

Computer Vision – TP7

Pattern Recognition

Miguel Coimbra, Hélder Oliveira

Outline

- Introduction to Pattern Recognition
- Statistical Pattern Recognition
- Visual Features
- Detection of interest points
- Local invariant descriptors

Topic: Introduction to Pattern Recognition

- Introduction to Pattern Recognition
- Statistical Pattern Recognition
- Visual Features
- Detection of interest points
- Local invariant descriptors

This is a
horse

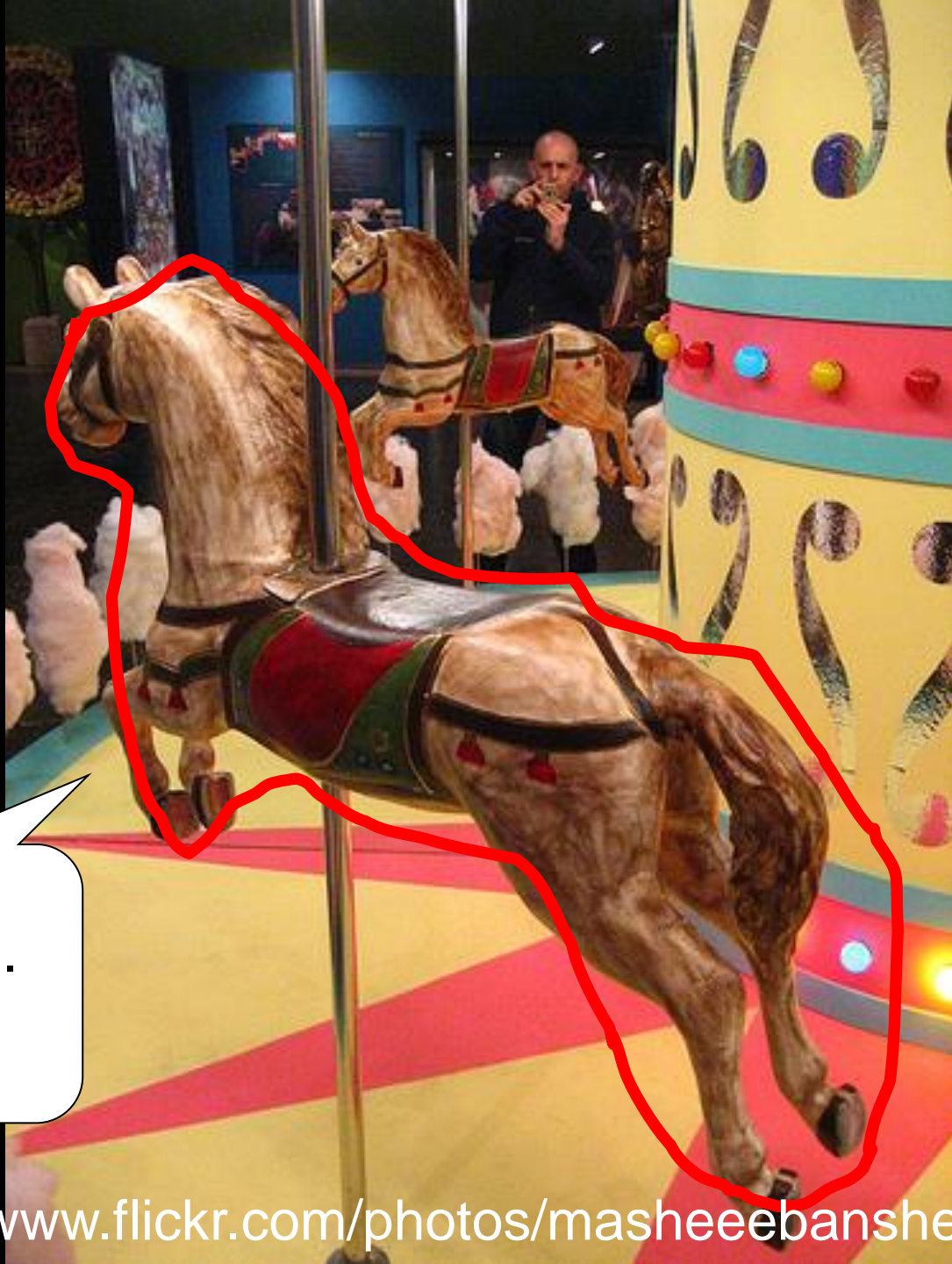


This is a
horse



andrea.lindenberg © 2007

<http://www.flickr.com/photos/genewolf/2031802050/>



This is a...
Horse?

One definition

- **Pattern recognition**

"the act of taking in raw data and taking an action based on the category of the data"

Wikipedia

- How do I do this so well?
- How can I make machines do this?

The problem



Do you 'see'
a horse?

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

What a computer sees

Mathematics

- **We only deal with numbers.**
 - How do we represent knowledge?
 - How do we represent visual features?
 - How do we classify them?
- **Very complex problem!!**
 - Let's break it into smaller ones...

Typical PR system

Sensor

Gathers the observations to be classified or described



Feature Extraction

Computes numeric or symbolic information from the observations;



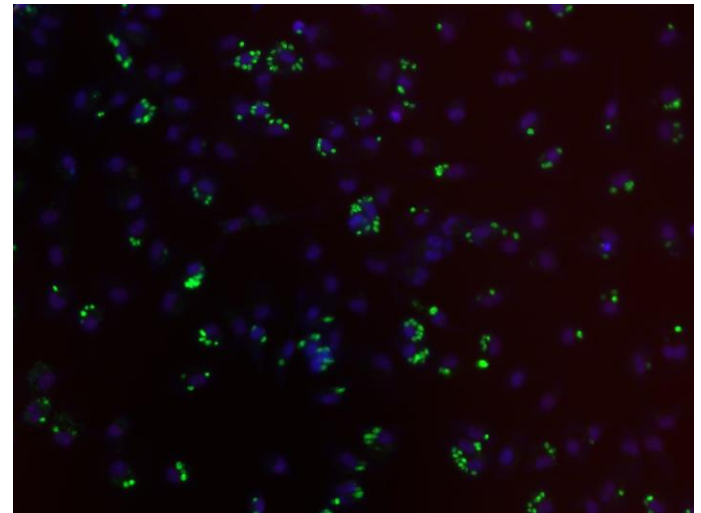
Classifier

Does the actual job of classifying or describing observations, relying on the extracted features.

Sensor

- In our specific case:
 - Image acquiring mechanism
 - Let's assume we don't control it

One observation = One Image
Video = Multiple Observations



Feature Extraction

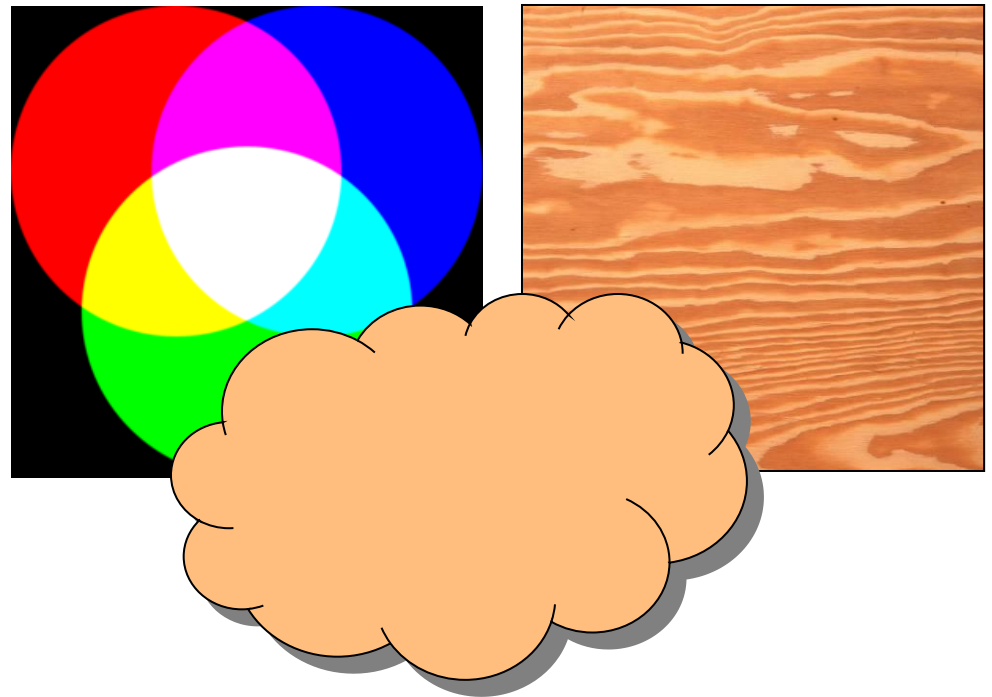
- **What exactly are features?**
 - Colour, texture, shape, etc
 - Animal with 4 legs
 - Horse
 - Horse jumping
- **These vary a lot!**
- **Some imply some sort of ‘recognition’**
e.g. How do I know the horse is jumping?

Broad classification of features

- **Low-level**
 - Colour, texture
- **Middle-level**
 - Object with head and four legs
 - Object moving up
 - Horse
- **High-level**
 - Horse jumping
 - Horse competition

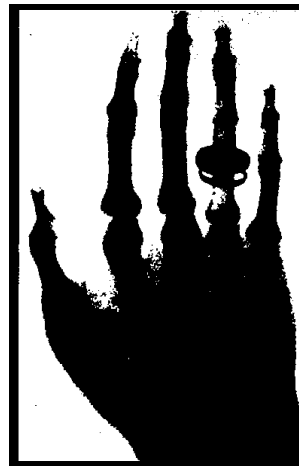
Low-level features

- Objective
- Directly reflect specific image and video features
 - Colour
 - Texture
 - Shape
 - Motion
 - Etc.



Middle-level features

- Some degree of subjectivity
- They are typically one solution of a problem with multiple solutions
- Examples:
 - Segmentation
 - Optical Flow
 - Identification
 - Etc.



High-level features

- Semantic Interpretation
- Knowledge
- Context
- Examples:



How do humans do this so well?

- This person suffers from epilepsy
- The virus attacks the cell with some degree of intelligence
- This person is running from that one

The semantic gap

- Fundamental problem of current research!

Low-level:

- Colour
- Texture
- Shape
- ...



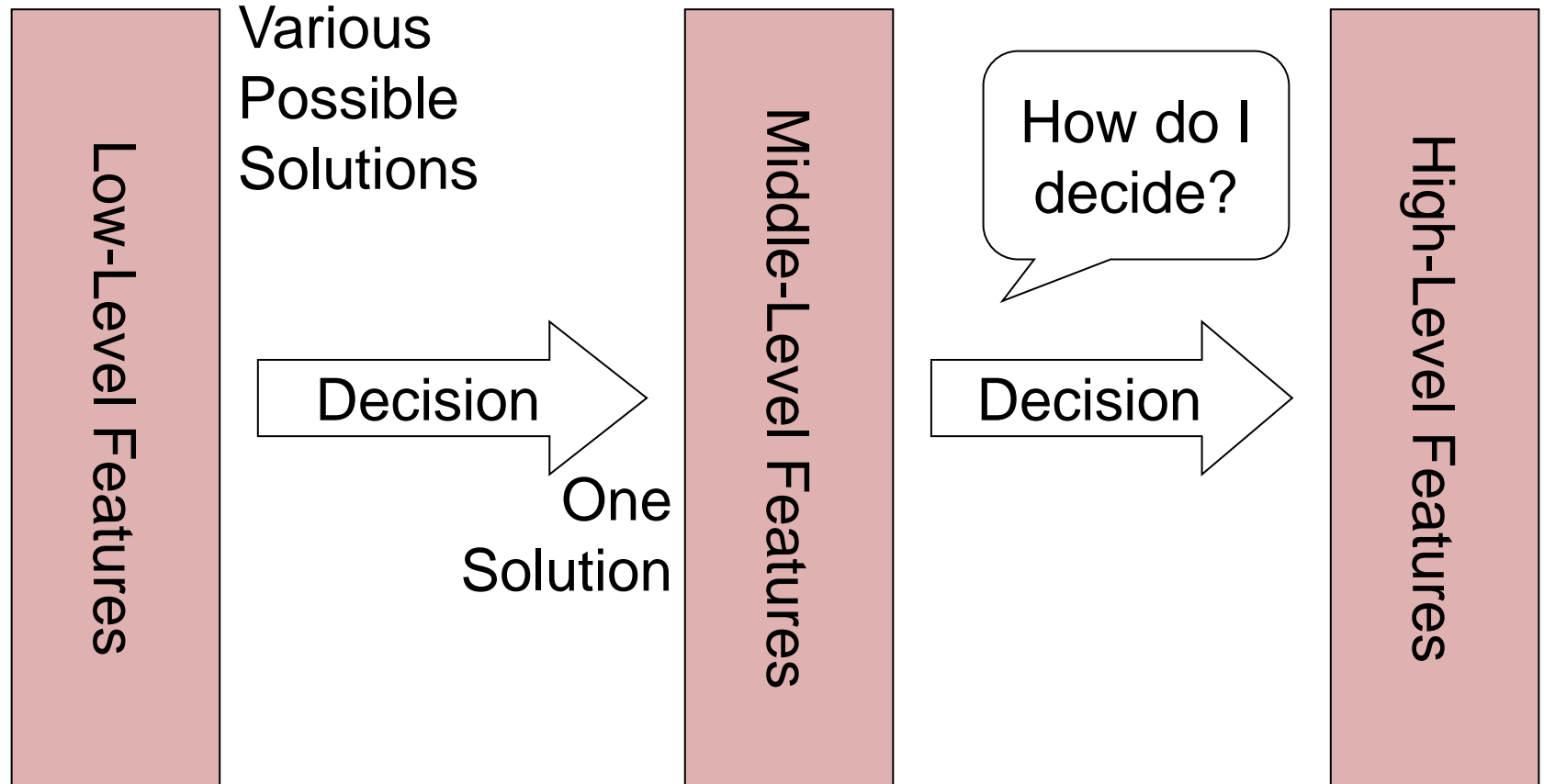
High-level:

- Interpretation
- Decision
- Understanding
- ...

Now what??
How do i cross this
bridge?



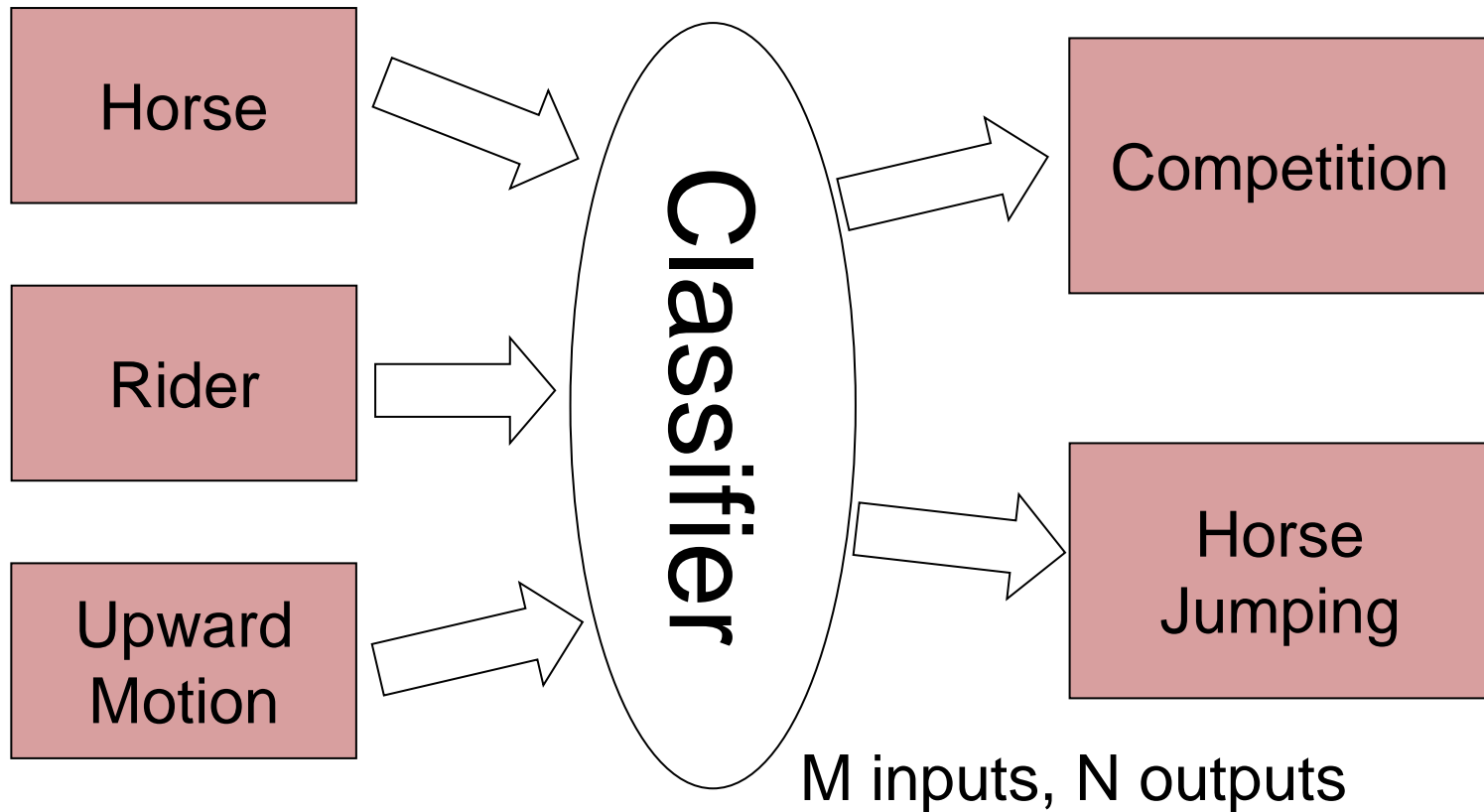
Features & Decisions



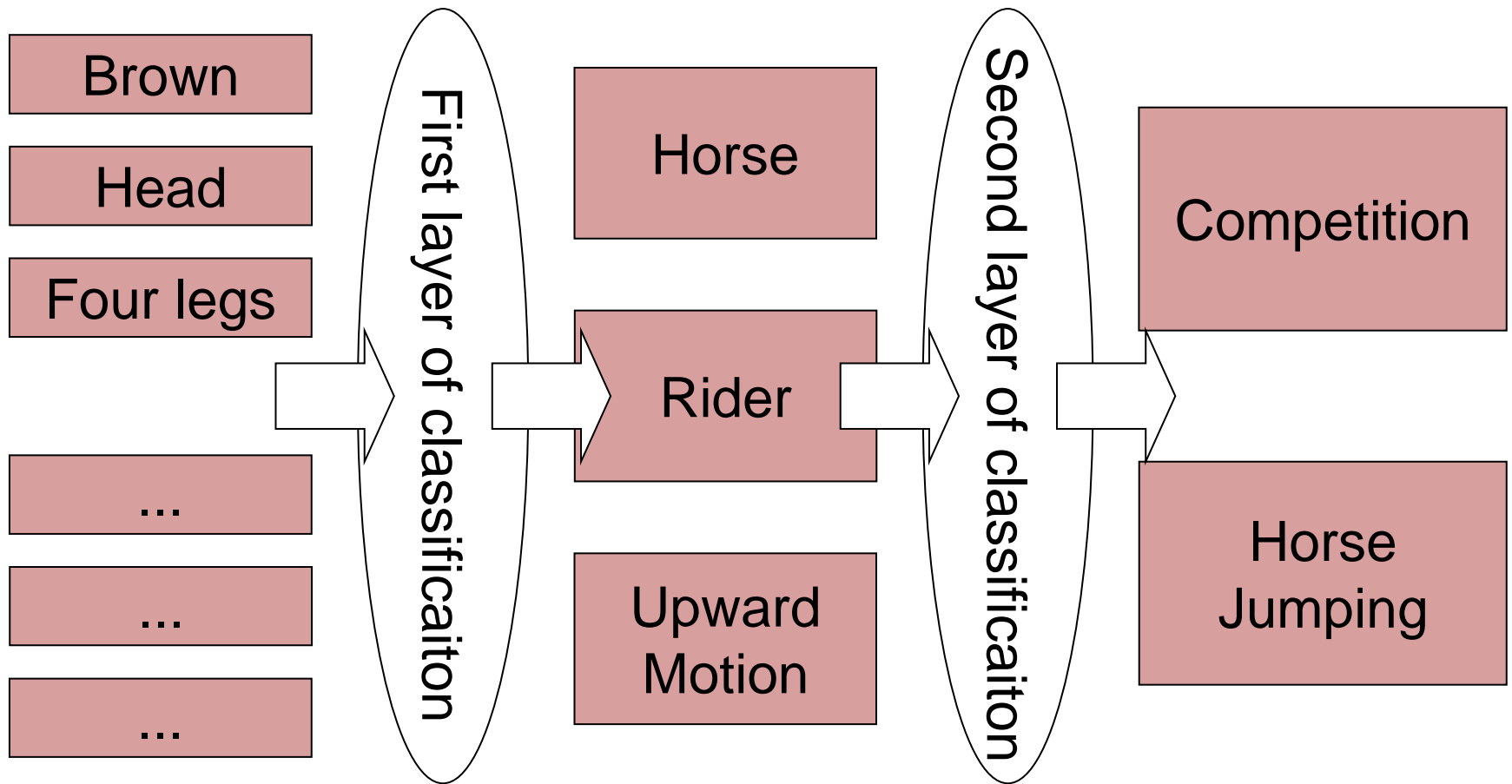
Classification

Middle-Level Features

High-Level Features



Layers of classification



Classifiers

- How do I map my M inputs to my N outputs?
- Mathematical tools:
 - Distance-based classifiers
 - Rule-based classifiers
 - Support Vector Machines
 - Neural Networks
 - ...

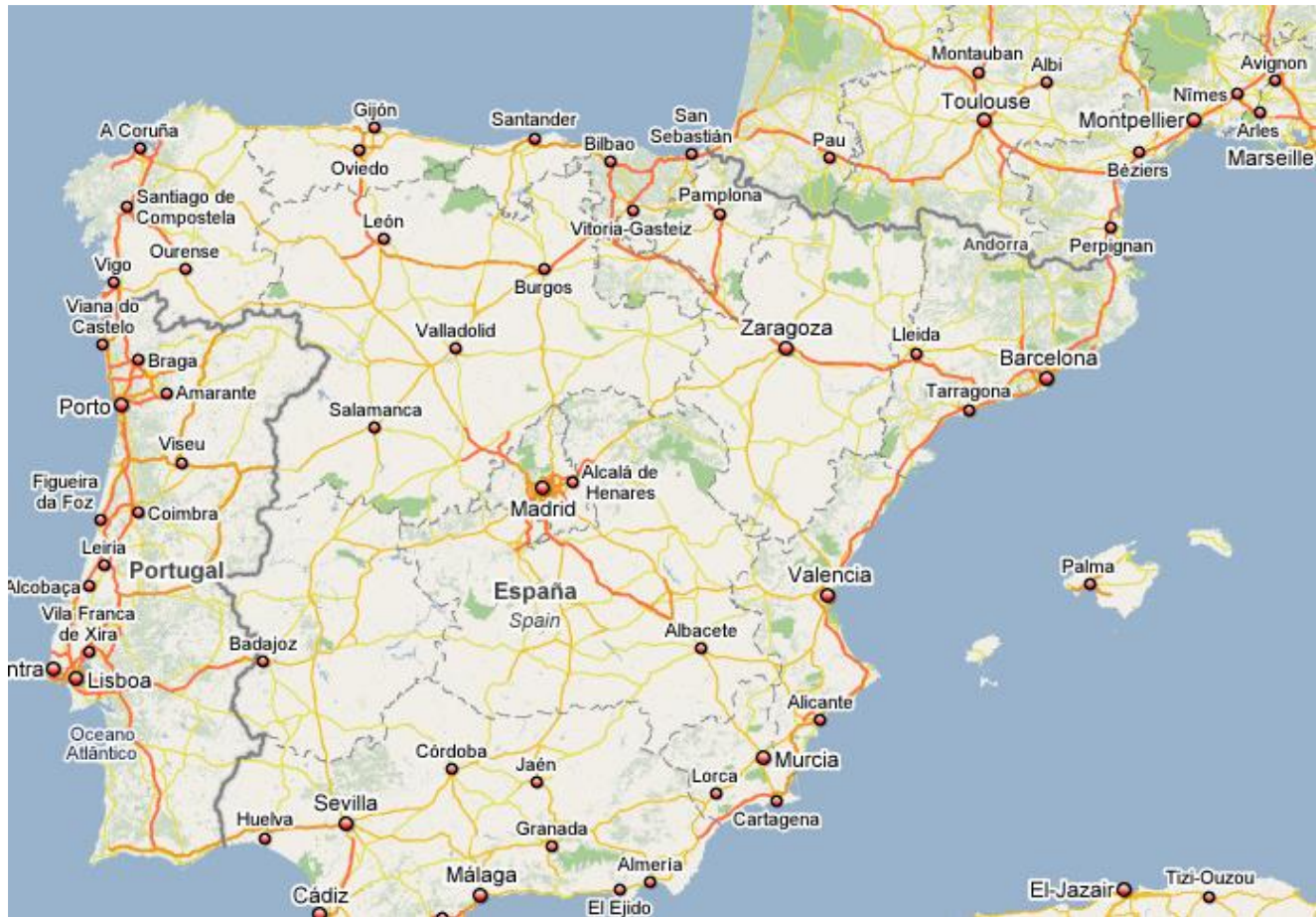
Types of PR methods

- **Statistical pattern recognition**
 - based on statistical characterizations of patterns, assuming that the patterns are generated by a probabilistic system
- **Syntactical (or structural) pattern recognition**
 - based on the structural interrelationships of features

Topic: Statistical Pattern Recognition

- Introduction to Pattern Recognition
- **Statistical Pattern Recognition**
- Visual Features
- Detection of interest points
- Local invariant descriptors

Is Porto in Portugal?



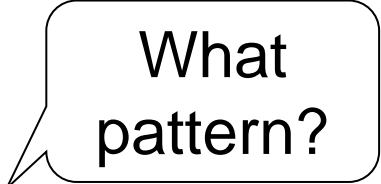
Porto is in Portugal

- I want to make decisions
 - Is Porto in Portugal?
- I know certain things
 - A world map including cities and countries
- I can make this decision!
 - Porto is in Portugal
- I had enough *a priori* knowledge to make this decision

What if I don't have a map?

- I still want to make this decision
- I observe:
 - Amarante has coordinates x_1, y_1 and is in Portugal
 - Viseu has coordinates x_2, y_2 and is in Portugal
 - Vigo has coordinates x_3, y_3 and is in Spain
- I classify:
 - Porto is close to Amarante and Viseu so **Porto is in Portugal**
- What if I try to classify *Valença*?

Statistical PR

- I used **statistics** to make a decision
 - I can make **decisions** even when I don't have full a priori knowledge of the whole process
 - I can make **mistakes**
- How did I **recognize** this pattern?
 - I **learned** from previous observations where I knew the classification result
 - I **classified** a new observation

Back to the Features

- Feature F_i $F_i = [f_i]$

- Feature F_i with N values.

$$F_i = [f_{i1}, f_{i2}, \dots, f_{iN}]$$

- Feature vector F with M features.

$$F = [F_1 | F_2 | \dots | F_M]$$

- Naming conventions:
 - Elements of a **feature vector** are called **coefficients**
 - **Features** may have one or more **coefficients**
 - **Feature vectors** may have one or more **features**

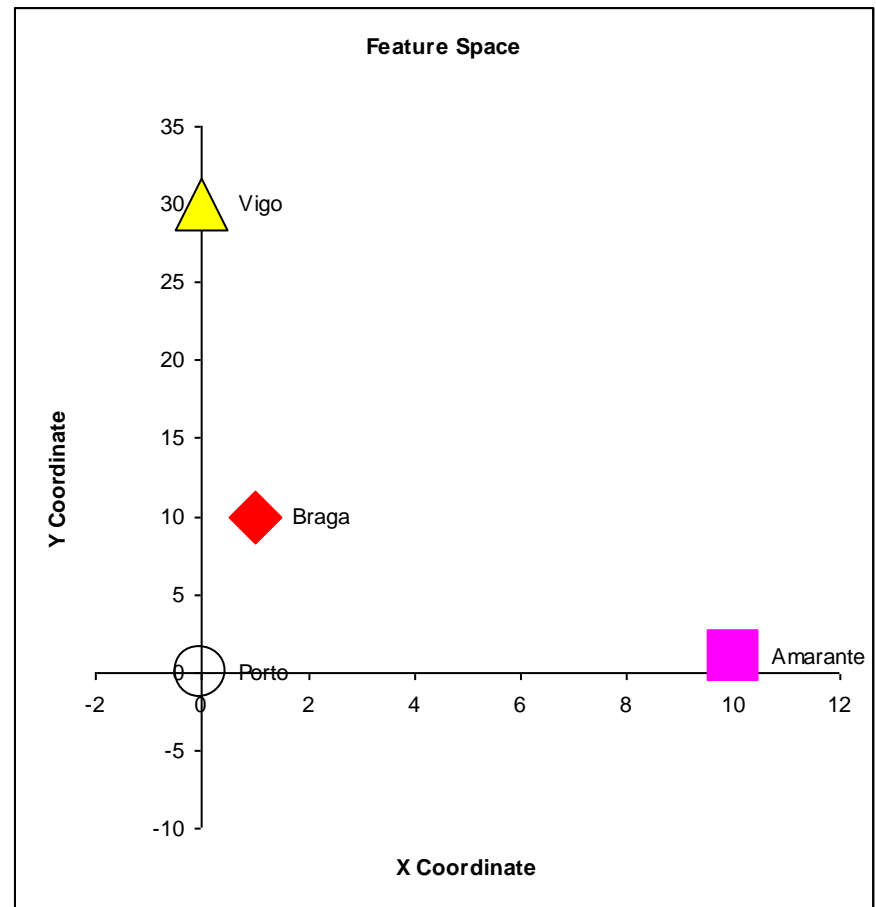
Back to our Porto example

- I've classified that Porto is in Portugal
- What feature did I use?
 - Spatial location
- Let's get more formal
 - I've defined a feature vector \mathbf{F} with one feature \mathbf{F}_1 , which has two coefficients f_{1x} , f_{1y}

$$\mathbf{F} = [\mathbf{F}_1] = [f_{1x}, f_{1y}]$$

Feature Space

- **Feature Vector**
 - Two total coefficients
 - Can be seen as a feature 'space' with two orthogonal axis
- **Feature Space**
 - Hyper-space with N dimensions where N is the total number of coefficients of my feature vector



A *Priori* Knowledge

- I have a precise **model** of my feature space based on **a priori** knowledge

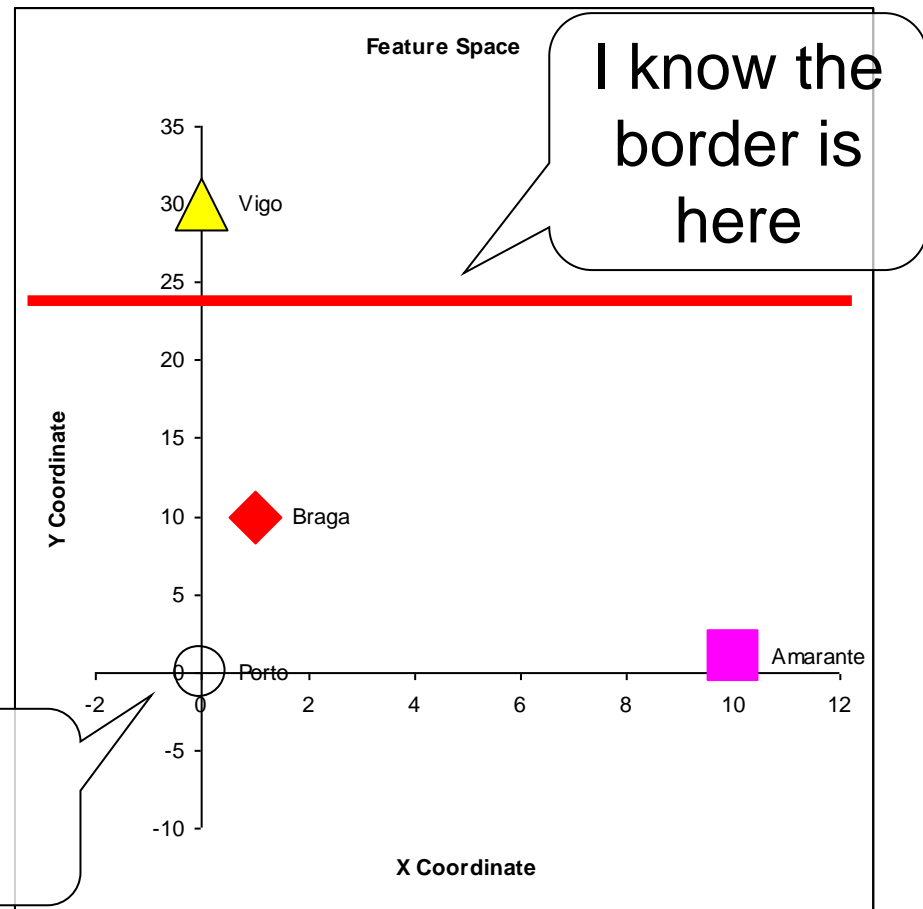
City is in Spain if $F_{1Y} > 23$

- Great models = Great classifications

$F_{1Y}(\text{London}) = 100$

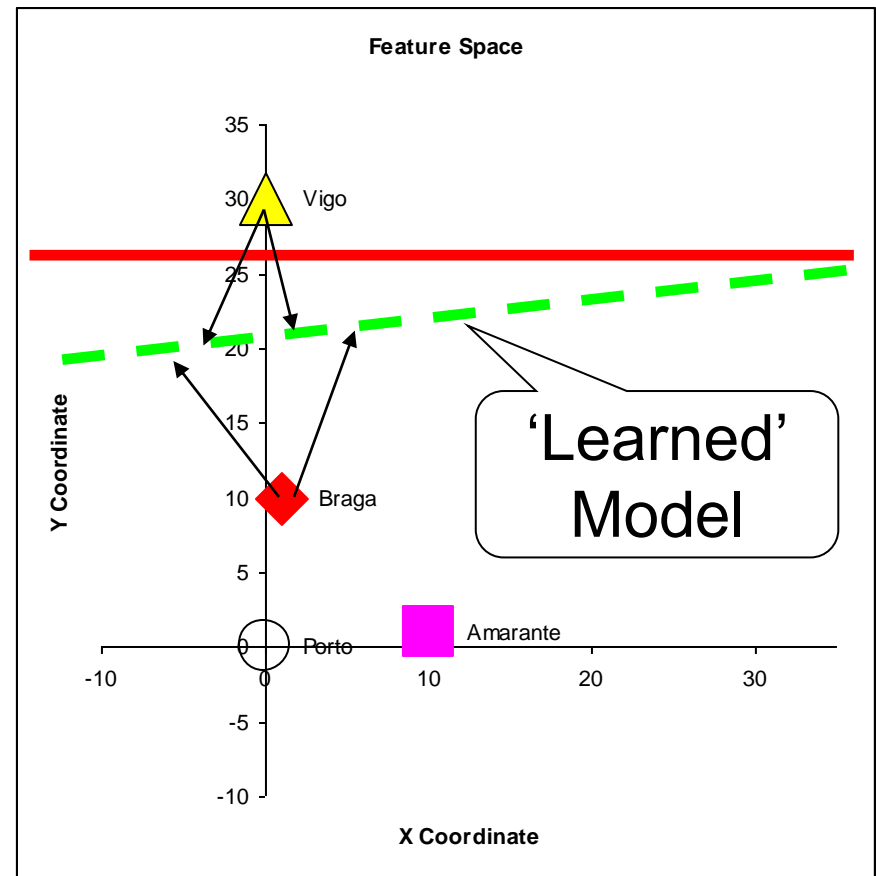
London is in Spain (??)

Porto **is** in Portugal!



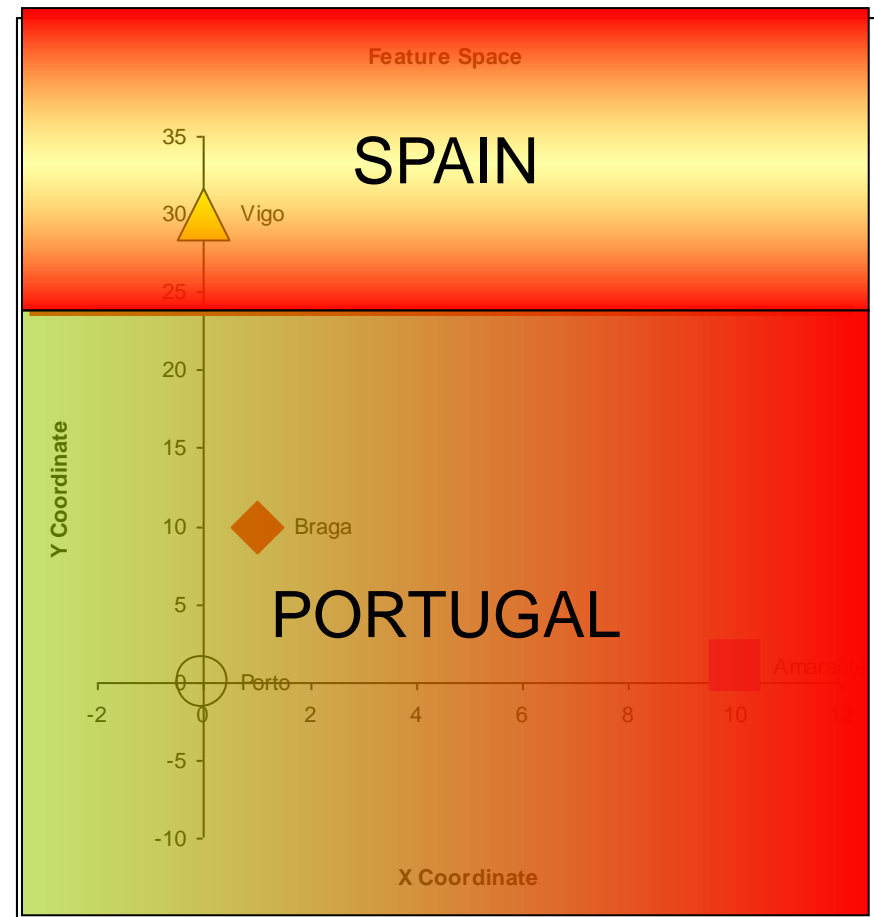
What if I don't have a model?

- I need to **learn** from observations.
 - Derive a model
 - Direct classification
- **Training stage**
 - Learn model parameters
- **Classification**



Classes

- In our example, cities can belong to:
 - Portugal
 - Spain
- I have two **classes** of cities
- A **class** represents a sub-space of my feature space



Classifiers

- A **Classifier C** maps a class into the feature space

$$C_{\text{Spain}}(x, y) = \begin{cases} \textit{true} & , y > K \\ \textit{false} & , \textit{otherwise} \end{cases}$$

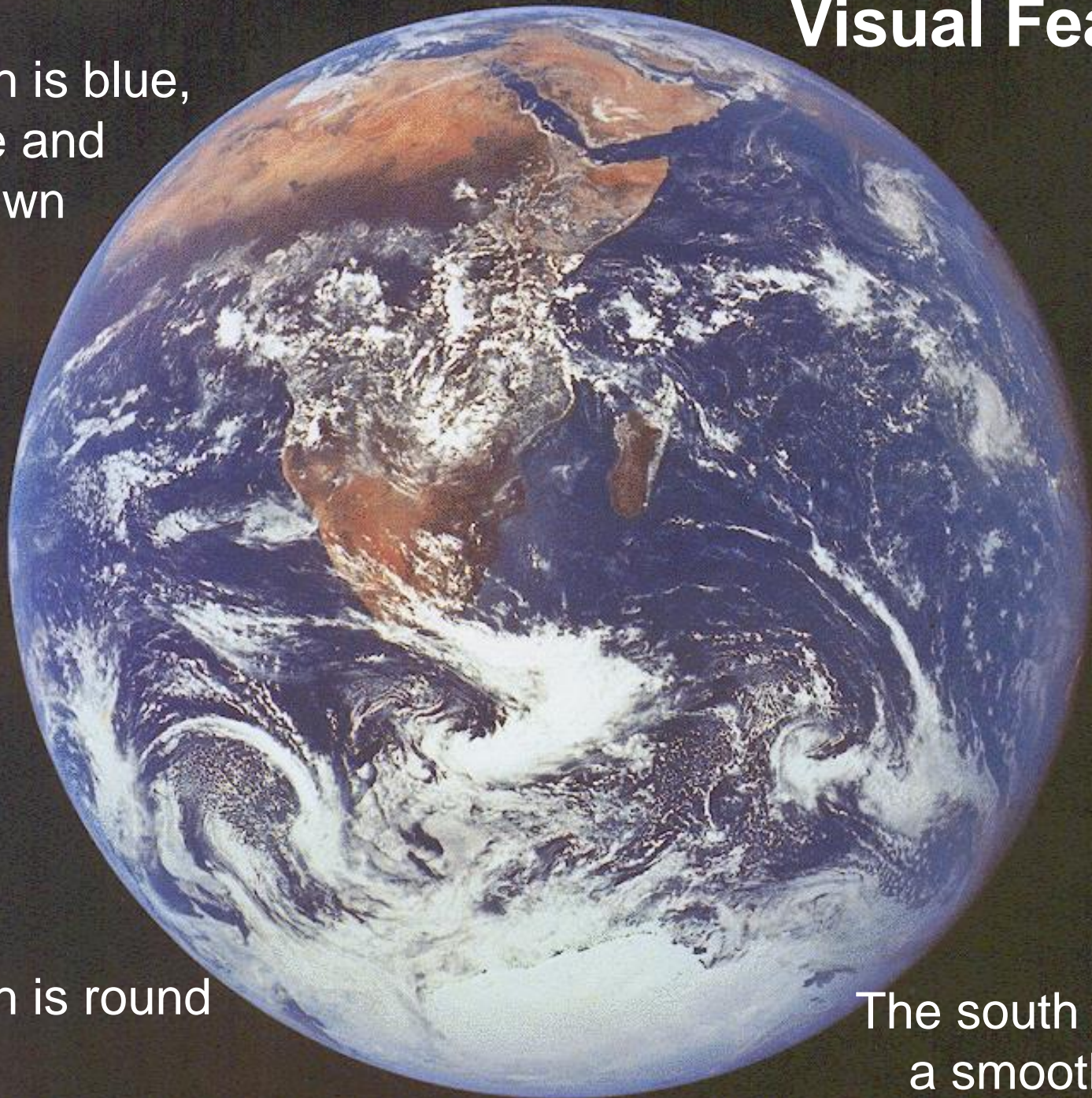
- Various types of classifiers
 - Nearest-Neighbours
 - Bayesian
 - Soft-computing machines
 - Etc...

Topic: Visual Features

- Introduction to Pattern Recognition
- Statistical Pattern Recognition
- **Visual Features**
- Detection of interest points
- Local invariant descriptors

Visual Features

The earth is blue,
white and
brown



The earth is round

The south pole has
a smooth texture

Visual Features

- **Features**
 - Measure specific characteristics
 - Numerical values
 - May have multiple values
- **Visual Features**
 - Quantify visual characteristics of an image
 - Popular features
 - Colour, Texture, Shape

Feature vector

- Feature F_i $F_i = [f_i]$

- Feature F_i with N values.

$$F_i = [f_{i1}, f_{i2}, \dots, f_{iN}]$$

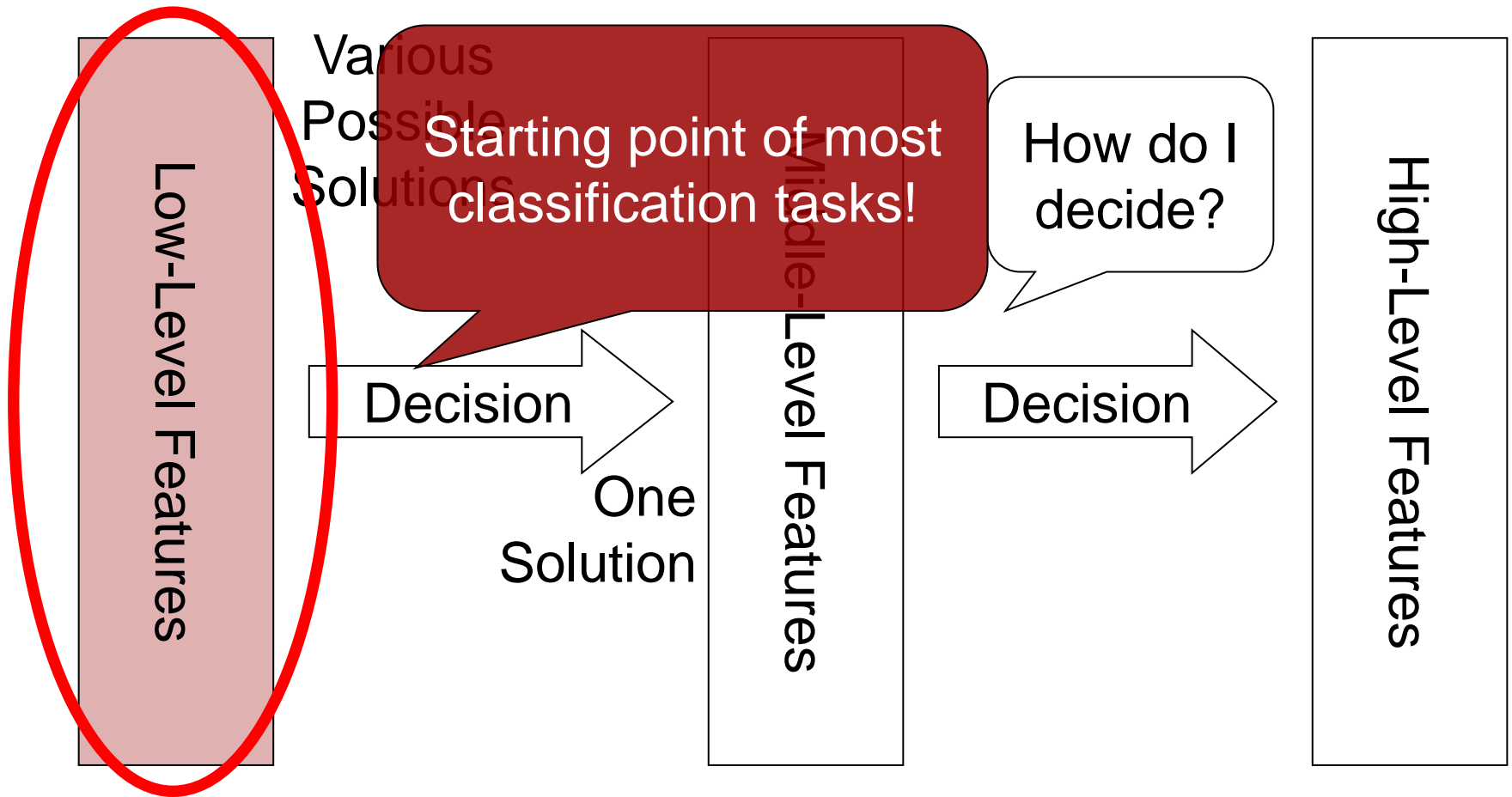
- Feature vector F with M features.

$$F = [F_1 | F_2 | \dots | F_M]$$

- Naming conventions for this module:

- Elements of a **feature vector** are called **coefficients**
- **Features** may have one or more **coefficients**
- **Feature vectors** may have one or more **features**

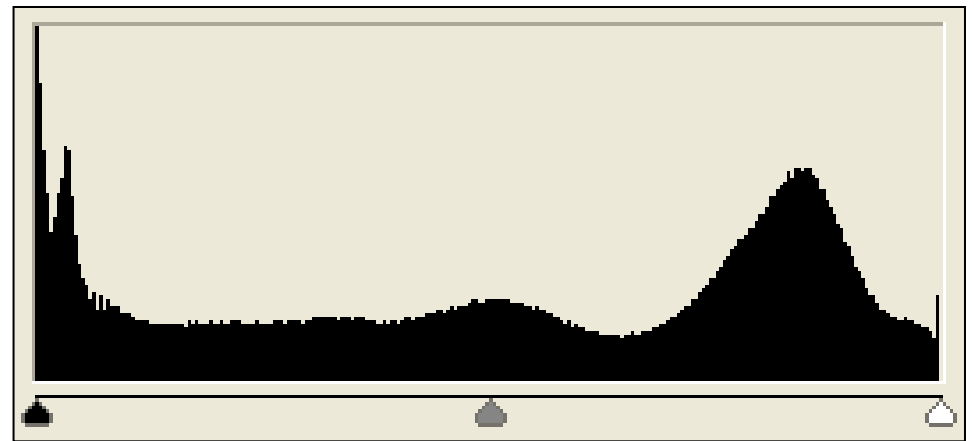
Features & Decisions



Gray-Level Histogram

- Intensity distribution (HSI)
- We can define the number of histogram bins
- Histogram bins = Feature coefficients

$$F = [f_0, \dots, f_{255}]$$



Colour Histogram

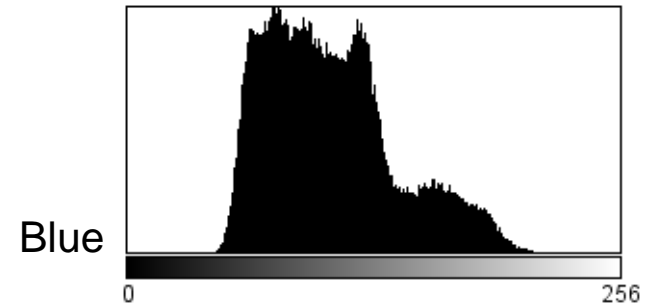
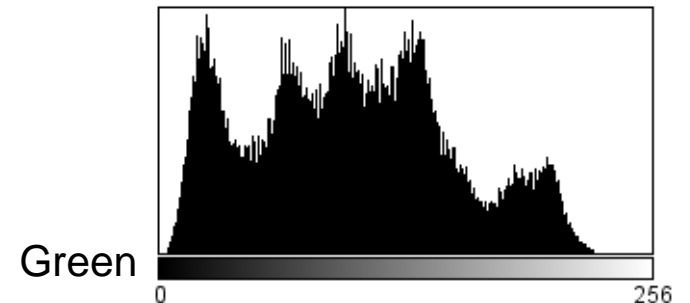
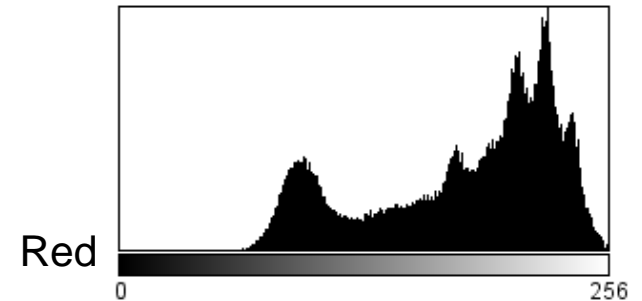
- We typically have three histograms

Ex: RGB Colour space

- Red Histogram
- Green Histogram
- Blue Histogram

- How do we build a feature vector?

- Concatenate vectors
- Multi-dimensional quantization of colour space



RGB Histogram

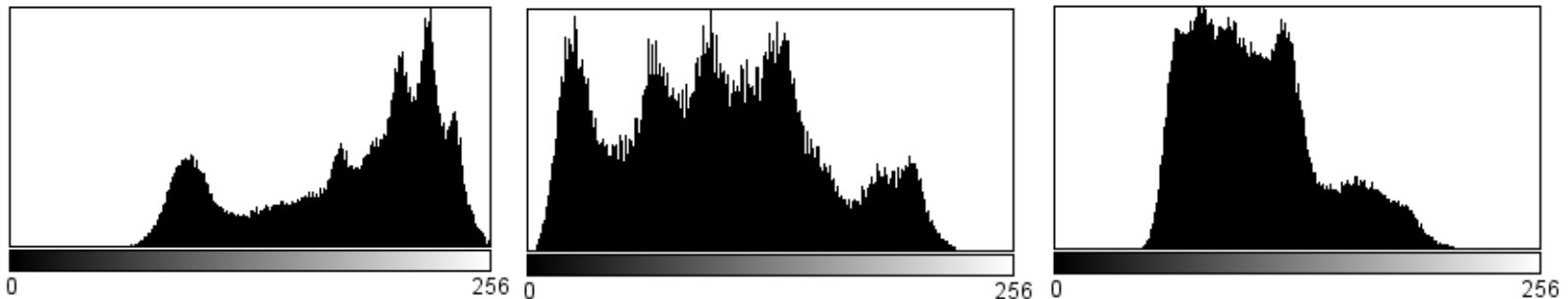
- Simply concatenate vectors
- Not very smart... (why?)

$$F_R = [f_{R0}, \dots, f_{R255}]$$

$$F_G = [f_{G0}, \dots, f_{G255}]$$

$$F_B = [f_{B0}, \dots, f_{B255}]$$

$$F_{RGB} = [F_R \mid F_G \mid F_B]$$

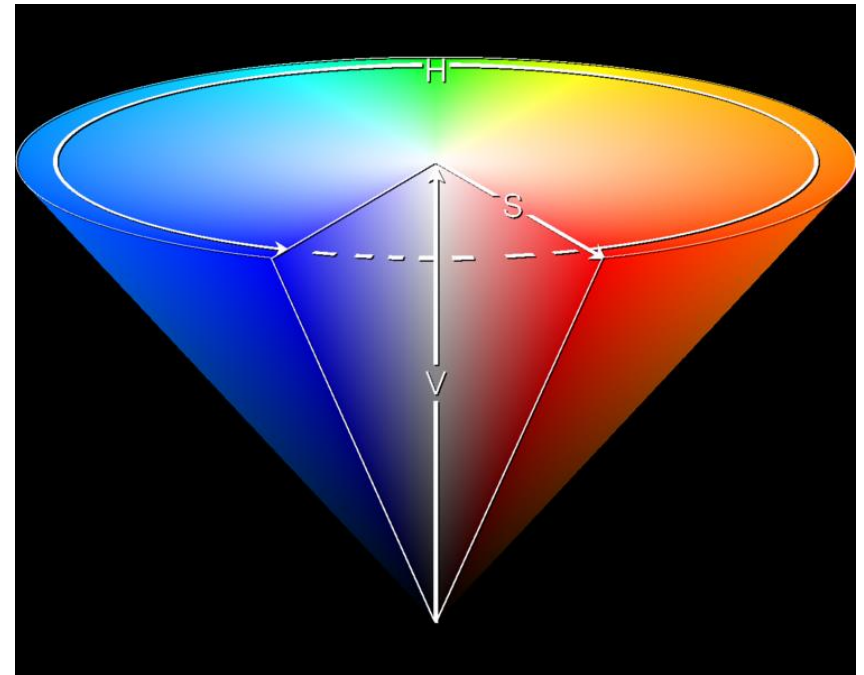


HSI Histogram

- Quantize HSI space
 - Define number of bins N .
 - Feature vector

$$F_{HSI} = [f_0, \dots, f_N]$$

- Typically better for object description



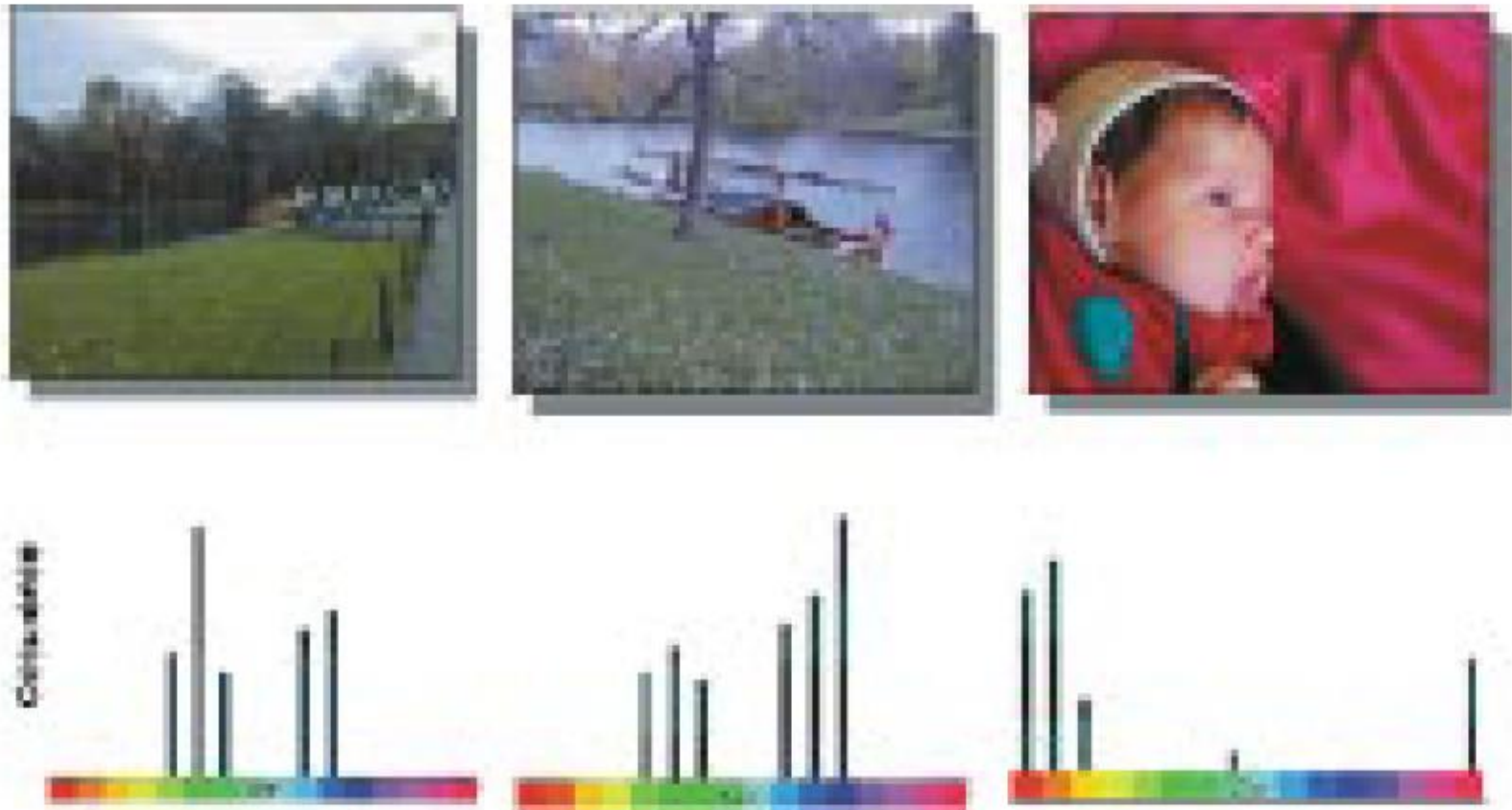
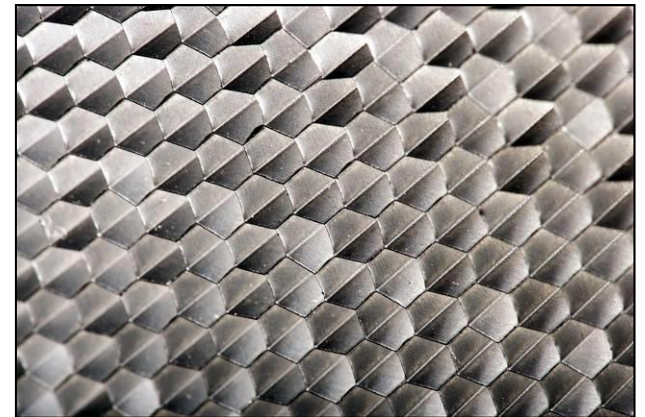
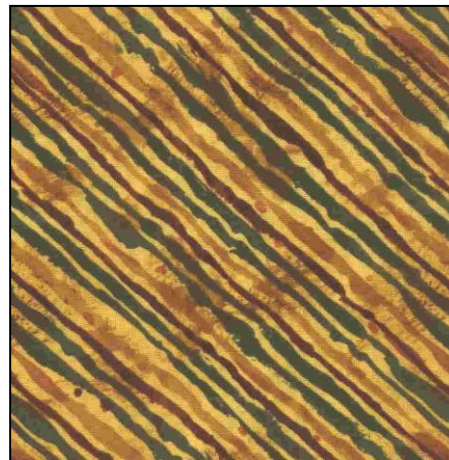


Fig. 2. Three color images and their MPEG-7 histogram color distribution, depicted using a simplified color histogram. Based on the color distribution, the two left images would be recognized as more similar compared to the one on the right.

What is texture?

“Texture gives us information about the spatial arrangement of the colours or intensities in an image”

[L. Shapiro]



Two approaches to texture

- **Structural approach**
 - Texture is a set of primitive *texels* in some regular or repeated relationship
 - Good for regular, ‘man-made’ textures
- **Statistical approach**
 - Texture is a quantitative measure of the arrangement of intensities in a region
 - More general and easier to compute

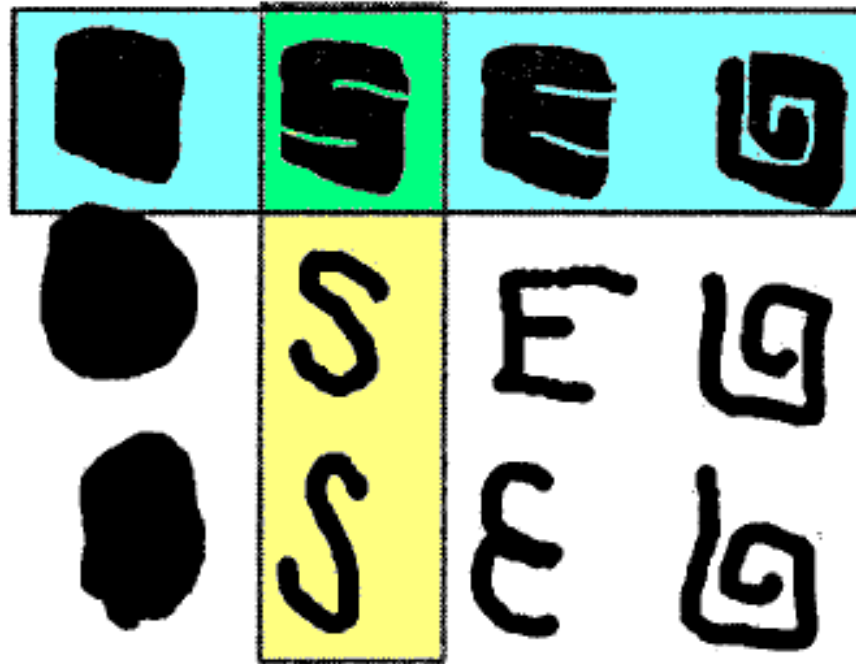
Statistical approaches

- Grey level of central pixels
- Average of grey levels in window
- Median
- Standard deviation of grey levels
- Difference of maximum and minimum grey levels
- Difference between average grey level in small and large windows
- Sobel feature
- Kirsch feature
- Derivative in x window
- Derivative in y window
- Diagonal derivatives
- Combine features



How do I pick one??

Shape Descriptors



- Blue: Similar shapes by Region-Based
- Yellow: Similar shapes by Contour-Based

Topic: Detection of interest points

- Introduction to Pattern Recognition
- Statistical Pattern Recognition
- Visual Features
- **Detection of interest points**
- Local invariant descriptors

Motivation: Same interest points

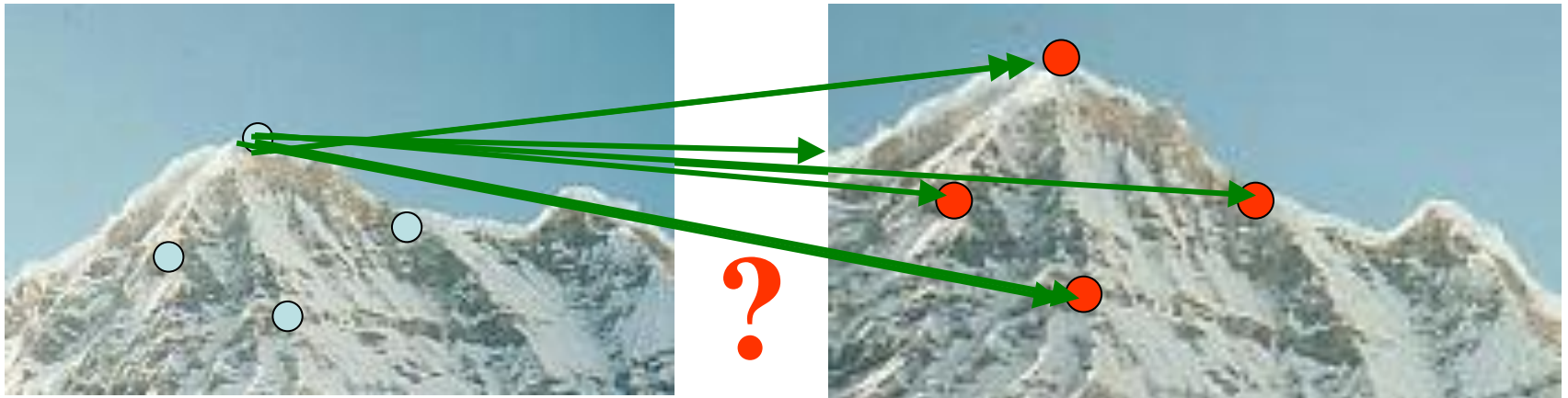
- We want to detect the same points in both images



No chance to find true matches!

Motivation: 'Unique' descriptor per interest point

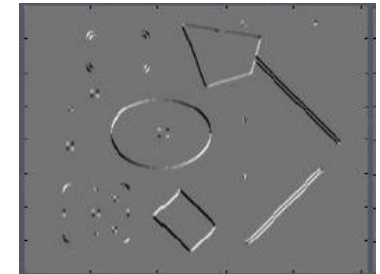
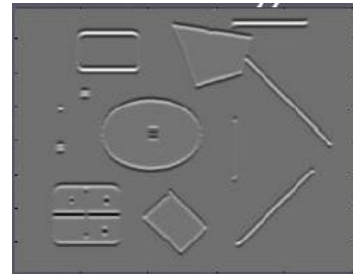
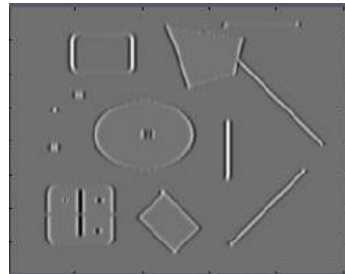
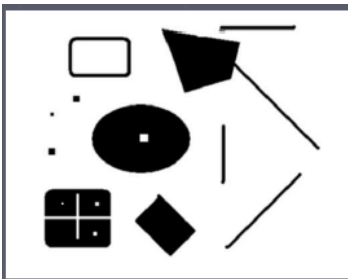
- We want to match the same interest points
- Need a descriptor invariant to geometric and photometric differences



Corners are distinctive interest points

$$M = \sum w(x, y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix}$$

2 x 2 matrix of image derivatives (averaged in neighborhood of a point)



Notation:

$$I_x \Leftrightarrow \frac{\partial I}{\partial x}$$

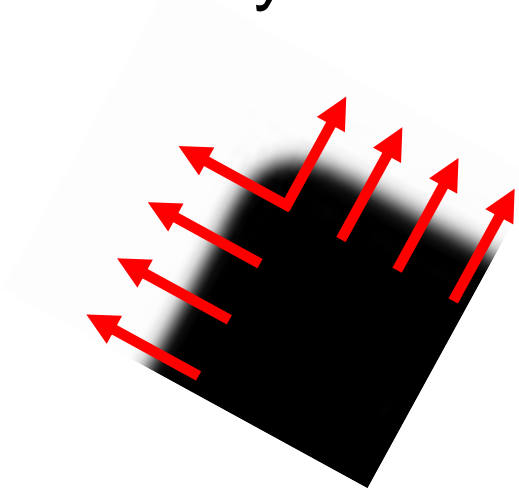
$$I_y \Leftrightarrow \frac{\partial I}{\partial y}$$

$$I_x I_y \Leftrightarrow \frac{\partial I}{\partial x} \frac{\partial I}{\partial y}$$

Gradient strength

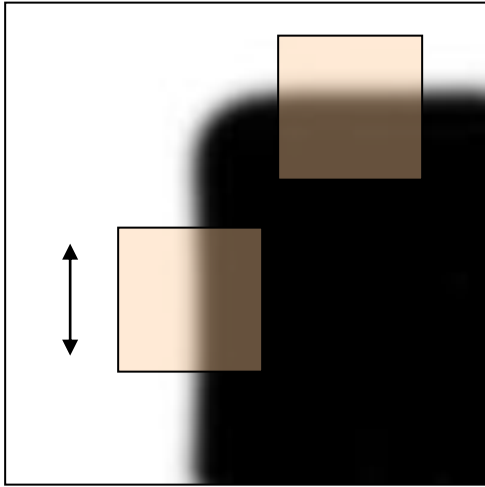
Since M is symmetric, we have $M = X \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} X^T$

$$Mx_i = \lambda_i x_i$$



The *eigenvalues* of M reveal the amount of intensity change in the two principal orthogonal gradient directions in the window

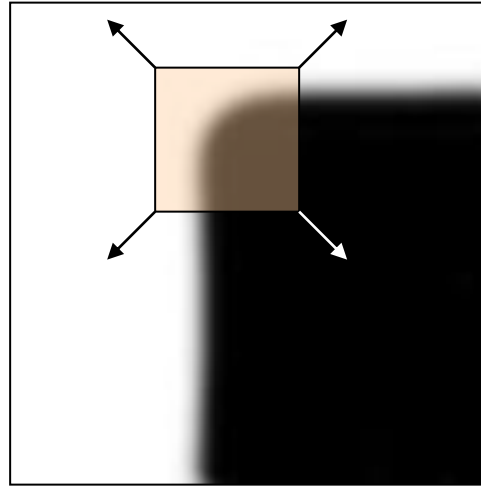
Scoring 'cornerness'



“edge”:

$$\lambda_1 \gg \lambda_2$$

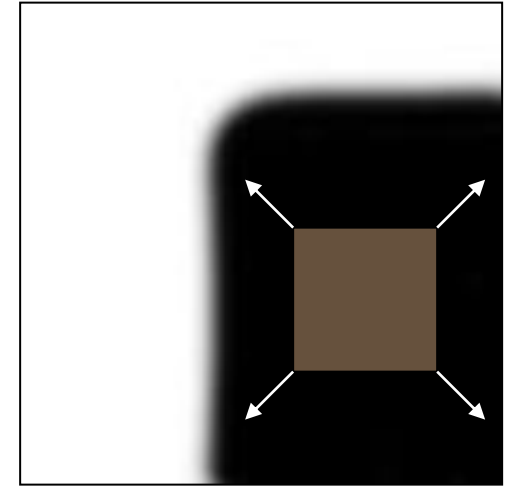
$$\lambda_2 \gg \lambda_1$$



“corner”:

λ_1 and λ_2 are large,

$$\lambda_1 \sim \lambda_2;$$



“flat” region

λ_1 and λ_2 are small;

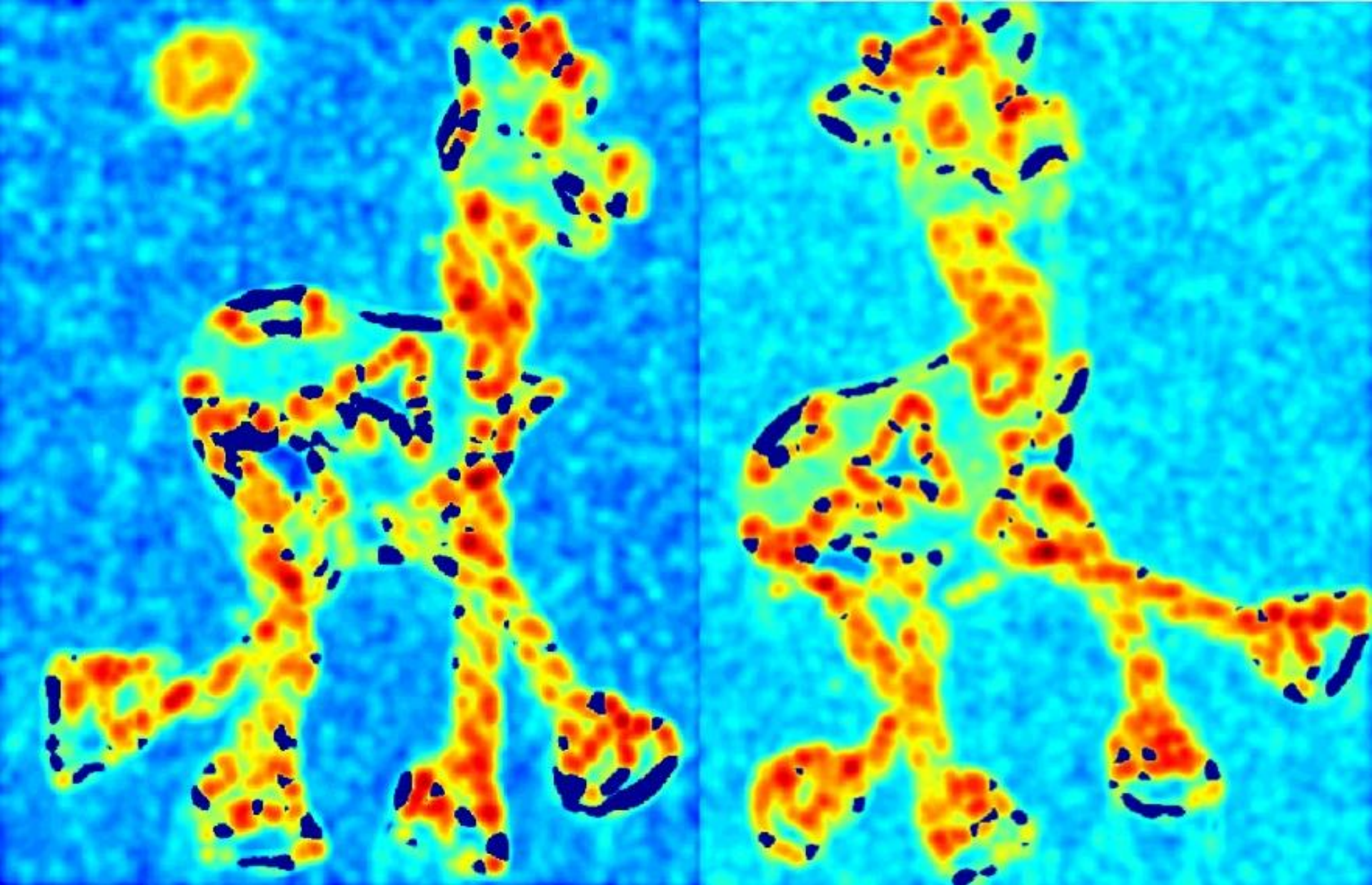
One way to score the cornerness:

$$f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}$$

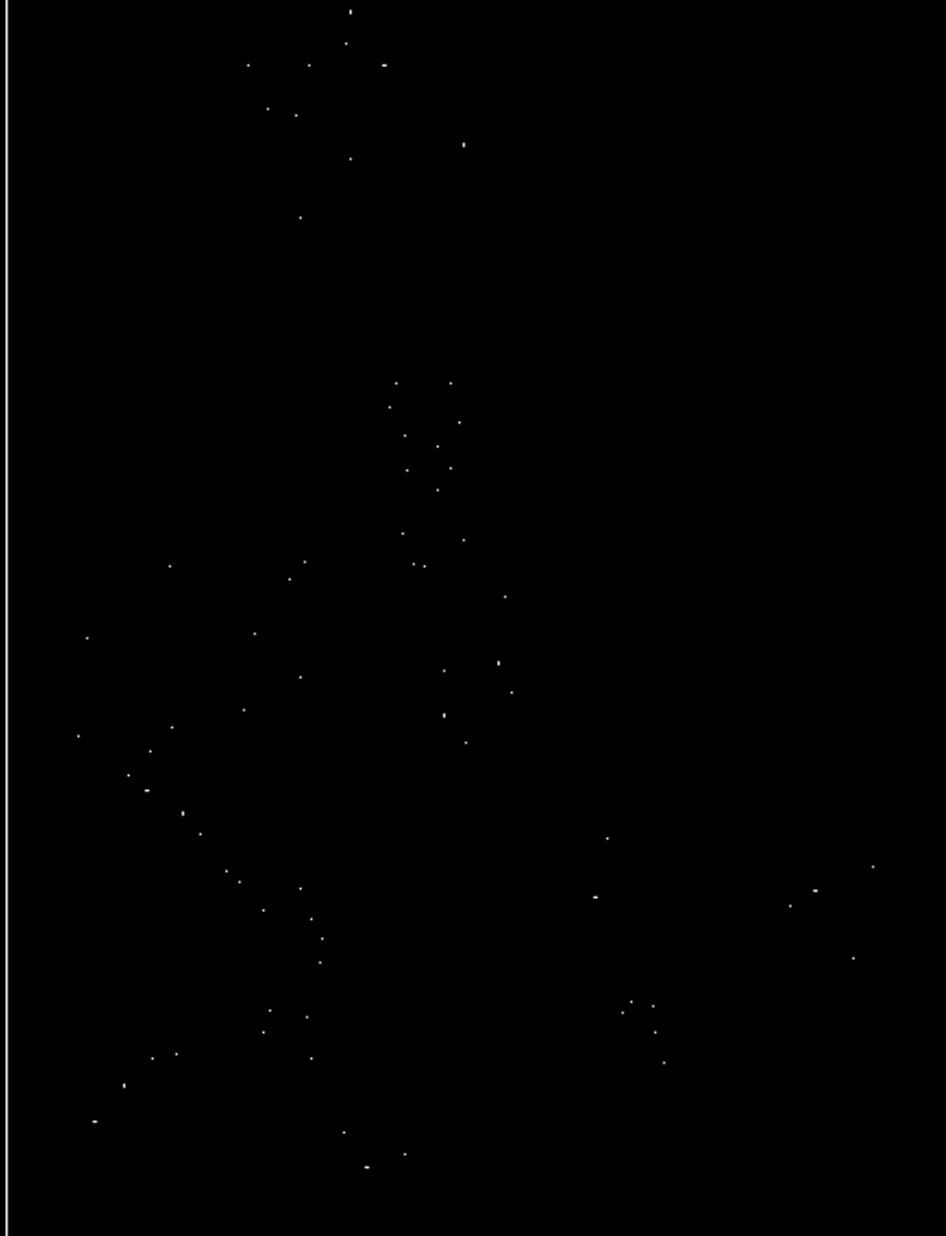
Harris corner detector

- 1) Compute M matrix for image window surrounding each pixel to get its *cornerness* score.
- 2) Find points with large corner response ($f >$ threshold)
- 3) Take the points of local maxima, i.e., perform non-maximum suppression











Properties of the Harris corner detector

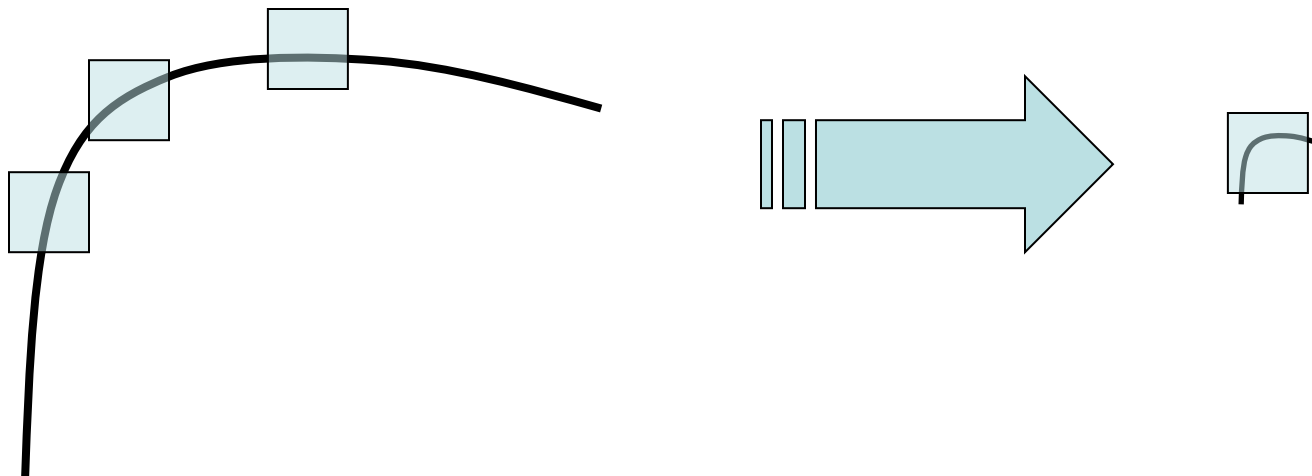
- Rotation invariant? Yes

$$M = X \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} X^T$$

- Scale invariant?

Properties of the Harris corner detector

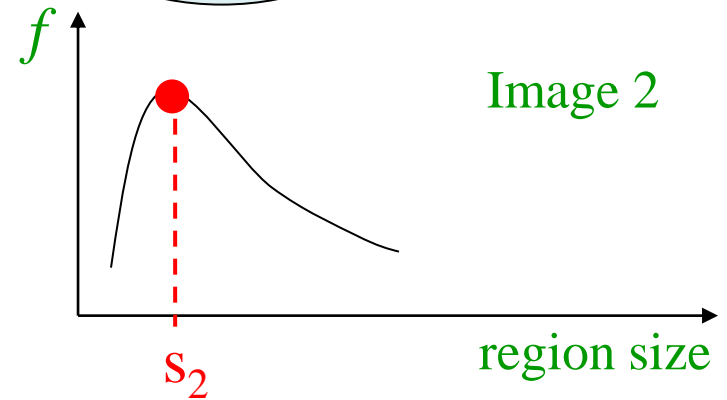
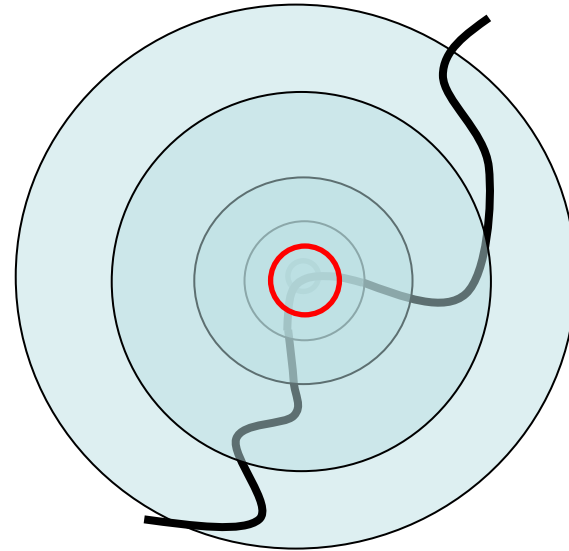
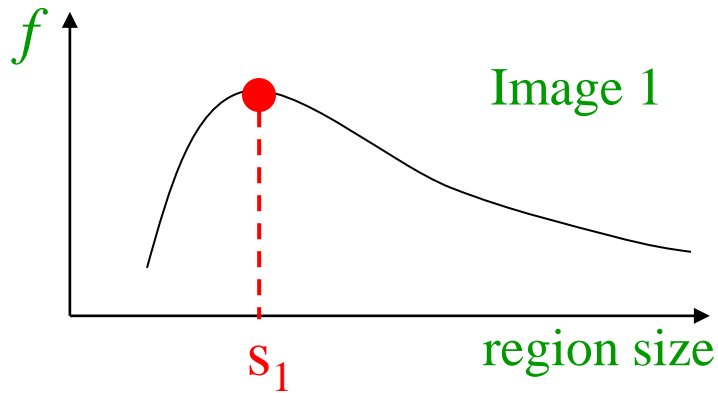
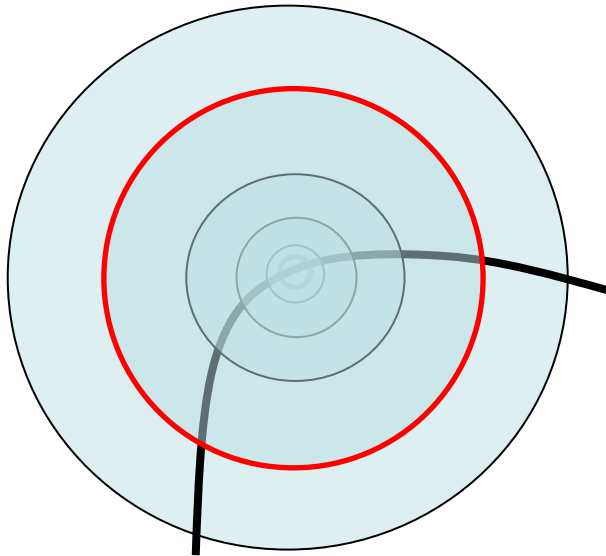
- Rotation invariant? Yes
- Scale invariant? No



All points will be classified as **edges**

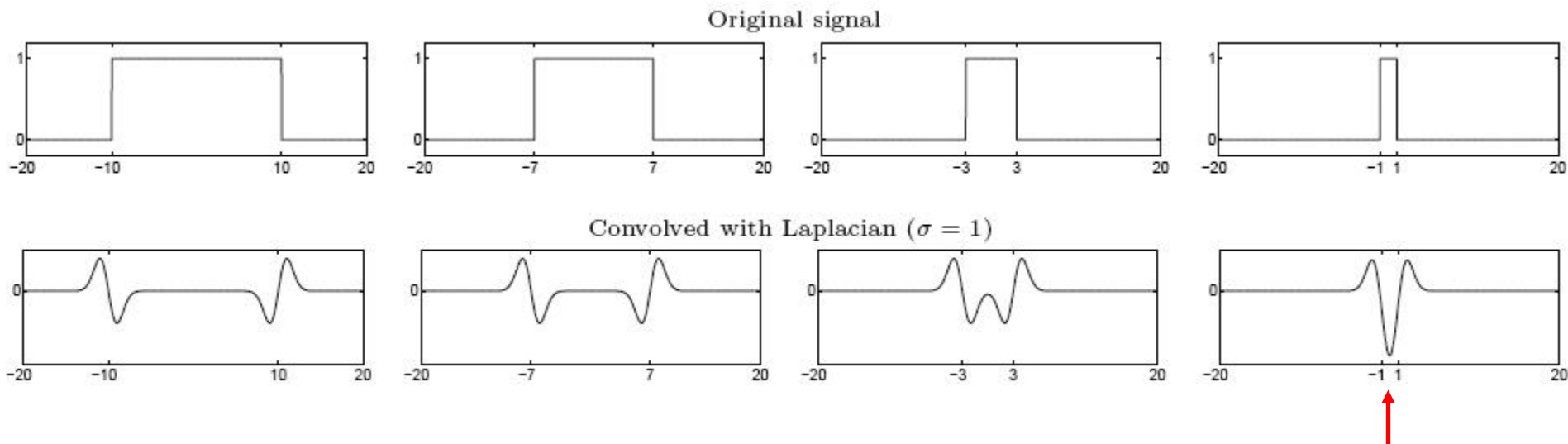
Corner !

Automatic scale selection



From edges to blobs

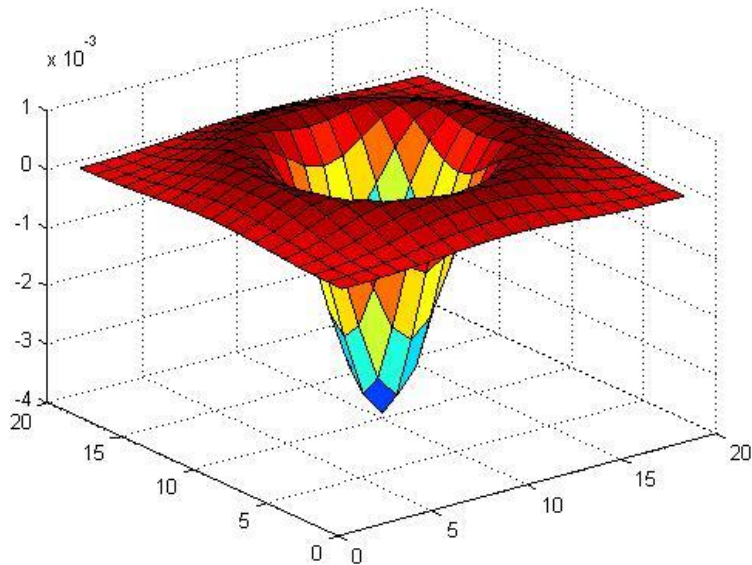
- Edge = ripple
- Blob = superposition of two ripples



- Spatial selection: the magnitude of the Laplacian response will achieve a maximum at the center of the blob, provided the scale of the Laplacian is “matched” to the scale of the blob

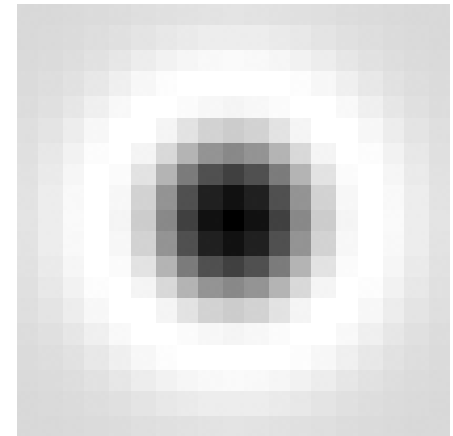
maximum

Blob detection in 2D



- Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D

$$\nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2}$$

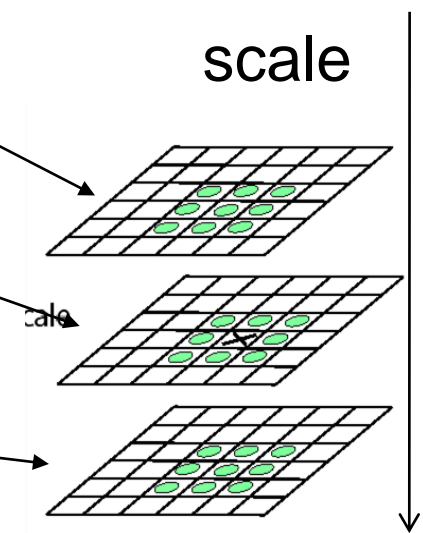
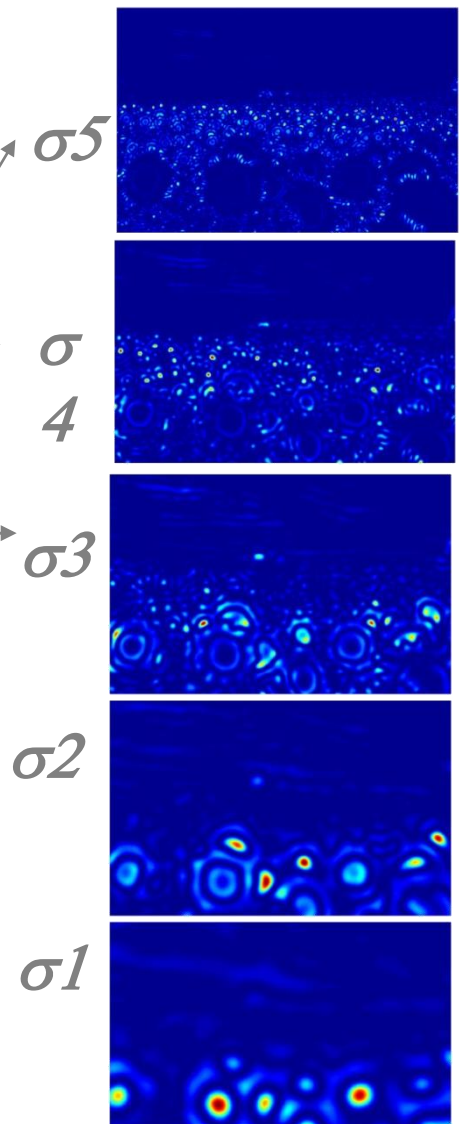


Scale invariant interest points

Interest points are local maxima in both position and scale



$$L_{xx}(\sigma) + L_{yy}(\sigma)$$



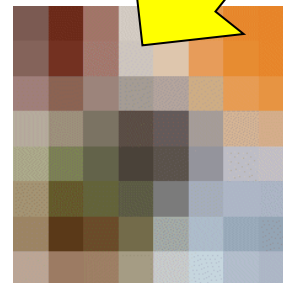
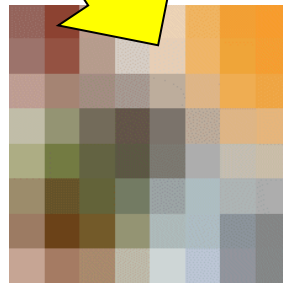
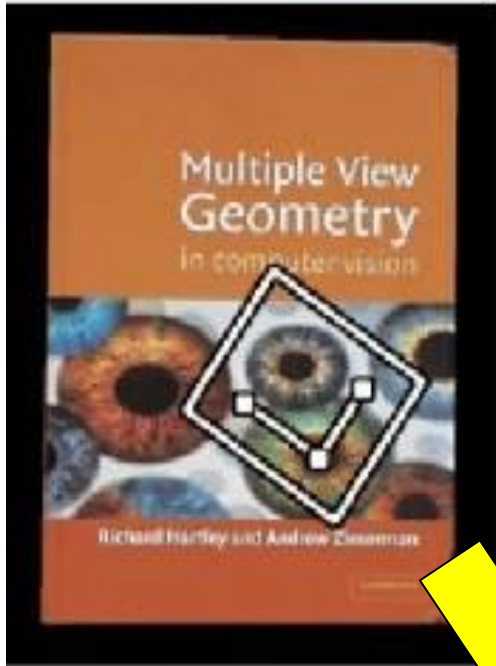
⇒ List of (x, y, σ)

Squared filter response maps

Topic: Local invariant descriptors

- Introduction to Pattern Recognition
- Statistical Pattern Recognition
- Visual Features
- Detection of interest points
- **Local invariant descriptors**

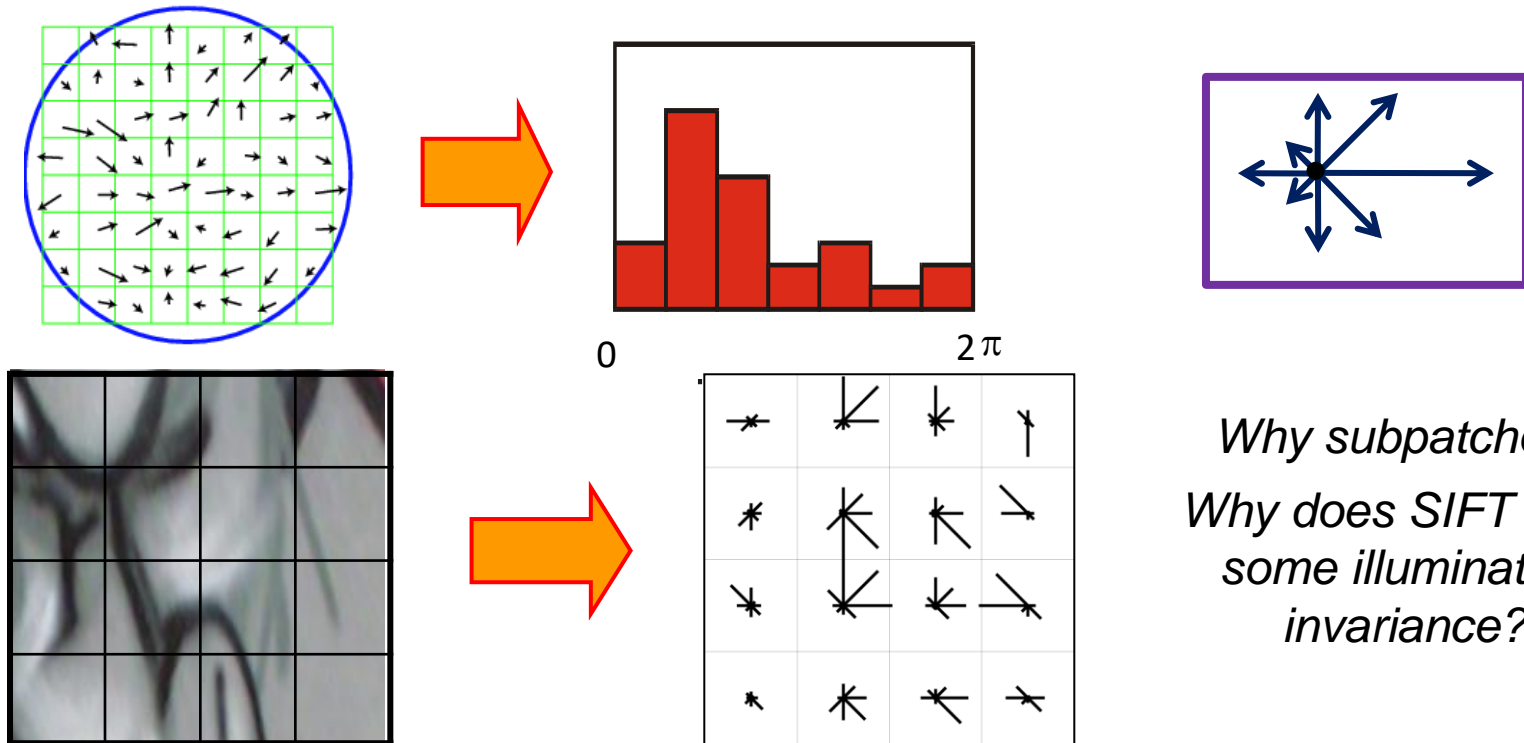
Geometric transformations



e.g. scale,
translation,
rotation

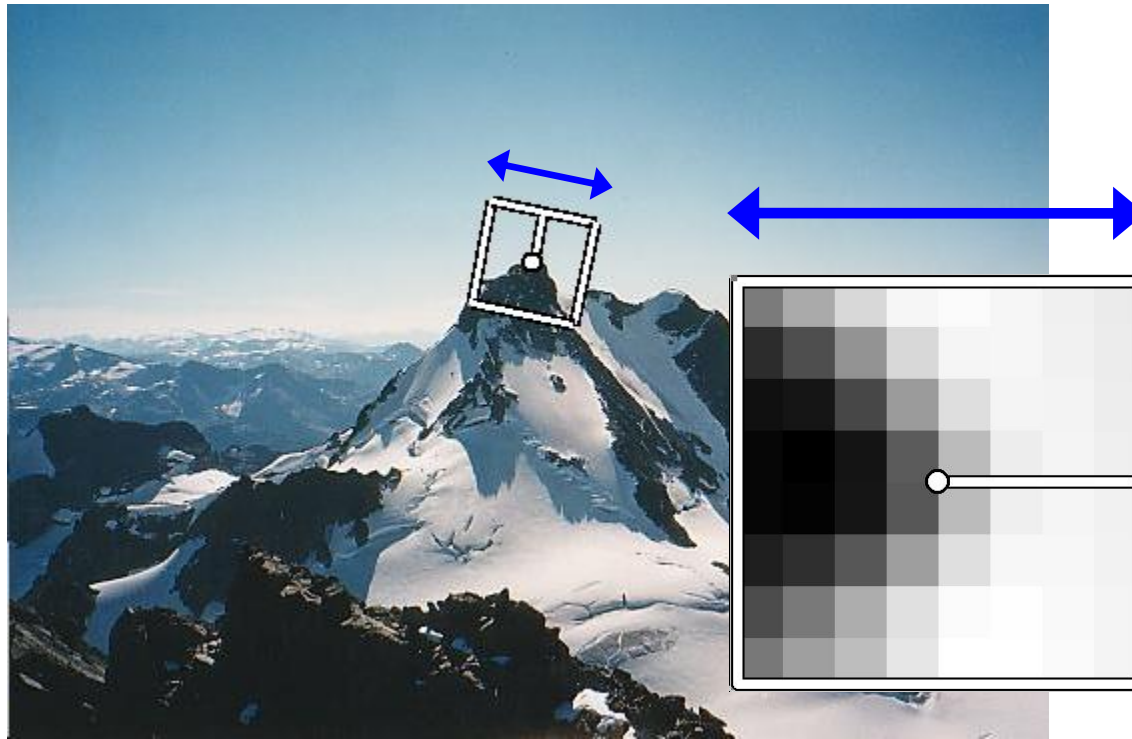
SIFT descriptor [Lowe 2004]

- Use histograms to bin pixels within sub-patches according to their orientation



*Why subpatches?
Why does SIFT have
some illumination
invariance?*

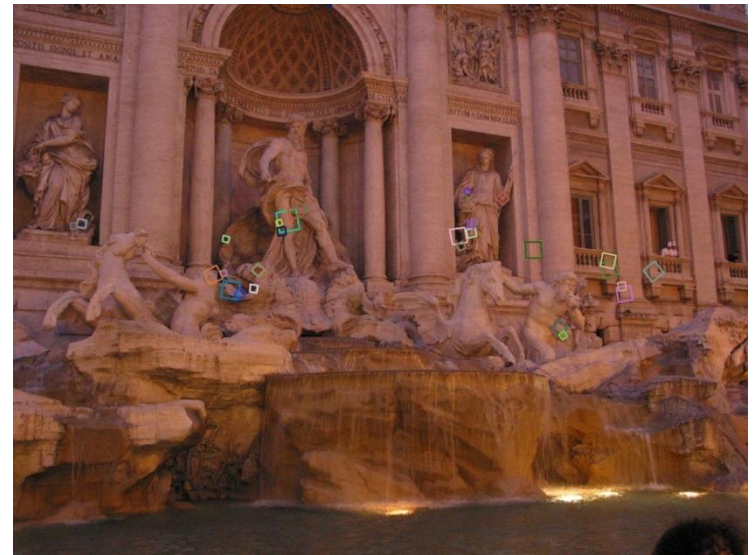
Making descriptor rotation invariant



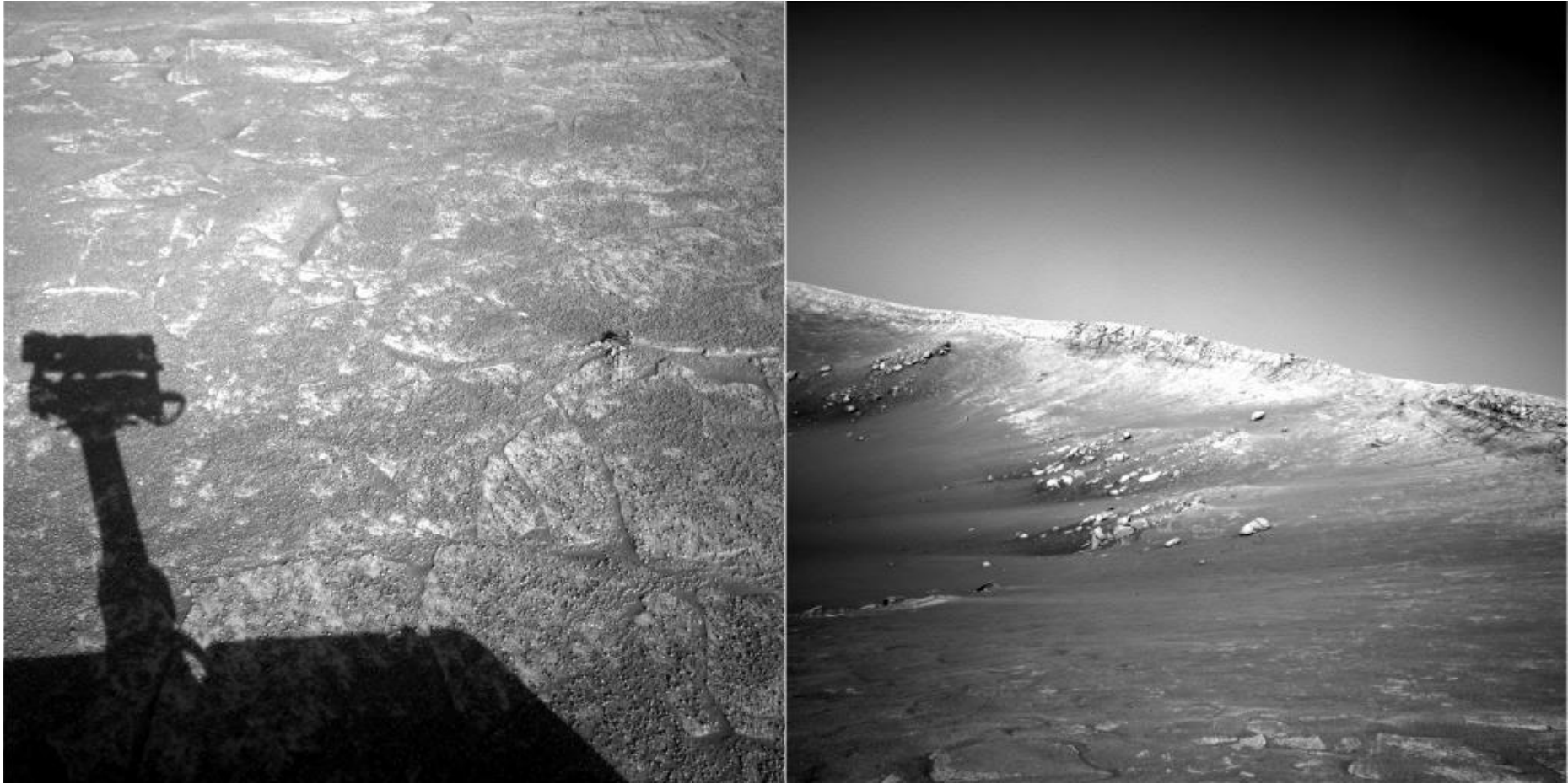
- Rotate patch according to its dominant gradient orientation
- This puts the patches into a canonical orientation

SIFT descriptor [Lowe 2004]

- **Extraordinarily robust matching technique**
 - Can handle changes in viewpoint
 - Can handle significant changes in illumination
 - Fast and efficient—can run in real time
 - Lots of code available

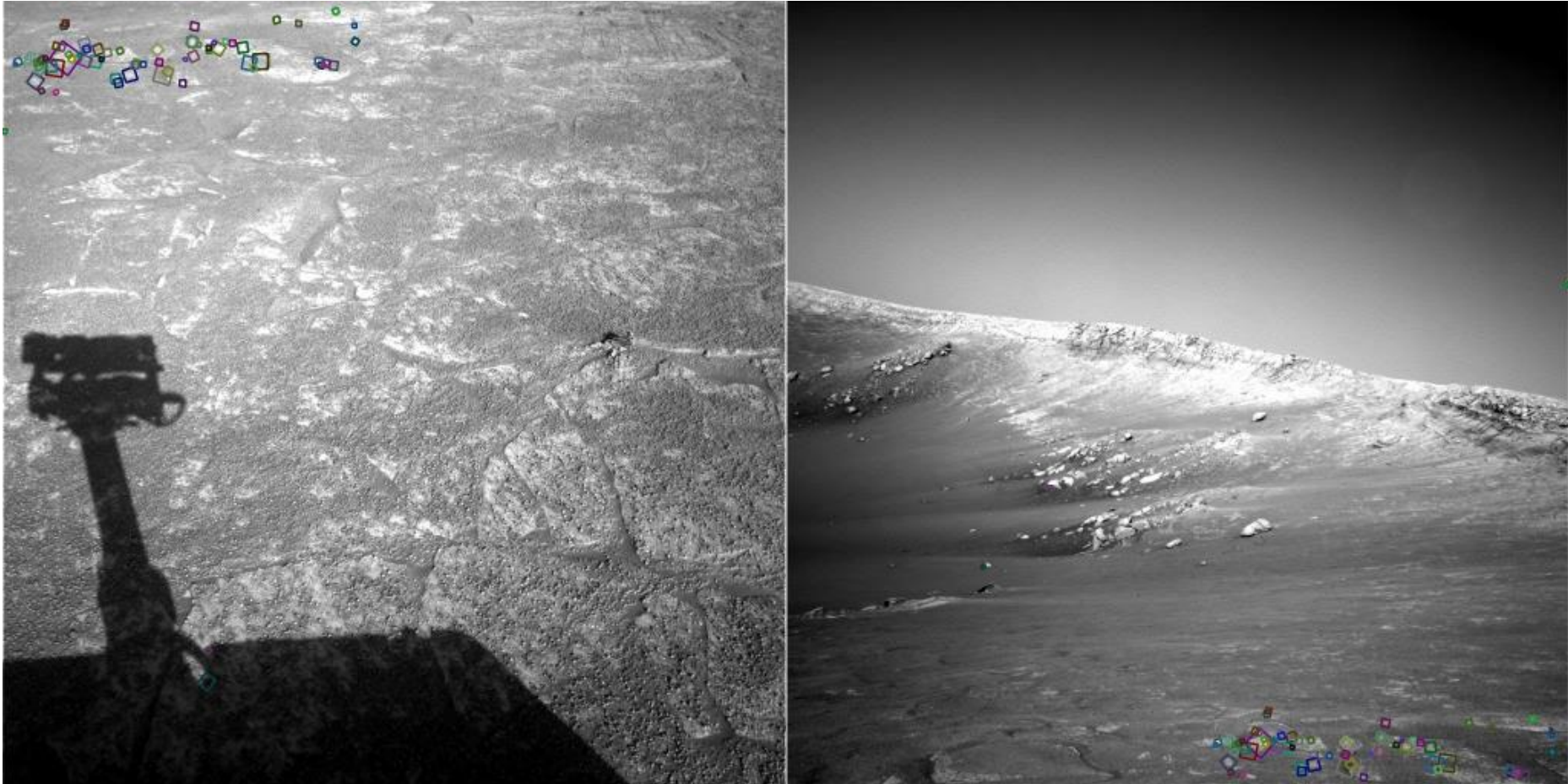


Example



NASA Mars Rover images

Example



NASA Mars Rover images

SIFT properties

- Invariant to
 - Scale
 - Rotation
- Partially invariant to
 - Illumination changes
 - Camera viewpoint
 - Occlusion, clutter

Resources

- Szeliski, “Computer Vision: Algorithms and Applications”, Springer, 2011
 - Chapter 14 – “Recognition”
 - Chapter 4 – “Feature Detection and Matching”