UNIVERSITÀ DEGLI STUDI DI PADOVA

# High-level vision: object recognition, template matching, bag of words

Stefano Ghidoni

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

INTELLIGENT AUTONOMOUS SYSTEMS LAB

# Agenda



- Finding object in an image
- Template matching
- Histogram of oriented gradients
- Bag of words

# Getting info about objects



- Consider the high-level task of *getting some information from an image about objects*
- **Object recognition** is a general term to describe a collection of related computer vision tasks that involve identifying objects in images or videos

# Getting info about objects

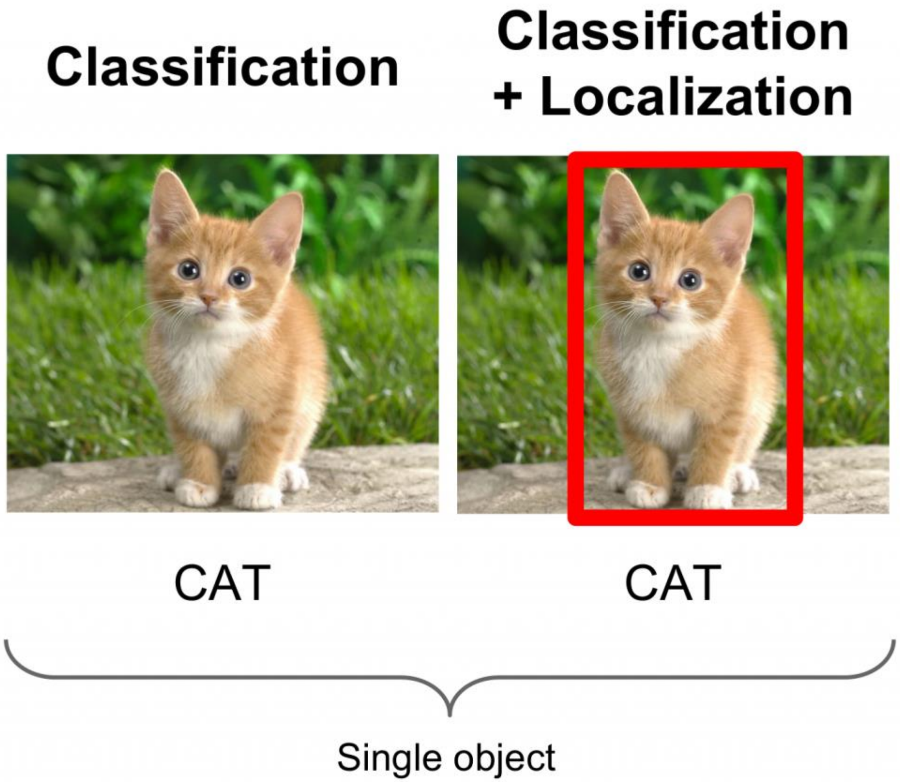
**Classification**

CAT

Output:
A label

# Getting info about objects

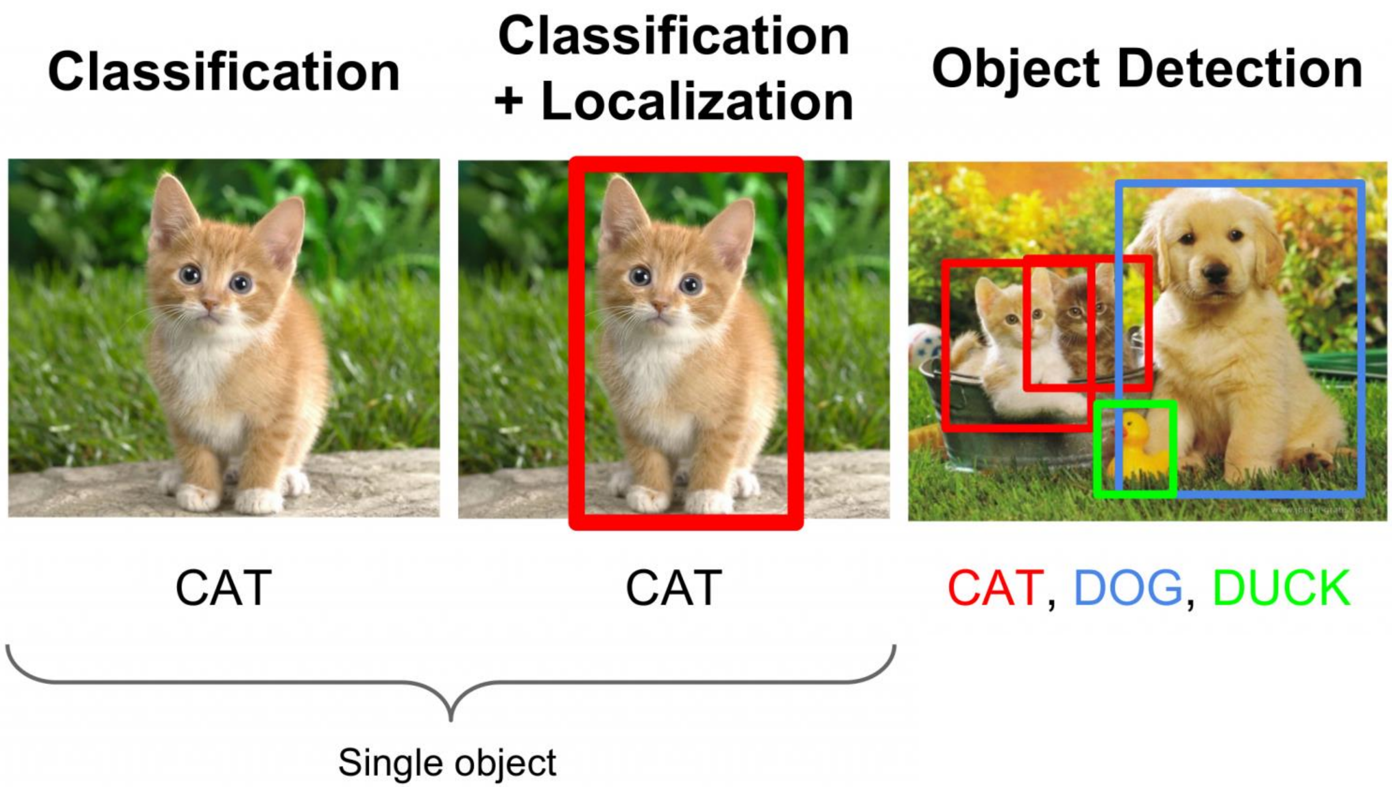





Output:
A label + a bounding box

# Getting info about objects

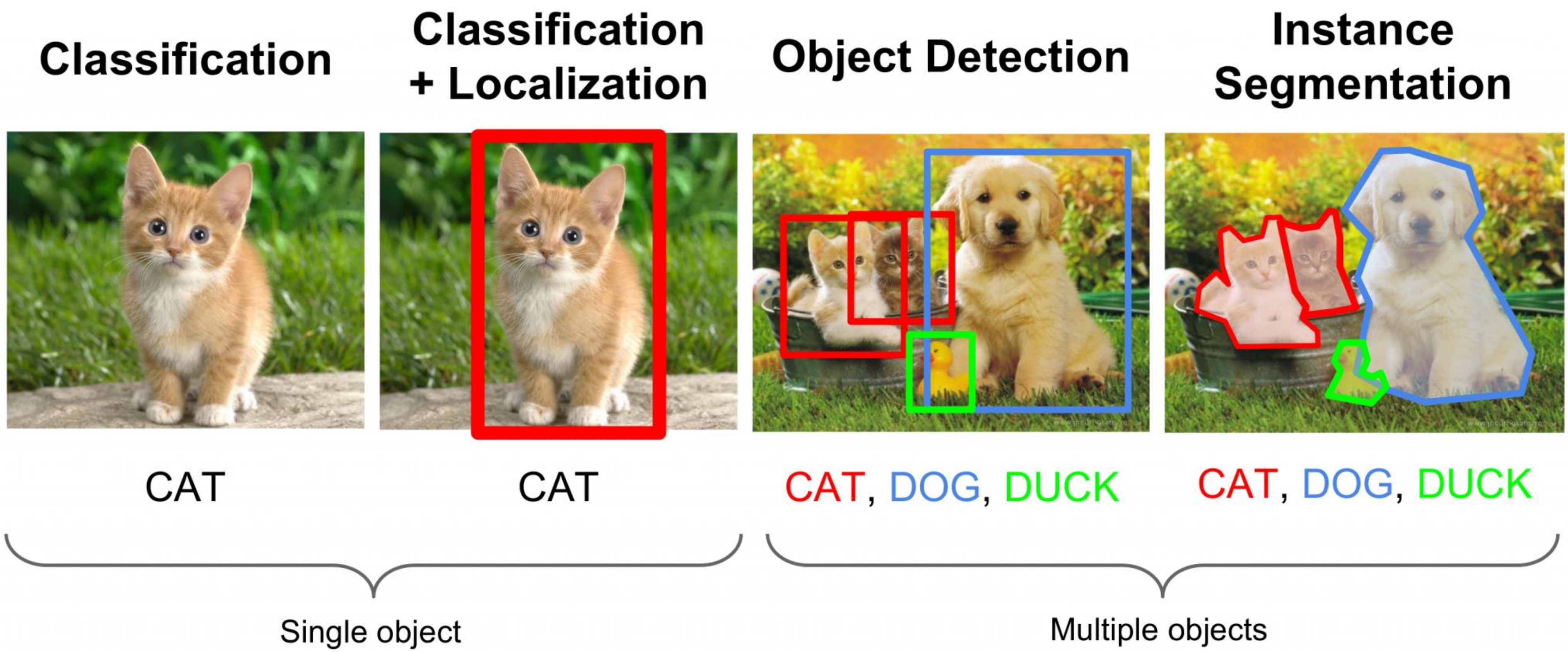





Output:
Multiple bounding boxes with label

# Getting info about objects






Output:
Multiple areas with label

https://www.augmentedstartups.com/blog/mastering-image-classification-techniques-enhancing-accuracy-and-efficiency

# Modeling variability



- The tasks discussed so far shall cope with
  - Different camera positions
  - Perspective deformations
  - Illumination changes
  - Intra-class variations

# Camera position



# Perspective deformation



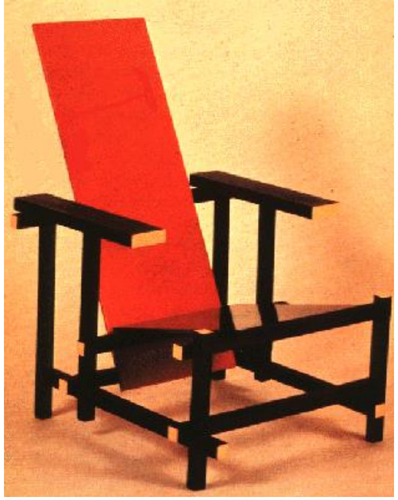

# Illumination changes





# Intra-class variations





# Approaches to object recognition



- We already covered a method for object detection
  - Which one?

- Anti spoiler ☺

# Approaches to object detection



- We already covered a method for object detection
  - Which one?
- Boosting for face detection (Viola and Jones)
  - Can be applied to other targets

# Approaches to object detection

- We already covered a method for object detection
  – Which one?
- Other approaches are available
  – Template matching
  – Histogram of Oriented Gradients (HOG)
  – Bag of Words
  – Machine learning / deep learning

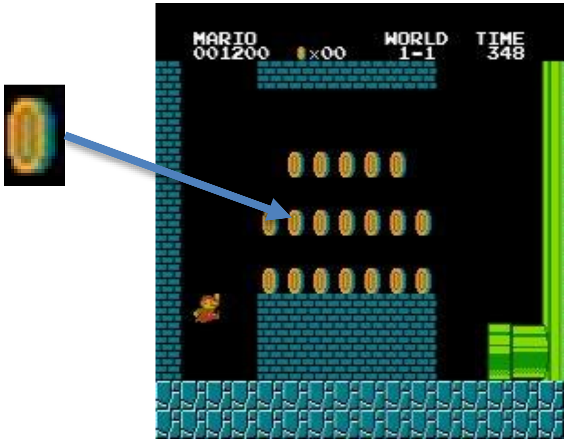
This lecture

# Template matching

# Templates



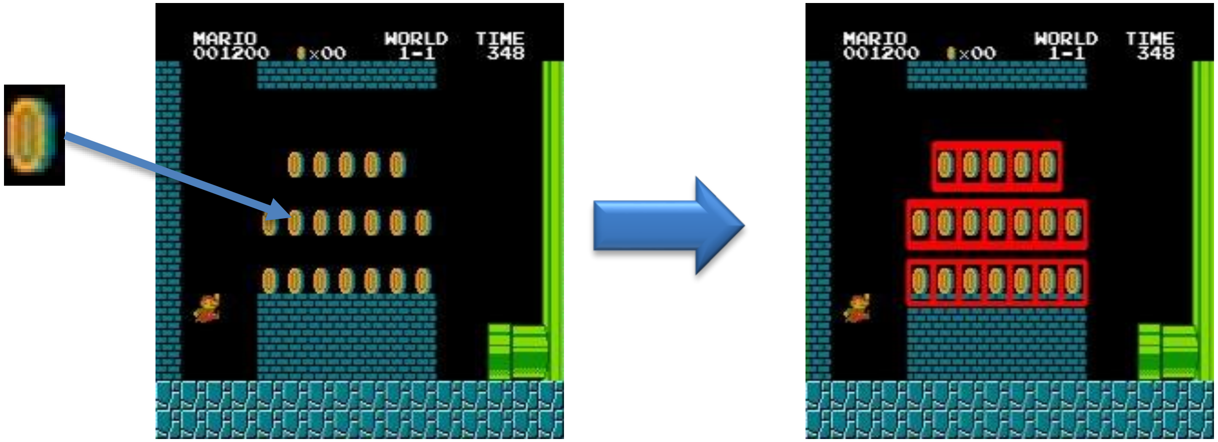
- A template is
  - Something fashioned, shaped or designed to serve as a model
  - Something formed after a model
  - A representative instance (an example)



# Template matching



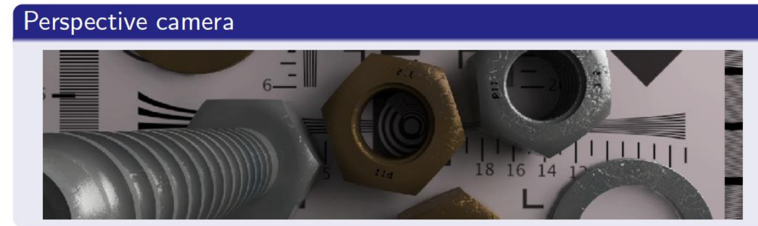
- "Where is a given object?"
- Find instances of the templates in the image
- A similarity measure should be chosen



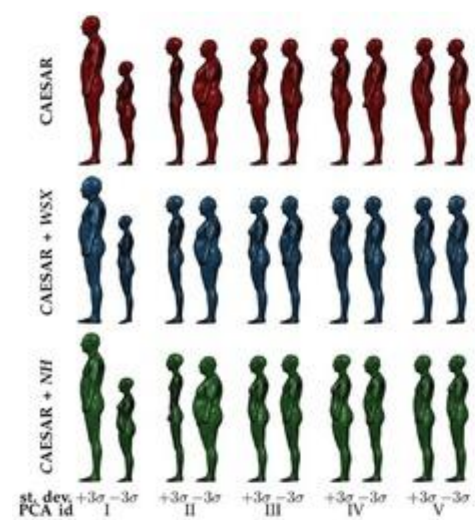
# Challenges



- Template variability, deformable objects
- Imaging device properties
- Viewpoint changes
- Affine transforms – scale, rotations, translations, …
- Noise
- Illumination







# How to apply a template?



- Given
  - An image
  - A template
- How can we evaluate the match?

- Anti spoiler

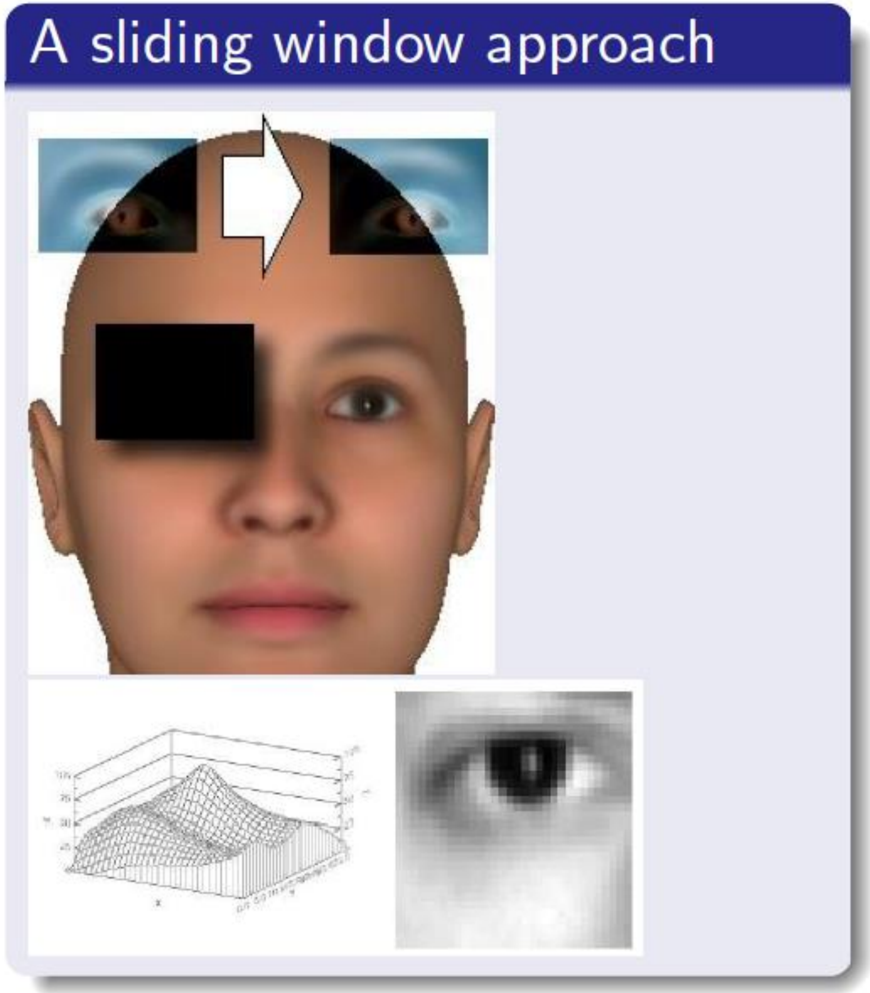
# Matching the template

- Common option: correlation-based approach
- Template T: rigid object, often a small image
- Sliding window
- Comparison of
  - Pixel values
  - Features
  - Edges or gradient orientation
- Similarity metrics: SSD, SAD, ZNCC

# Sliding window



- Template-based approaches are often based on a sliding window
- The template is placed in every possible position across the image
  – Basic approach w/out rotation and scaling



# Correlation-based



- Common option: correlation-based approach
- Template T: rigid object, often a small image
- Sliding window
- Comparison of
  - Pixel values
  - Features
  - Edges or gradient orientation
- Similarity metrics: SSD, SAD, ZNCC

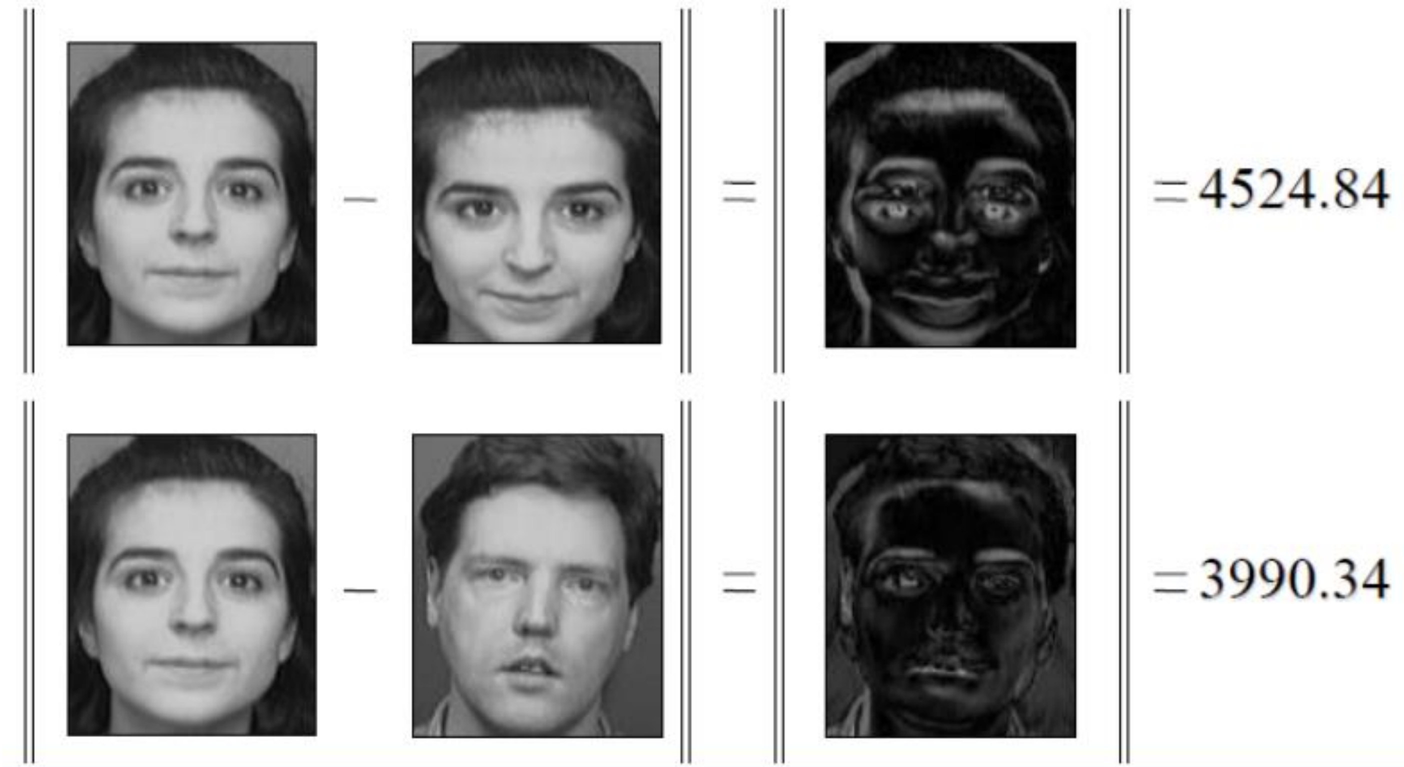
# Correlation-based

- Common option: correlation-based approach
- Template T: rigid object, often a small image
- Sliding window
- Comparison of
  - Pixel values
  - Features
  - Edges or gradient orientation
- Similarity metrics: SSD, SAD, ZNCC

# Challenges



- Simple differencing does not always provide reliable results – why in this case?



# Correlation-based

- Common option: correlation-based approach
- Template T: rigid object, often a small image
- Sliding window
- Comparison of
  - Pixel values
  - Features
  - Edges or gradient orientation
- Similarity metrics: SSD, SAD, ZNCC

# Correlation-based



- Common option: correlation-based approach
- Template T: rigid object, often a small image
- Sliding window
- Comparison of
  - Pixel values
  - Features
  - Edges or gradient orientation
- Similarity metrics: SSD, SAD, ZNCC

# Similarity metrics



- Sum of Squared Differences (SSD)

$$\phi(x,y) = \sum_{u,v \in T} \left(I(x+u, y+v) - T(u,v)\right)^2$$

- Sum of Absolute Differences (SAD)

$$\phi(x,y) = \sum_{u,v \in T} \left|I(x+u, y+v) - T(u,v)\right|$$

- Zero-mean Normalized Cross-Correlation (ZNCC)

$$\phi(x,y) = \frac{\sum_{u,v \in T}\left(I(x+u, y+v) - \bar{I}(x,y)\right)\left(T(u,v) - \bar{T}\right)}{\sigma_I(x,y)\sigma_T}$$

- $\bar{I}(x,y)$: average on window, $\bar{T}$: template average, $\sigma_I(x,y), \sigma_T$: standard deviation on image window and on template

# TM weak points and how to cope with

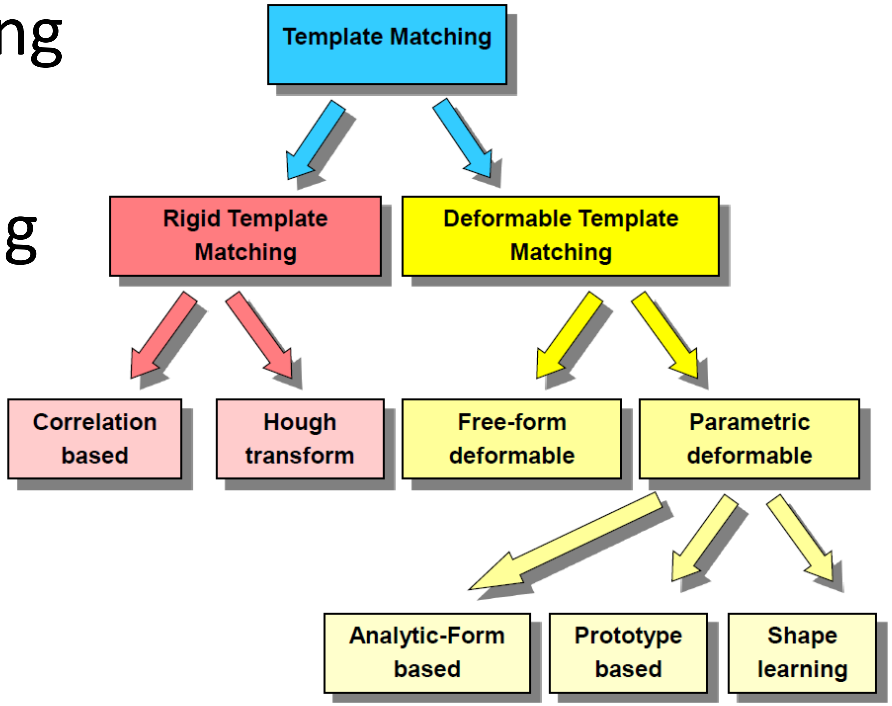


- Dealing with illumination changes
  - Use edge maps instead of images
  - Use ZNCC: subtracts the uniform illumination component
- Dealing with scale changes
  - Matching with several scaled versions of the template
  - Work with multiple rescaled copies of the image
- Dealing with rotation
  - Matching with several rotated versions of the template

# TM in general



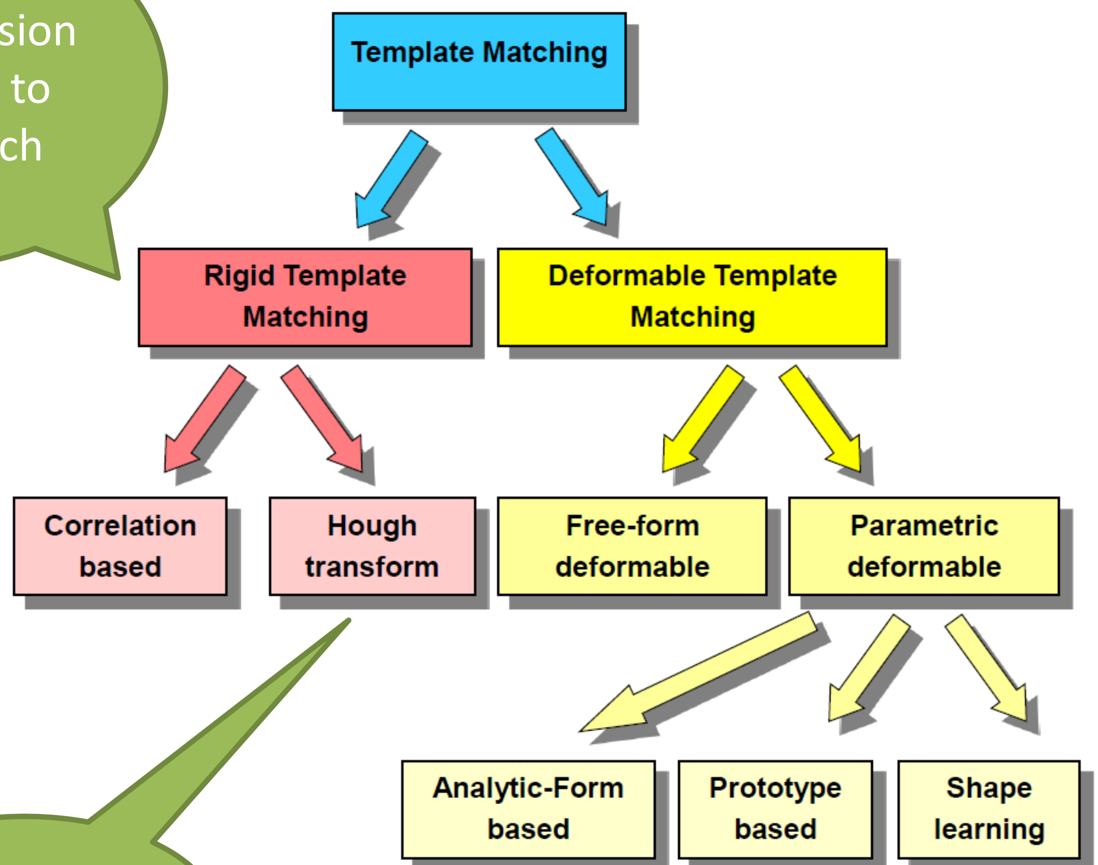
- The basic idea of TM generated a family of approaches
  - Different ways of defining the template
  - Different ways of dealing with the template



# TM in general



Our discussion is related to this branch




Check this!

# Generalized Hough transform



- The generalized Hough transform can be seen as a form of template matching
- The Hough transform works for more complex shapes
- General equation:

$$g(\boldsymbol{v}, \boldsymbol{c}) = 0$$

  – Where $\boldsymbol{v}$ is a vector of coordinates and $\boldsymbol{c}$ a vector of coefficients

# Generalized Hough transform



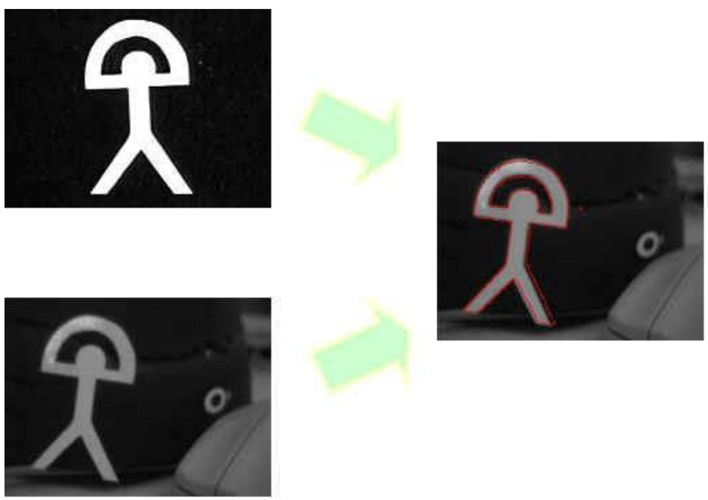
- E.g. (circle):

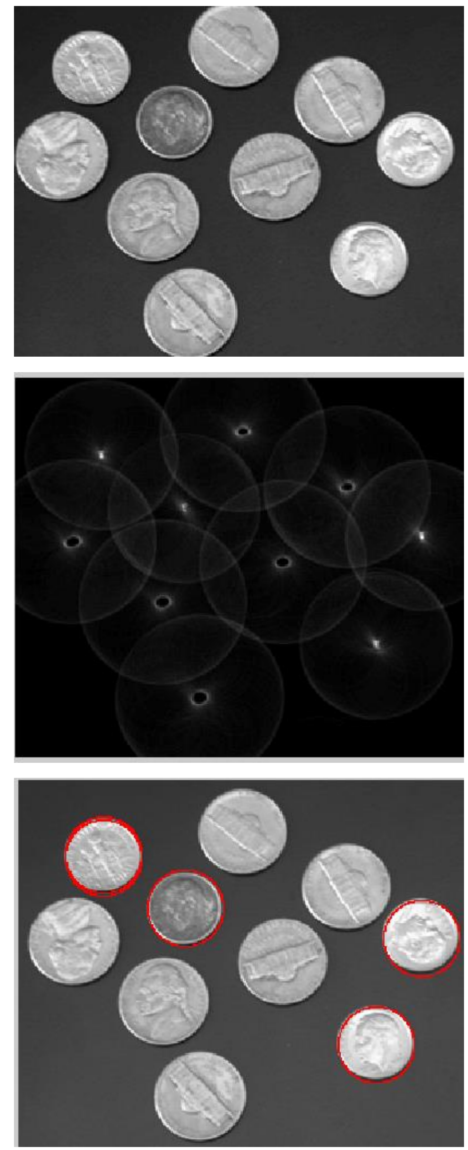
$$(x - c_1)^2 + (y - c_2)^2 = c_3^2$$

- The parameter space might have high dimensionality

# Generalized Hough transform – ex.







# Histogram of Oriented Gradients (HOG)

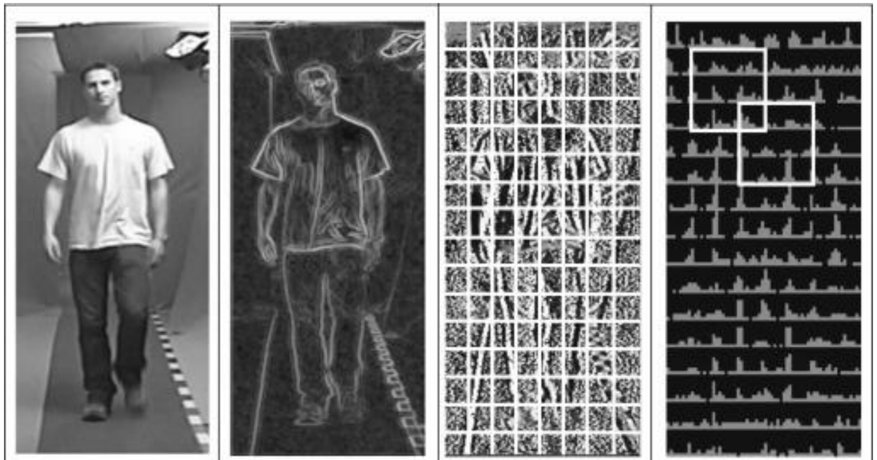
# HOG



- HOG-based detectors work by
  - Sliding a window (similarly to TM)
  - Characterizing the window by evaluating the edge magnitude and phase – this produces a descriptor (similar to feature descriptor)
  - Classifying the descriptor

# HOG descriptor

- HOG descriptor evaluation
  1. Intensity normalization/histogram equalization + smoothing
  2. Calculate edge map (magnitude + phase)

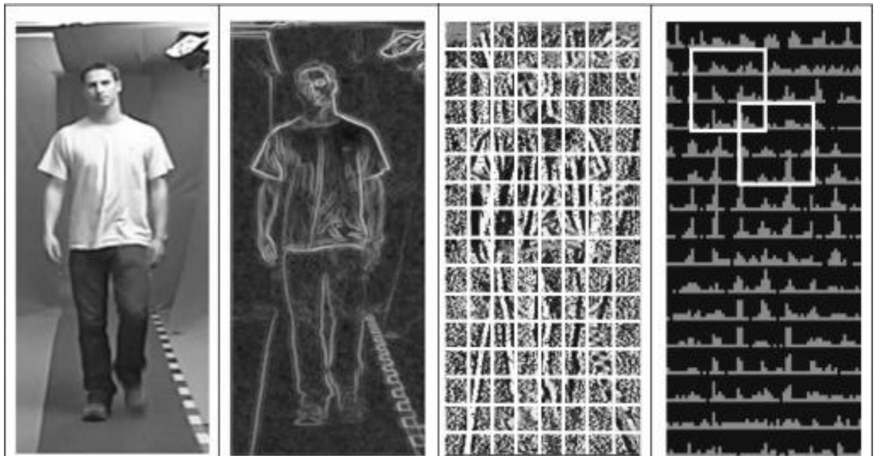


(A) Original image | (B) Gradient magnitude of (A) | (C) Grid of 8×8 cells | (D) Grouping histogram information into blocks

# HOG descriptor



- HOG descriptor evaluation
  3. Evaluate edge histogram on 8x8 non-overlapping cells – this creates voting vectors (e.g., 9 bins of 20° for covering 180°)

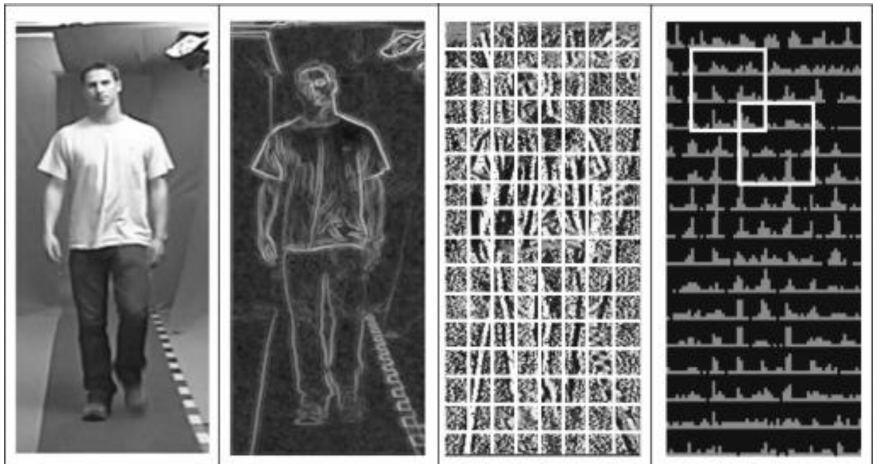


(A) Original image | (B) Gradient magnitude of (A) | (C) Grid of 8×8 cells | (D) Grouping histogram information into blocks

# HOG descriptor

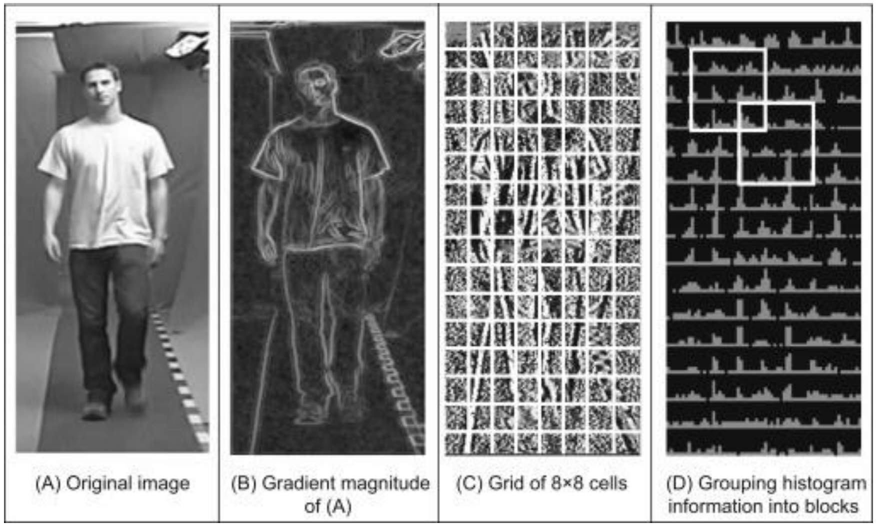


- HOG descriptor evaluation
  4. Create overlapping blocks of 2x2 cells
  5. Normalize voting vectors over each block and create block vectors (36 elements)



(A) Original image | (B) Gradient magnitude of (A) | (C) Grid of 8×8 cells | (D) Grouping histogram information into blocks

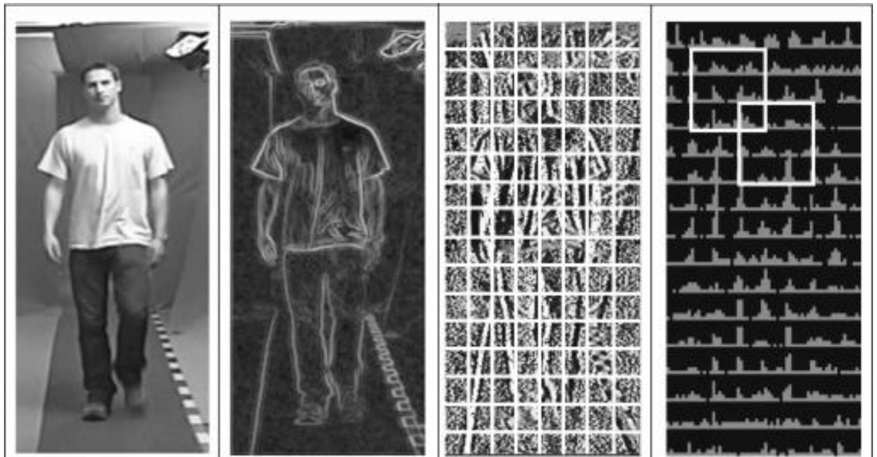
- HOG descriptor evaluation
  6. Serialize block vectors



# HOG descriptor



- Consider a 64x128 window
  - How many bins?
  - How many blocks?
  - What is the feature size?



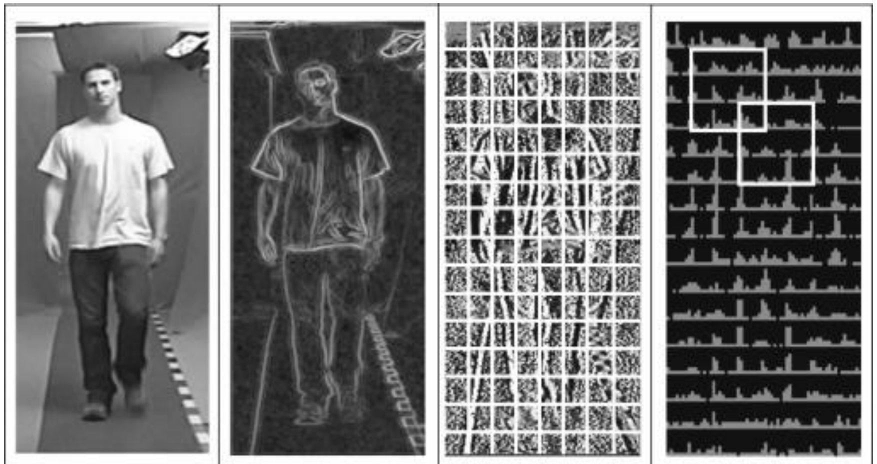
(A) Original image | (B) Gradient magnitude of (A) | (C) Grid of 8×8 cells | (D) Grouping histogram information into blocks

- Anti spoiler ☺

# HOG descriptor



- Consider a 64x128 window
  - How many bins? – 8x16 bins
  - How many blocks? – 7x15 blocks
  - What is the feature size? – 3780 elements



(A) Original image (B) Gradient magnitude of (A) (C) Grid of 8×8 cells (D) Grouping histogram information into blocks

# Characteristics of the HOG descriptor



- The HOG descriptor characterizes the content of a bounding box
- The HOG approach needs BBoxes normalized to a standard size
- Multiple scales can be managed by resizing BBoxes of different dimensions to the standard size

# Using the HOG descriptor

- The HOG descriptor is commonly used to train a classifier
- **Note** – the number (cell and block size) described above are one possible implementation for HOG
  - This has an influence on the descriptor size and meaning

# Bag of words

# Bag of words: approach

- Approach taken from document analysis

Object → Bag of 'words'

Slides adapted from Fei-Fei Li, Rob Fergus, and Antonio Torralba

# Bag of words

- Image and object classification
- Designed to be invariant to several factors
  - Mainly viewpoint and deformations
- Decomposes complex patterns into (semi) independent features

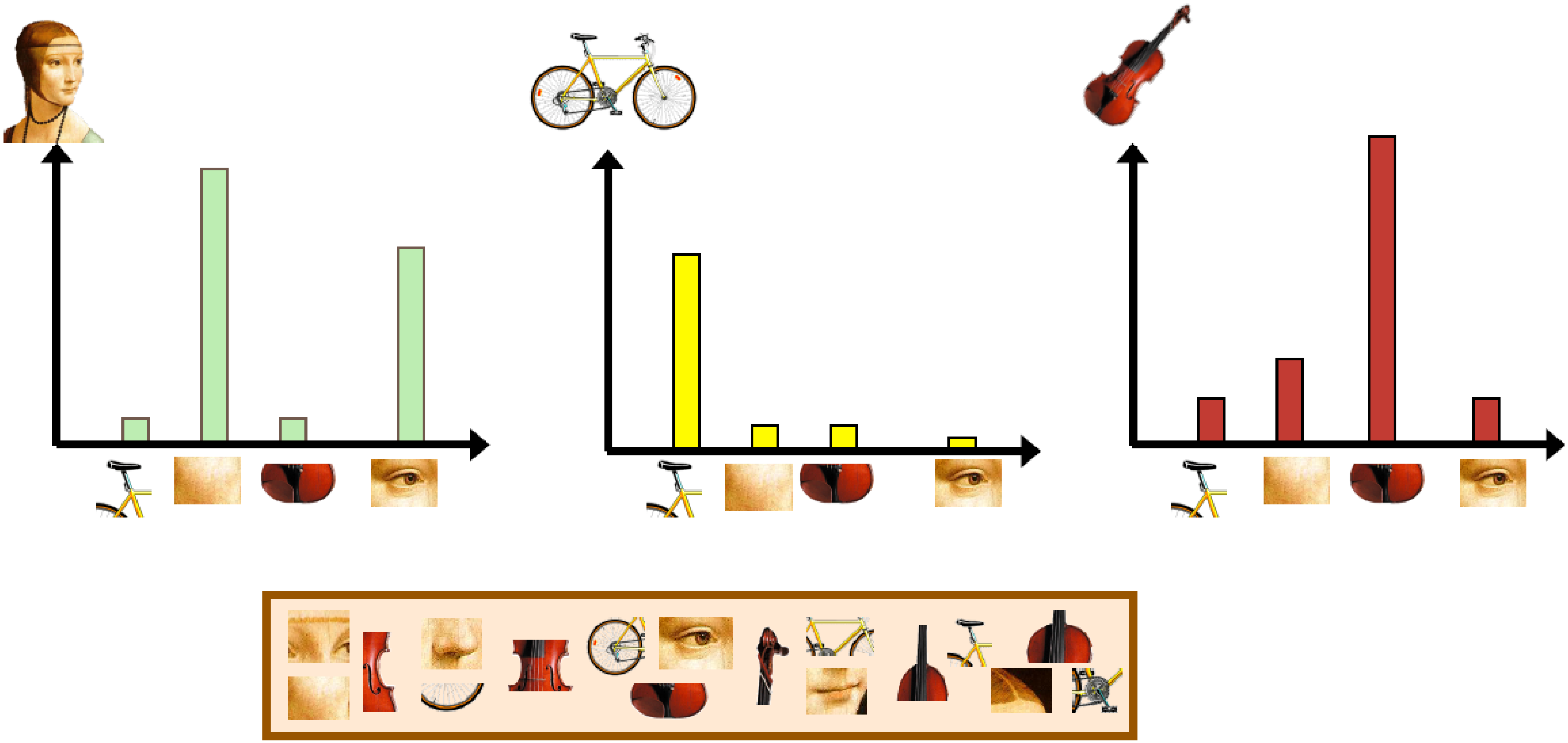
# Bag of words

- Decomposition into visual words

# Bag of words



- Histogram representation



codewords dictionary

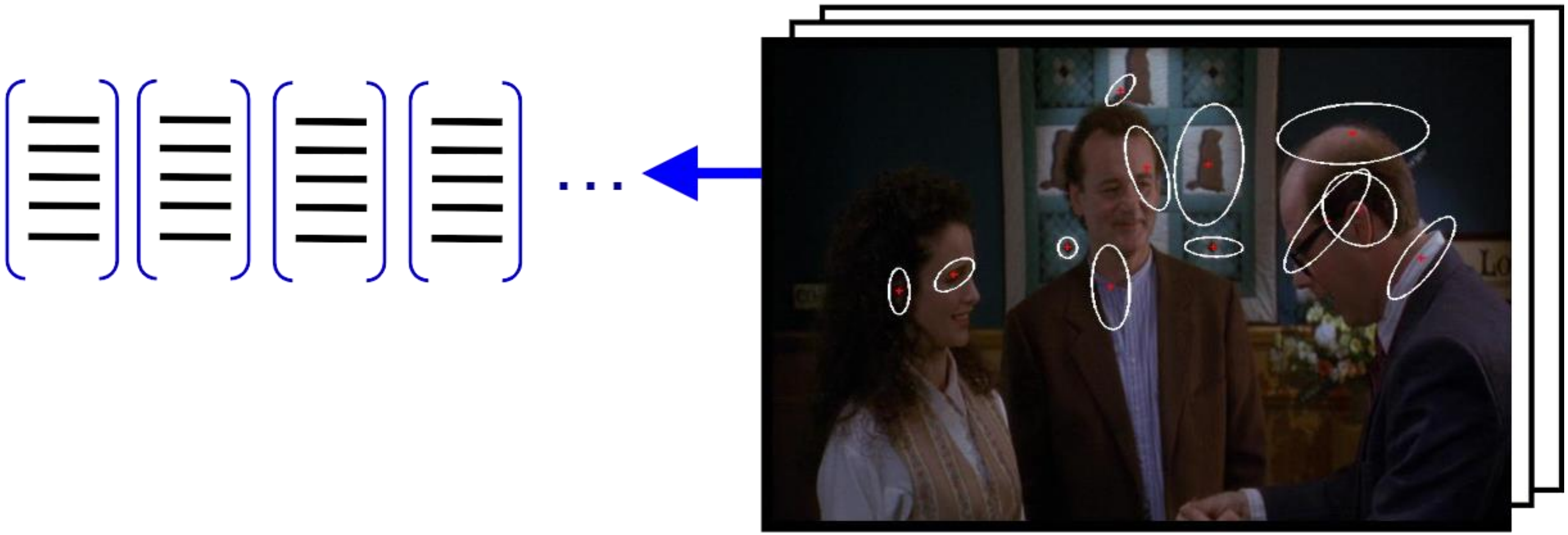
# BoW: image classification



- Words can be represented using features
  - Exploit discriminative properties
  - Exploit invariance properties
  - Re-use an efficient description

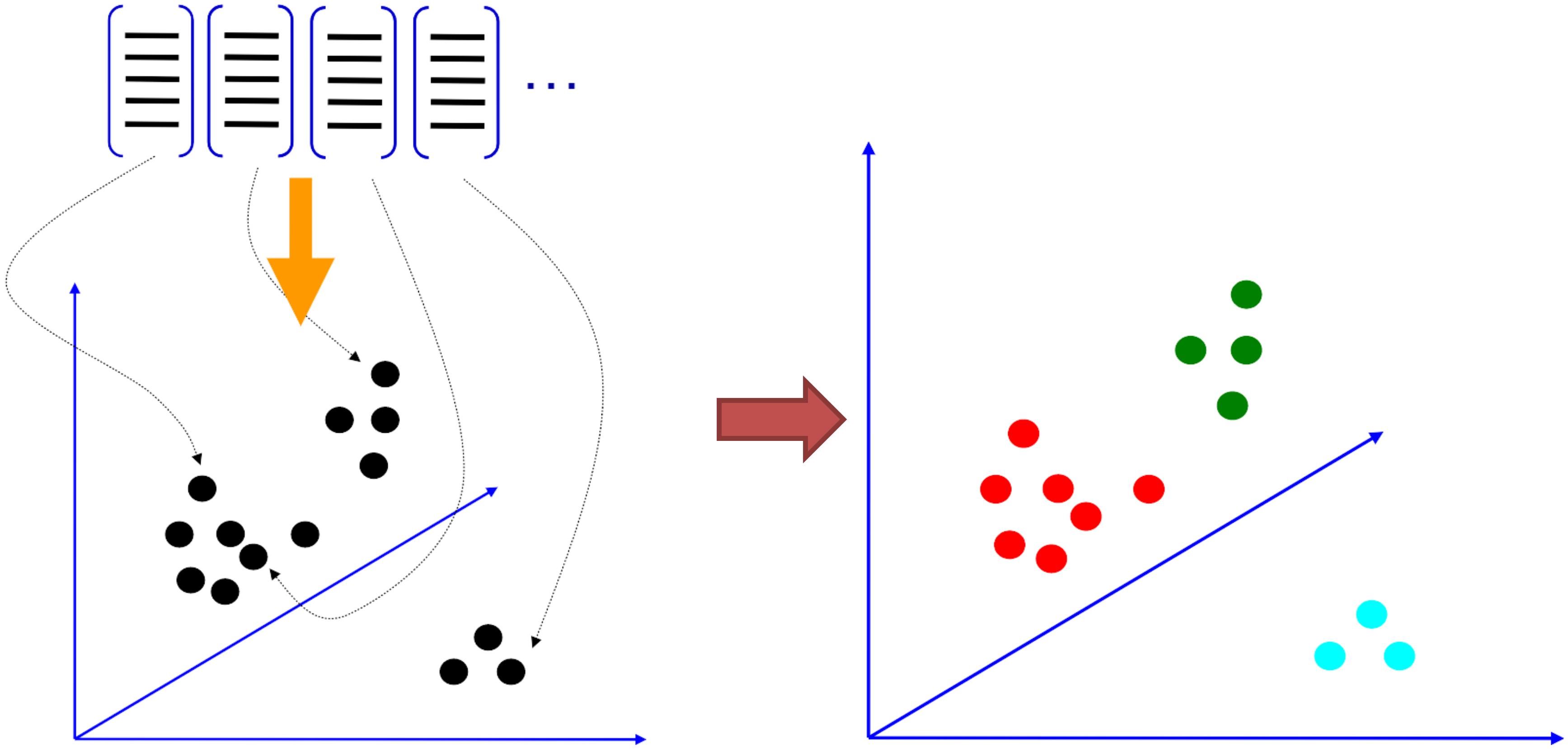
# BoW: image classification

1. Extract features – keypoints and descriptors



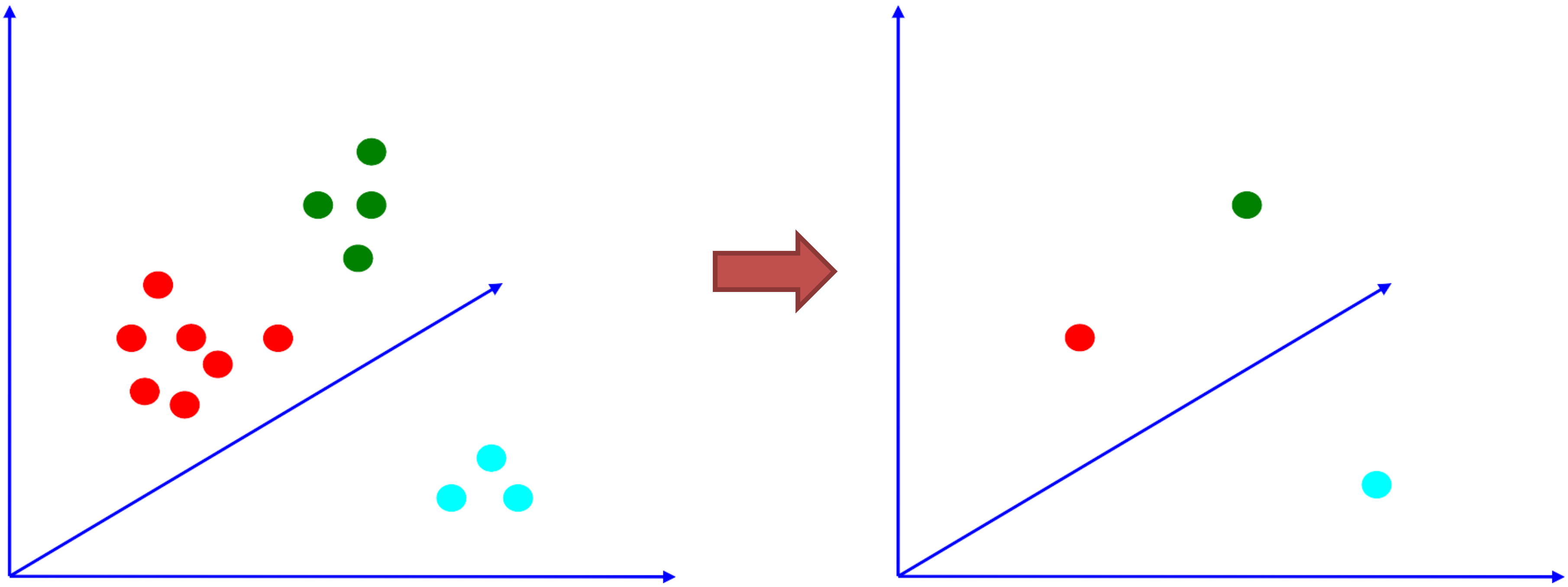
# BoW: image classification

## 2. Clustering in the feature space (e.g., K-means)



# BoW: image classification

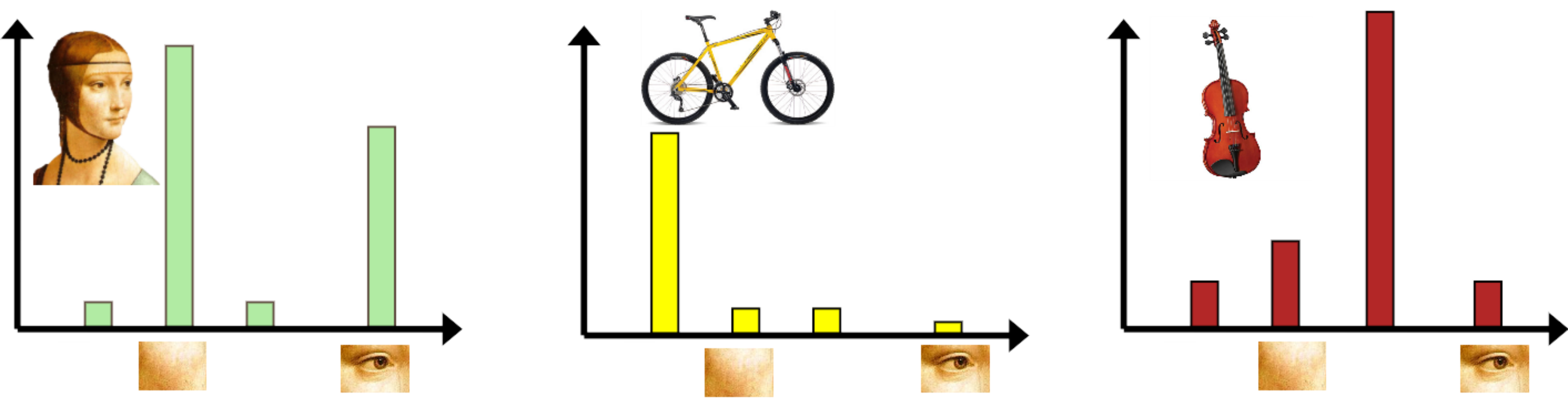
3. Codebook generation: each cluster generates a representative sample (e.g., centroid)



# BoW: image classification



- Image classification:
  - Evaluate the occurrence of each word in the codeword
  - Classify based on histogram



UNIVERSITÀ DEGLI STUDI DI PADOVA

# High-level vision: object recognition, template matching, bag of words

Stefano Ghidoni