INFO0948
Feature Extraction

Renaud Detry

University of Liège, Belgium

Updated November 27, 2013

These slides are based on Chapter 13 of the book *Robotics, Vision and Control: Fundamental Algorithms in MATLAB* by Peter Corke, published by Springer in 2011.
The Bag-of-feature section is based on a presentation by Cordelia Schmid `http://www.di.ens.fr/willow/events/cvml2011/materials/CVML2011_Cordelia_bof.pdf`

Motivation

Raw images contain too much data to be of direct practical use for high(er)-level robot vision (object recognition, pose estimation, tracking, ...).

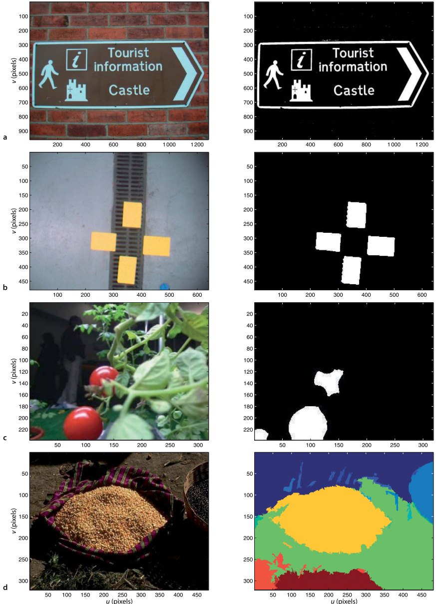
We need to reduce the dimensionality of raw image data, ideally focusing on

- discarding redundant information.
- extracting entities that are invariant to the conditions that typically change while a robot is working (viewpoint, illumination, ...).

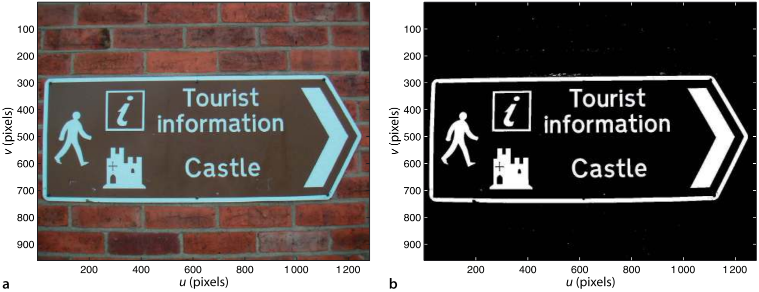
Feature extraction is an information concentration step that reduces the data rate from 10^6–10^8 bytes s^{-1} at the output of a camera to something of the order of tens of features (vectors of a few dozen scalars) per frame that can be used as input to a robot's control system.

Plan

Region Features



Thresholding



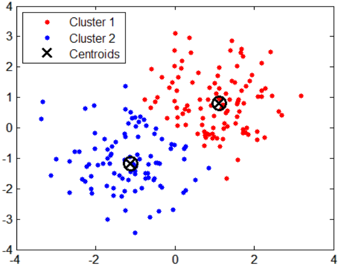
$$c[u,v] = \begin{cases} 0 & I[u,v] < t \\ 1 & I[u,v] \geq t \end{cases} \quad \forall (u,v) \in I$$

The k-means Algorithm

Given a set of observations $(x_1, x_2, ..., x_n)$, where each observation is a d-dimensional real vector, k-means clustering aims to partition the n observations into k sets $(k \leq n)$ $S = \{S_1, S_2, \ldots, S_k\}$ so as to minimize the within-cluster sum of squares (WCSS)

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in S_i} \left\| \mathbf{x}_j - \boldsymbol{\mu}_i \right\|^2$$

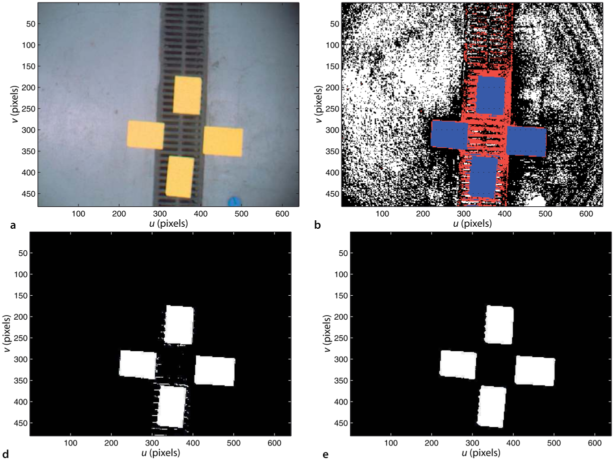
where μ_i is the mean of points in S_i.




```
>> [cls,centre,r] = kmeans(a, 3);
```

Color Clustering and Classification

```
>> [cls, cxy, resid] = colorkmeans(im_targets, 4);
```



Plan

Point Features

Point features relate to two problems:

- Identifying "interesting" points (also called salient points or keypoints).
- Characterizing patches of pixels (centered on an interest point or not).

Corner Detectors

Corners are characterized by a strong gradient in two orthogonal directions. Let us define

$$s(u, v, \delta_u, \delta_v) = \sum_{(i,j)\in\mathcal{W}} \boldsymbol{W}[i, j]\left(\underbrace{\boldsymbol{I}[u + \delta_u + i, v + \delta_v + j]} - \boldsymbol{I}[u + i, v + j]\right)^2$$

An interest point is one for which s is high for all directions of the vector (δ_u, δ_v).

The underlined term can be approximated (Taylor) with

$$\boldsymbol{I}[u + \delta_u + j, v + \delta_v + i] \approx \boldsymbol{I}[u + i, v + j] + \boldsymbol{I}_u[u + i, v + j]\delta_u + \boldsymbol{I}_v[u + i, v + j]\delta_v$$

Corner Detectors

$$s(u,v,\delta_u,\delta_v) = \sum_{(i,j)\in\mathcal{W}} \boldsymbol{W}[i,j]\Big(\underbrace{\boldsymbol{I}[u+\delta_u+i, v+\delta_v+j]} - \boldsymbol{I}[u+i,v+j]\Big)^2$$

$$\boldsymbol{I}[u+\delta_u+j, v+\delta_v+i] \approx \boldsymbol{I}[u+i,v+j] + \boldsymbol{I}_u[u+i,v+j]\delta_u + \boldsymbol{I}_v[u+i,v+j]\delta_v$$

$$\begin{aligned}
s(u,v,\delta_u,\delta_v) &= \sum_{(i,j)\in\mathcal{W}} \boldsymbol{W}[i,j]\big(\boldsymbol{I}_u[u+i,v+j]\delta_u + \boldsymbol{I}_v[u+i,v+j]\delta_v\big)^2 \\
&= \delta_u^2 \sum_{(i,j)\in\mathcal{W}} \boldsymbol{W}[i,j]\,\boldsymbol{I}_u^2[u+i,v+j] + \delta_v^2 \sum_{(i,j)\in\mathcal{W}} \boldsymbol{W}[i,j]\,\boldsymbol{I}_v^2[u+i,v+j] \\
&\quad + 2\delta_u\delta_v \sum_{(i,j)\in\mathcal{W}} \boldsymbol{W}[i,j]\,\boldsymbol{I}_u[u+i,v+j]\,\boldsymbol{I}_v[u+i,v+j]
\end{aligned}$$

$$\boldsymbol{A} = \begin{pmatrix} \Sigma \boldsymbol{W}[i,j]\,\boldsymbol{I}_u^2[u+i,v+j] & \Sigma \boldsymbol{W}[i,j]\,\boldsymbol{I}_u[u+i,v+j]\,\boldsymbol{I}_v[u+i,v+j] \\ \Sigma \boldsymbol{W}[i,j]\,\boldsymbol{I}_u[u+i,v+j]\,\boldsymbol{I}_v[u+i,v+j] & \Sigma \boldsymbol{W}[i,j]\,\boldsymbol{I}_v^2[u+i,v+j] \end{pmatrix}$$

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{G}(\sigma_I)\otimes \boldsymbol{I}_u^2 & \boldsymbol{G}(\sigma_I)\otimes \boldsymbol{I}_u\boldsymbol{I}_v \\ \boldsymbol{G}(\sigma_I)\otimes \boldsymbol{I}_u\boldsymbol{I}_v & \boldsymbol{G}(\sigma_I)\otimes \boldsymbol{I}_v^2 \end{pmatrix}$$

$$\boxed{s(u,v,\delta_u,\delta_v) = (\delta_u \quad \delta_v)\boldsymbol{A}\begin{pmatrix}\delta_u \\ \delta_v\end{pmatrix}}$$

The Harris Corner Detector

$$s(u,v,\delta_u,\delta_v) = \begin{pmatrix}\delta_u & \delta_v\end{pmatrix} \boldsymbol{A} \begin{pmatrix}\delta_u \\ \delta_v\end{pmatrix} \qquad \boldsymbol{A} = \begin{pmatrix} \boldsymbol{G}(\sigma_I) \otimes \boldsymbol{I}_u^2 & \boldsymbol{G}(\sigma_I) \otimes \boldsymbol{I}_u \boldsymbol{I}_v \\ \boldsymbol{G}(\sigma_I) \otimes \boldsymbol{I}_u \boldsymbol{I}_v & \boldsymbol{G}(\sigma_I) \otimes \boldsymbol{I}_v^2 \end{pmatrix}$$

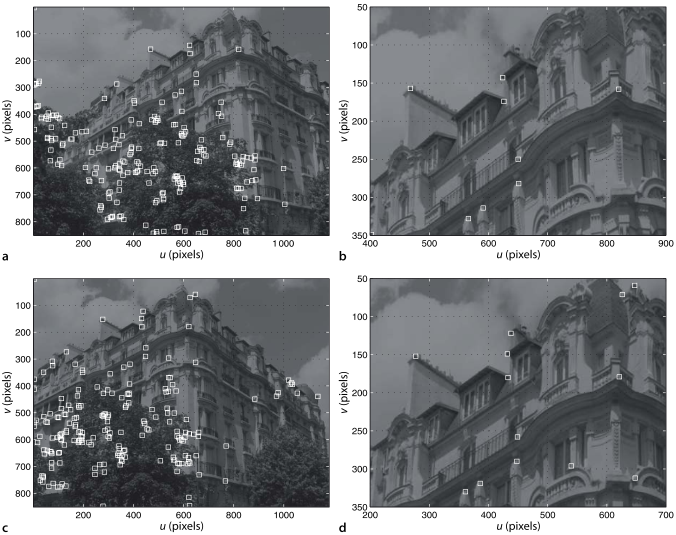
s is high for all directions of the vector (δ_u, δ_v) when $\boldsymbol{A}$ has two large eigenvalues.

The Harris detector is defined as

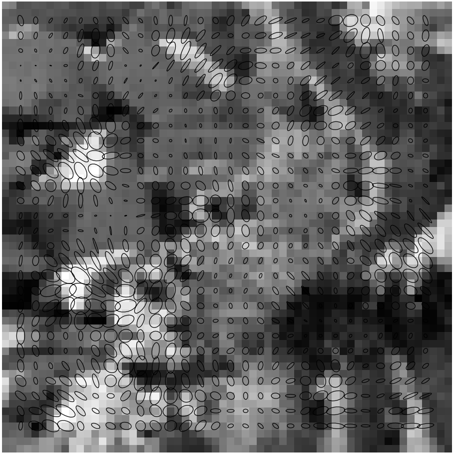
$$C_{\mathrm{H}}(u,v) = \det(\boldsymbol{A}) - k\,\mathrm{tr}^2(\boldsymbol{A})$$

It has a large value around sharp corners.

The Harris Corner Detector



The Harris Corner Detector



Ellipses $(u, v)^T \boldsymbol{A}^{-1}(u, v)$.
Their major and minor axes are along the eigenvectors of $\boldsymbol{A}$, and the extent of the ellipses along their major or minor axes corresponds to the size of the eigenvalues.

A large circle corresponds to a corner and a narrow extended ellipse indicates an edge.

D. Forsyth and J. Ponce. Computer vision: a modern approach. PHPTR, 2002.

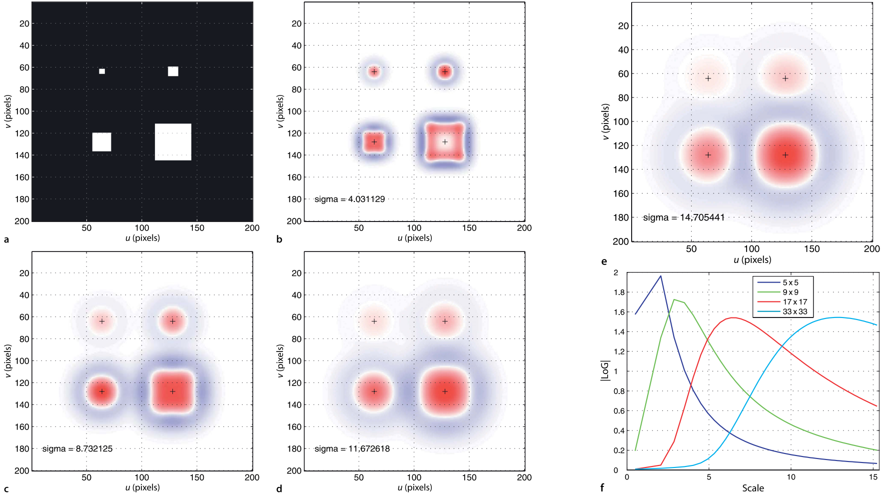
The Harris Corner Detector

The Harris detector is computed from image gradients and is therefore robust to offsets in illumination, and the eigenvalues of the structure tensor $\boldsymbol{A}$ are invariant to rotation. However the detector is not invariant to changes in scale.

The Harris detector responds strongly to fine texture (see leaves above), but does not detect larger structures.

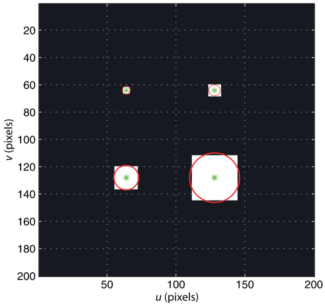
Plan

Scale-space Detector



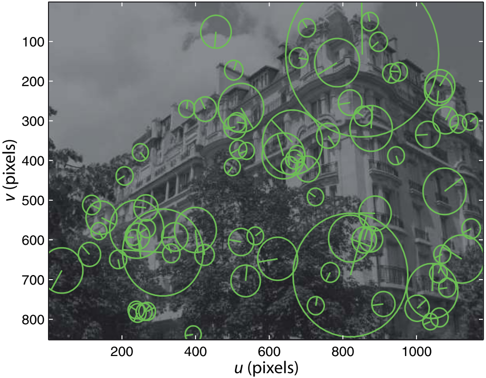
$$\nabla^2 \mathbf{I} = \frac{\partial^2 \mathbf{I}}{\partial u^2} + \frac{\partial^2 \mathbf{I}}{\partial v^2} = \mathbf{I}_{uu} + \mathbf{I}_{vv} = \mathbf{L} \otimes \mathbf{I}$$

Scale-space Detector



Scale-Space Point Feature

Scale-space maximum detection is at the root of SIFT and SURF features.



Plan

Object/Category Recognition

Image classification: assigning a class label to the image

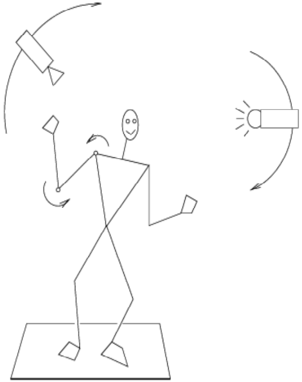
Car: present
Cow: present
Bike: not present
Horse: not present
...

Object localization: define the location and the category

Location

Category

Challenge 1: Intra-instance Variations



Viewpoint, illumination, kinematic configuration, ...

Challenge 2: Intra-class Variations

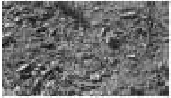
Image Classification, Case 1: Detection

Given

- Positive examples (containing bike)



- Negative examples (containing no bike)



Decide whether a test image contains a bike or not

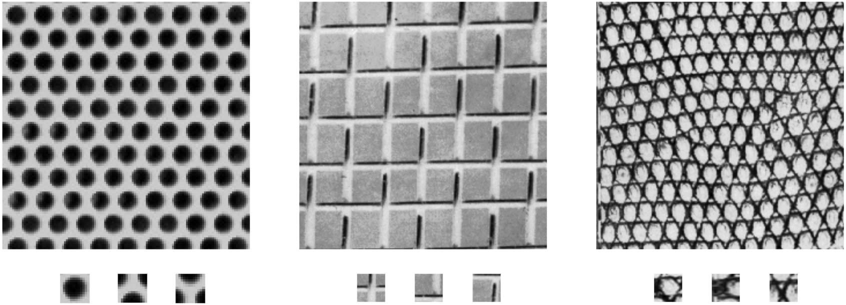
Image Classification, Case 2: Recognition

Given

bikes books building cars people phones trees

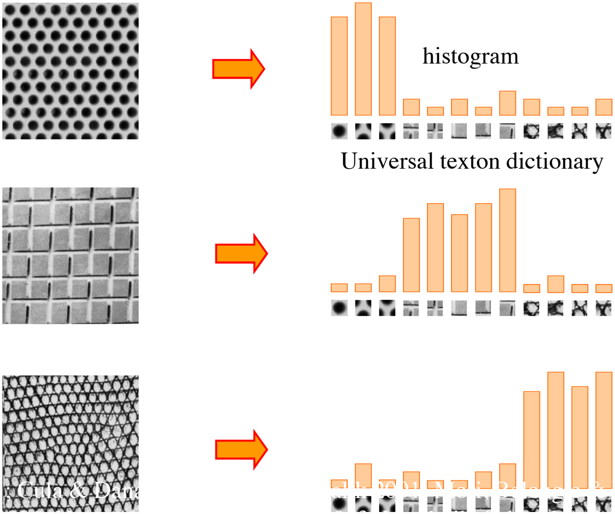
Decide what object is the most likely object in an image

BoF Origin: Texture Classification



Julesz, 1981; Cula & Dana, 2001; Leung & Malik 2001; Mori, Belongie & Malik, 2001; Schmid 2001; Varma & Zisserman, 2002, 2003; Lazebnik, Schmid & Ponce, 2003

Texture Classification: Histograms over Textons



Bag-of-features for Image Classification

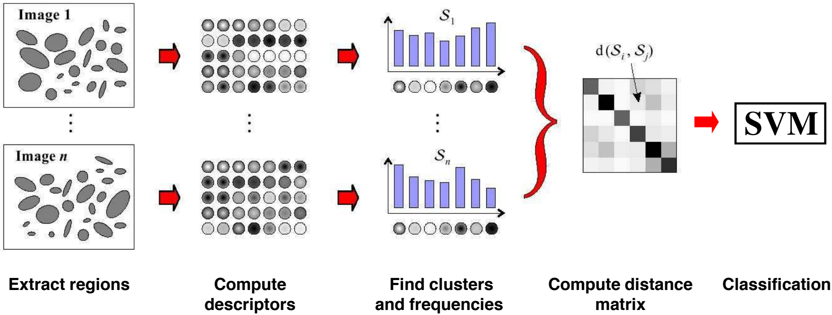
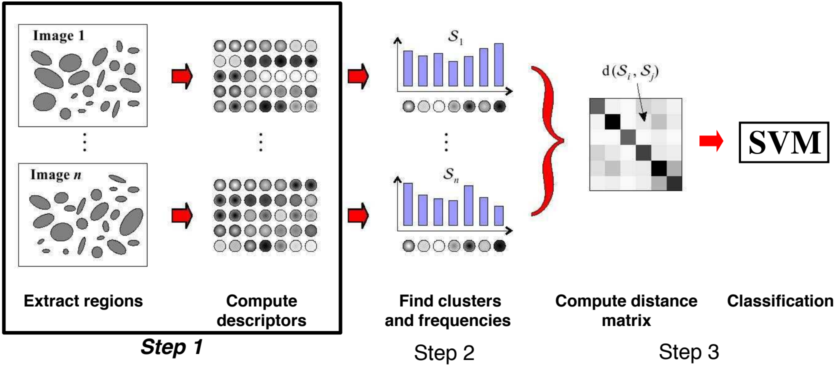


Image 1 contains a bike, image 2 contains a horse, image 3 contains a car, etc...

[Csurka et al., ECCV Workshop'04], [Nowak, Jurie, Triggs, ECCV'06], [Zhang, Marszalek, Lazebnik, Schmid, IJCV'07]

Plan

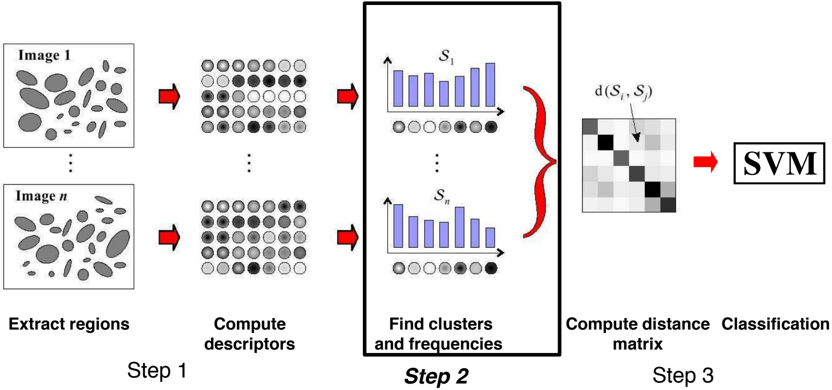
Step 1: Extract Features



SIFT features are a popular choice.

Plan

Step 2: Cluster Features, Compute Feature Frequencies



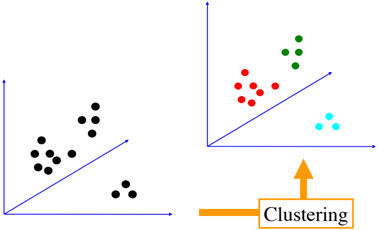
Clustering Features (k-means)

Because of viewpoint and lighting changes, it is unlikely that two images of even the same object will produce exactly the same SIFT features.

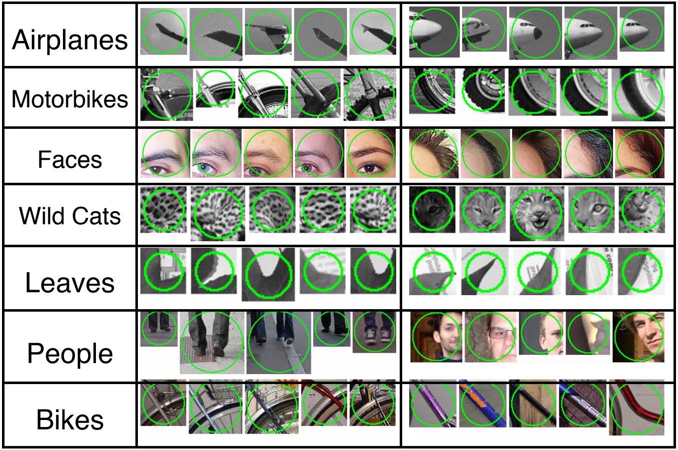
This gets even worse when working with different instances of a class (e.g., different cars).

As a result, it is not a good idea to model an object (or object class) with a histogram over all the features that the object produces.

Instead, we cluster *all* the features that come from the training images (of all classes), and keep only the cluster centers. The set of cluster centers is called a codebook, or dictionary. The elements of the codebook are called codewords.



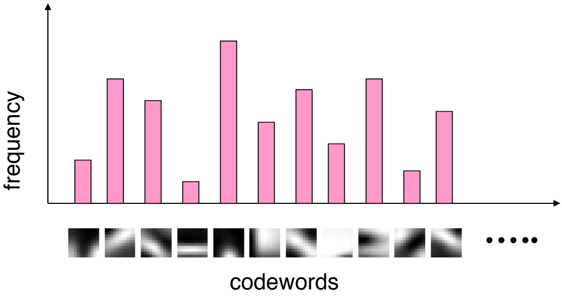
Examples of Clusters of Features



↓ Average of these becomes one codeword

↓ Average of these becomes one codeword

Object/Class Instance Representation: Codeword Frequencies



Typically: 1000–4000 codewords:

- More codewords: towards object representation
- Less codewords: towards object class representation

One image of an instance of an object/class is represented with a vector V of frequencies of the codewords. (L1/L2 normalization)

Plan

Step 3: Image Classification

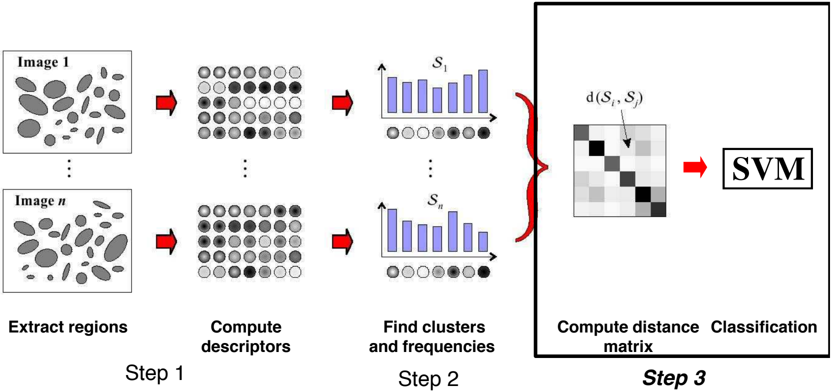
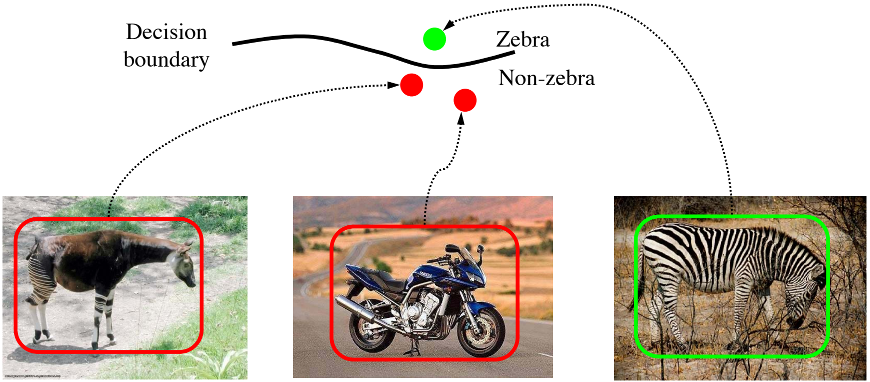
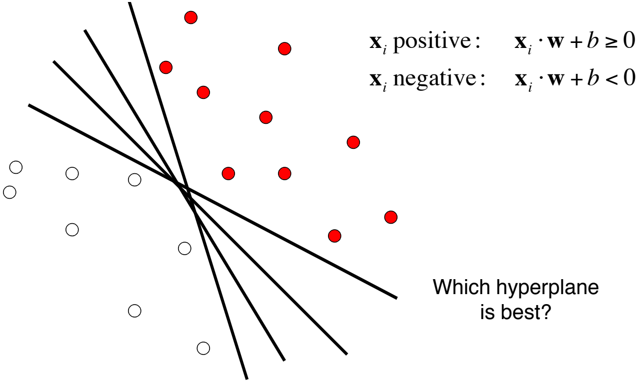


Image Classification

Goal: Learn a decision rule (classifier) to assign V to an object/class.



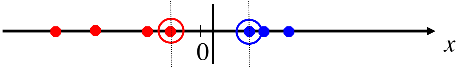
Linear Classification



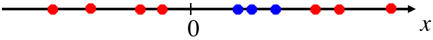
For instance: support vector machines (SVM), logistic regression, linear discriminant analysis (LDA), naive Bayes classification, ...

Nonlinear Classification

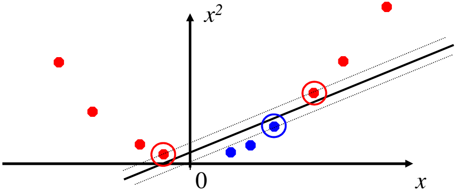
Datasets that are linearly separable work out great:



But what if the data set is just too hard?



Map the data to a higher dimensional space where it is linearly separable:



Nonlinear Classification

For instance: Kernel-SVM, kernel logistic regression. A radial basis function usually works well.

See

ELEN0062-1
Apprentissage inductif appliqué – Pierre Geurts, Louis Wehenkel

Bag-of-features: Summary

Advantages:

- Robust to position and orientation
- Very successful at modeling inter/intra-class variability.

Shortcomings:

- No explicit encoding of an object's structure
    - Two objects of similar local appearance but different global shapes are not discernible.
    - Poor at computing an object's position or orientation in an image.