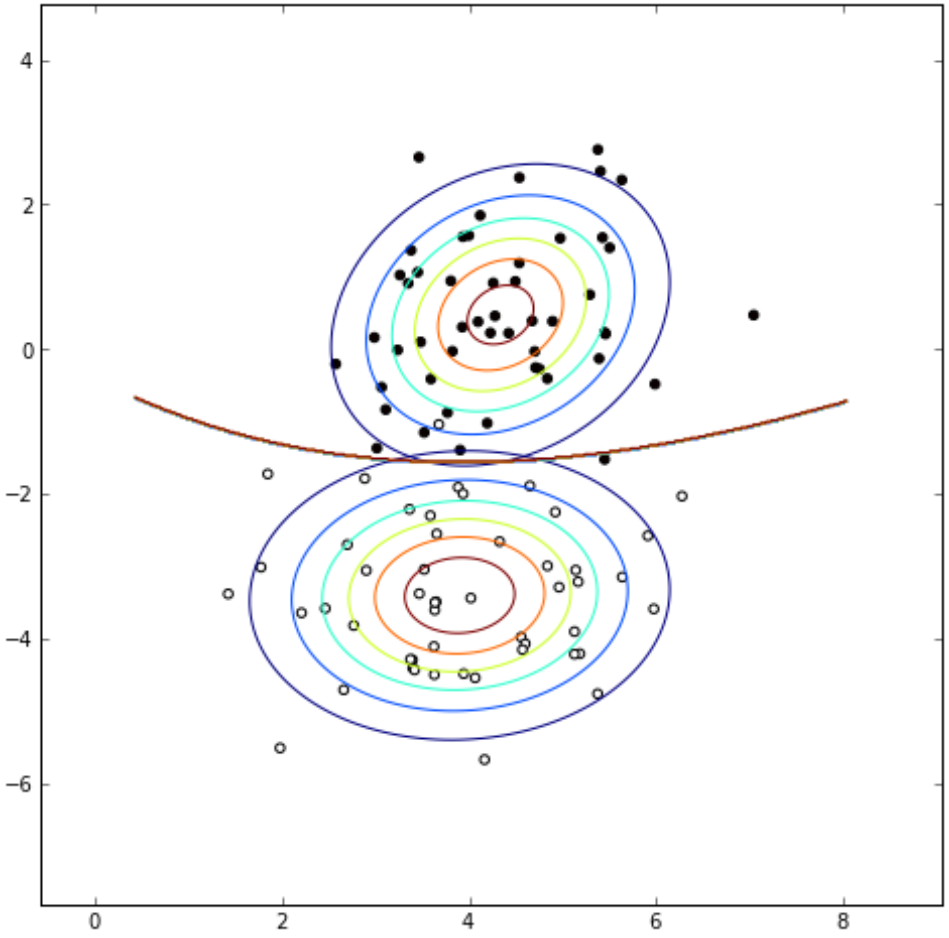
QDA Example



QDA Example



QDA Example



Decision boundary is a quadratic polynomial

QDA TRADEOFFS

Advantages

- Easy to implement
- Fast
- Performs well on a variety of problems

Limitations

- Assumes data are normally distributed in each class
  - » Data may be multimodal
- Works very well if assumption is approximately true
- Can work poorly is assumption is very wrong

Machine Learning Pitfalls

- Insufficiently discriminative features
  - » E.g. person classifier that only uses image width and height as predictors
- Under-fitting
  - » Insufficiently powerful ML model
  - » E.g. Normal assumption invalid
- Over-fitting
  - » Overly complex model
  - » Model becomes tuned to idiosyncrasies in dataset
  - » Does not generalize well to new data
  - » Prevention: split data into train set and test set
  - » Occam's razor: use the simplest model that can explain your data
    - Principle of parsimony

Other Learning Algorithms

- Linear discriminant analysis (LDA)
  - » Like QDA, but assumes equal covariance
- Logistic regression
- Gaussian mixture model classifier (GMM)
- K-nearest neighbors (KNN)
- Decision tree based classifiers
- Support vector machines (SVMs)

Multimedia content analysis

CONCLUSION

SUMMARY

- Applications of multimedia content analysis
- Describing image content with local and global descriptors
- Distance measures for comparing these descriptors
- Machine learning algorithms for classification using these descriptors
- Didn't cover audio or motion descriptors
  - » Principles are the same

Some Resources

- Books
  - » Computer vision: models, inference, and learning
  - » Computer vision: algorithms and applications
  - » The elements of statistical learning
- Software
  - » VLFeat (University of Oxford)
  - » scikits-image (Python)
  - » scikits-learn (Python)
  - » Libsvm (C++)
  - » OpenCV (C/C++)

QUESTIONS?