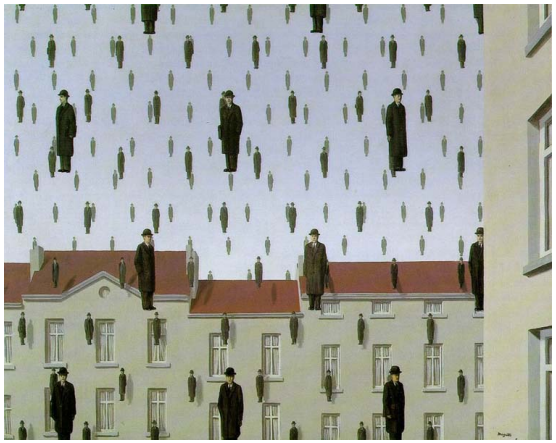
COMPUTATIONAL VISION:

TEMPLATES AND HOG

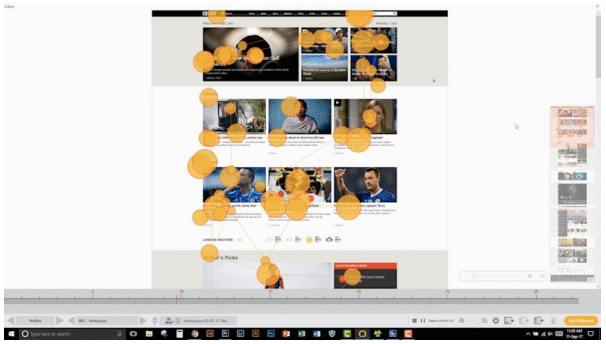
Class 4: Computational Vision



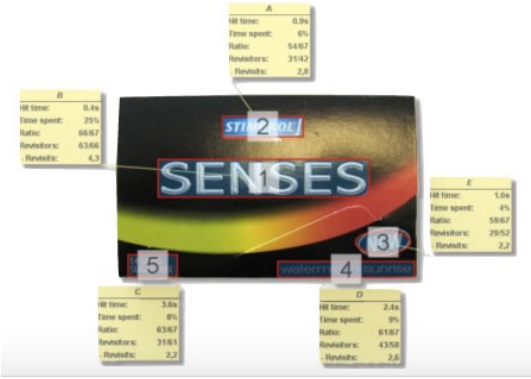
Eye-tracking – why?

Eye-trackers applications

Market research



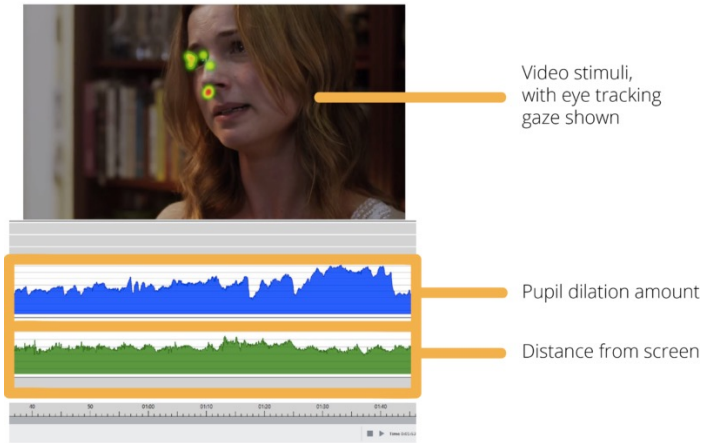
Usability research



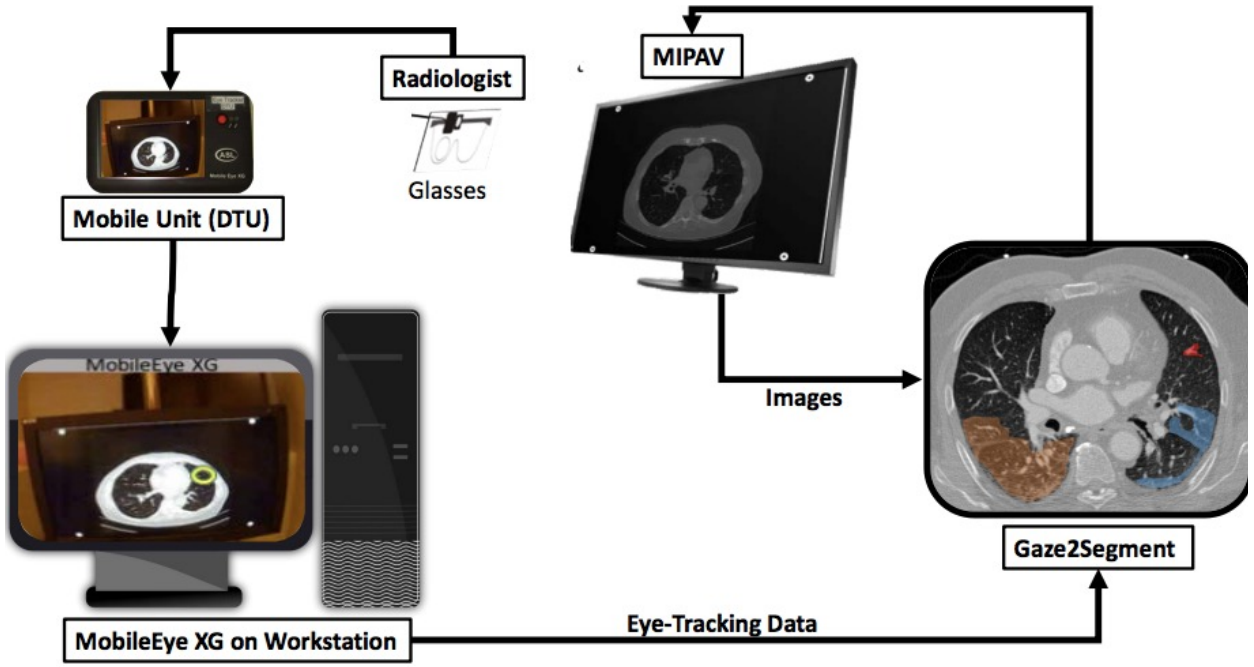
Packaging research

Human factors and simulation research

PC and gaming research



Eye tracking in Medical Image Analysis



	What is measured?	How is it measured?	Which metrics can be derived?	How can the data be interpreted?
Eye tracking (infrared)	Corneal reflection & pupil dilation	Infrared camera point towards eyes	Eye moments (gaze, fixations, saccades), blinks, pupil dilation	Visual attention, engagement, drowsiness & fatigue, emotional arousal

Today's plan

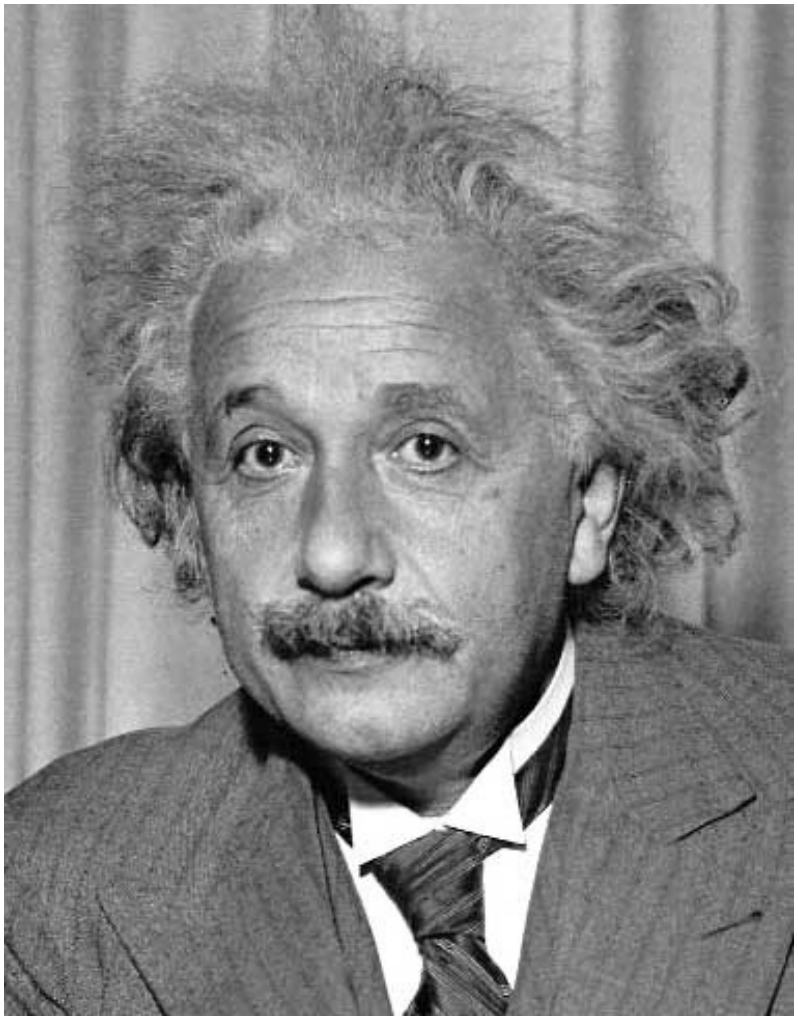
- Template Matching
- What are and why we need image descriptors?
  - A particular kind of image descriptor (HOG)
- Pedestrian detection with HOG
- Image retrieval and classification

Template matching

Goal: find  in an image

Main challenge:

- What is a good similarity or distance measure between two patches?

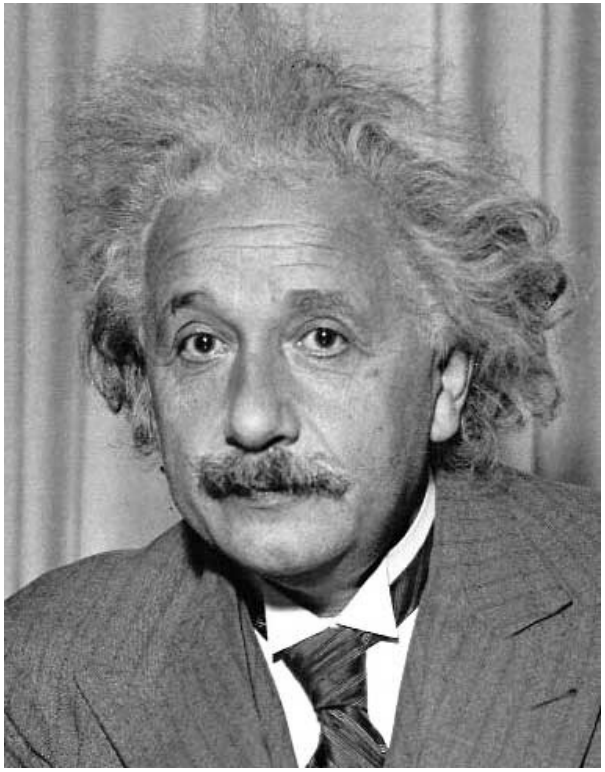


Slide: Hoiem

Template Matching with filters

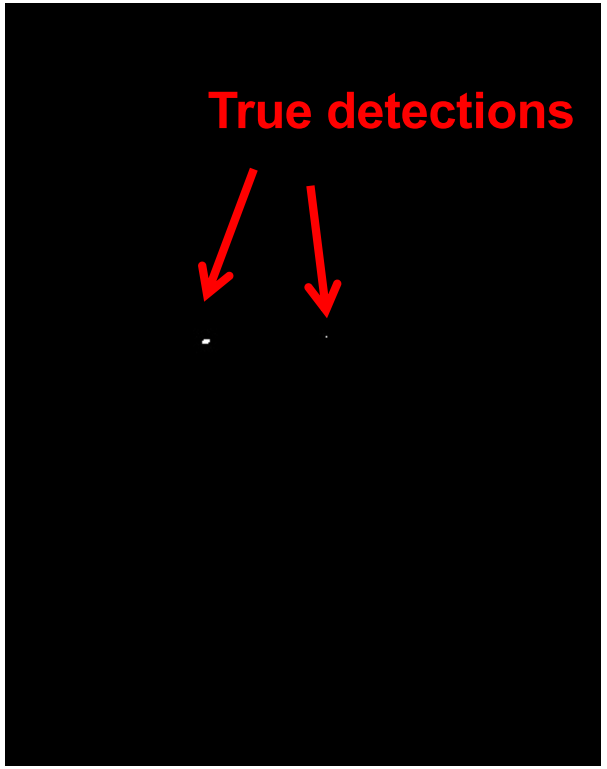
- Goal: find  in image
- Method 1: SSD

$$h[m,n] = \sum_{k,l} (g[k,l] - f[m+k,n+l])^2$$



Input

1- sqrt(SSD)



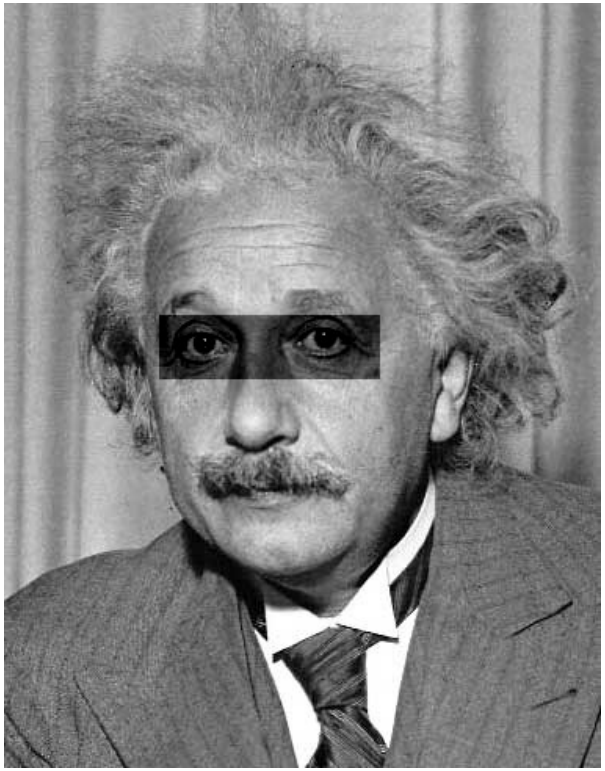
Thresholded Image

Slide: Hoiem

Template Matching with filters

- Goal: find  in image with changed contrast
- Method 1: SSD

$$h[m,n] = \sum_{k,l} (g[k,l] - f[m+k,n+l])^2$$



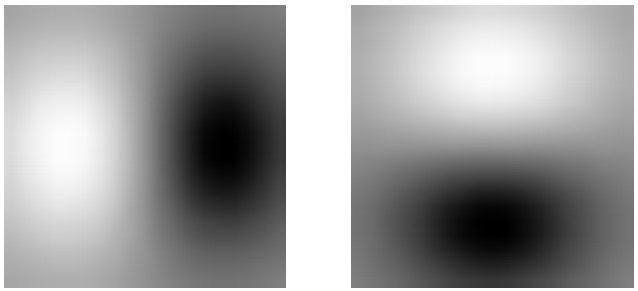
Input

1- sqrt(SSD)

Slide: Hoiem

Image derivative by convolving with Gaussians derivatives

Recall



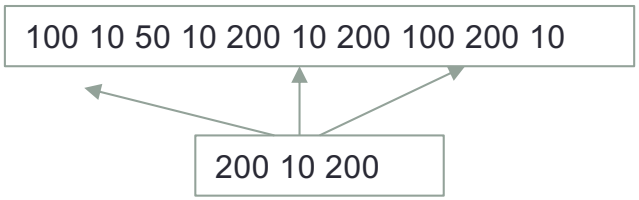
σ = 1 pixel

Template Matching with filters

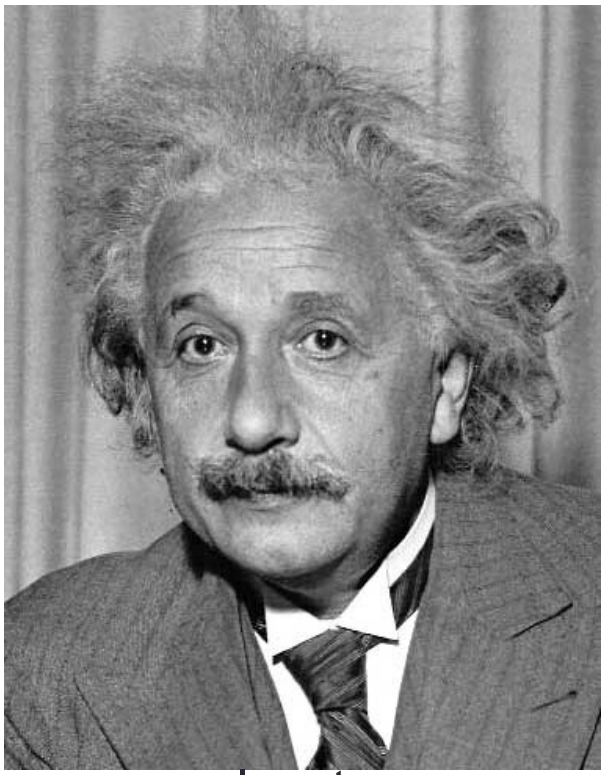
- Goal: find  in image
- Method 2: Convolutional filtering the image with eye patch

$$h[m,n] = \sum_{k,l} g[k,l]\, f[m+k, n+l]$$

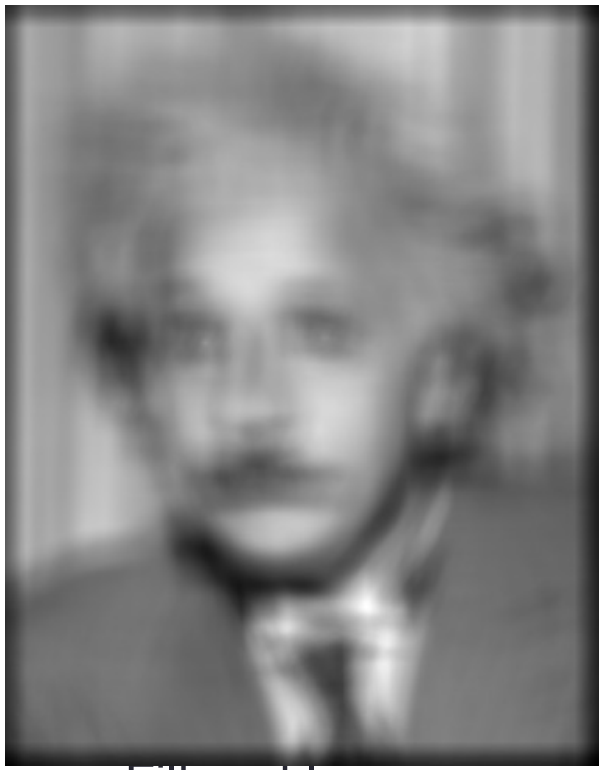
f = image
g = filter



What went wrong?



Input



Filtered Image

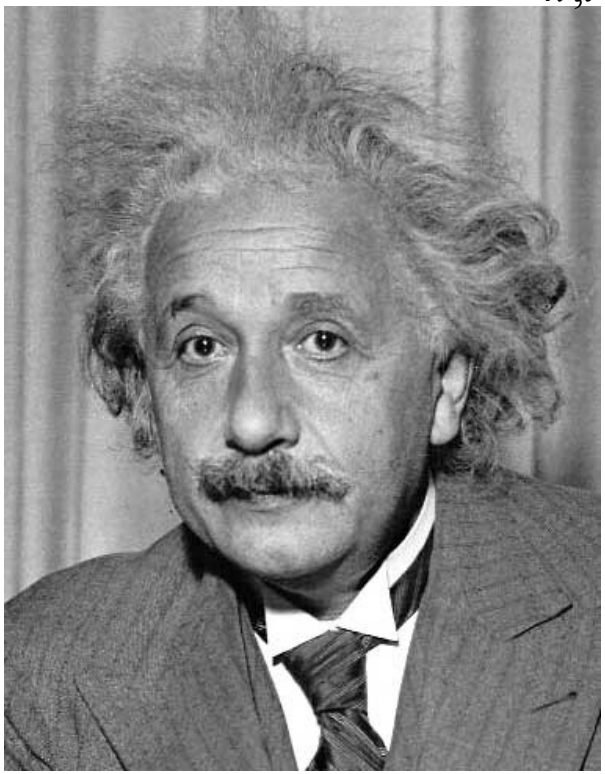
Slide: Hoiem

Template Matching with filters

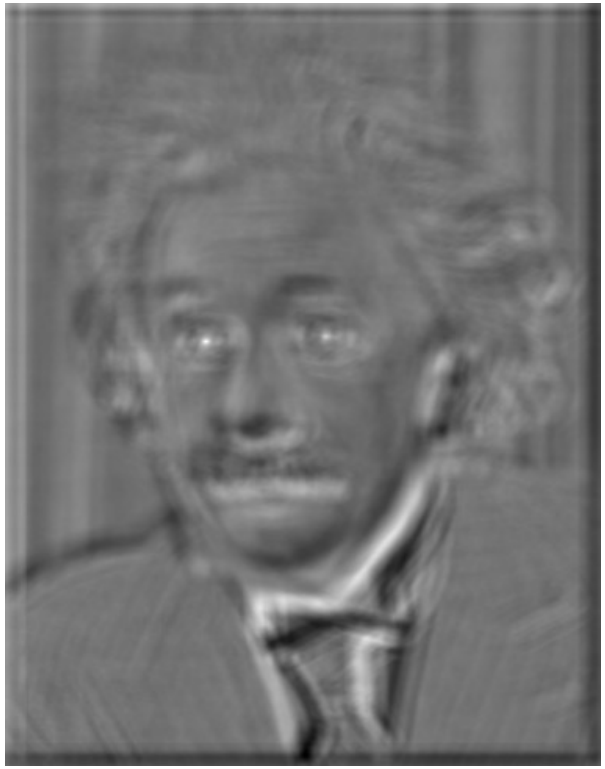
- Goal: find  in image
- Method 3: Convolutional filtering the normalized image

$$h[m,n] = \sum_{k,l} (g[k,l])(f[m+k,n+l] - \bar{f})$$

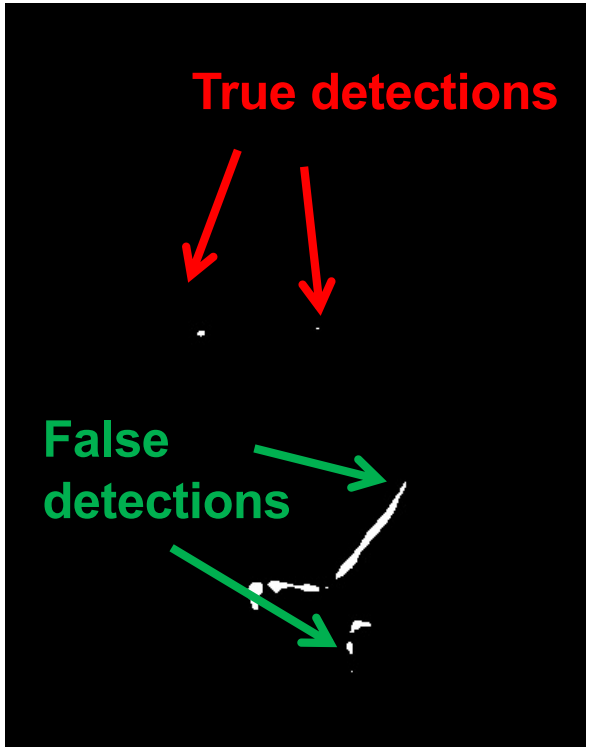
mean of f



Input



Filtered Image (scaled)



Thresholded Image

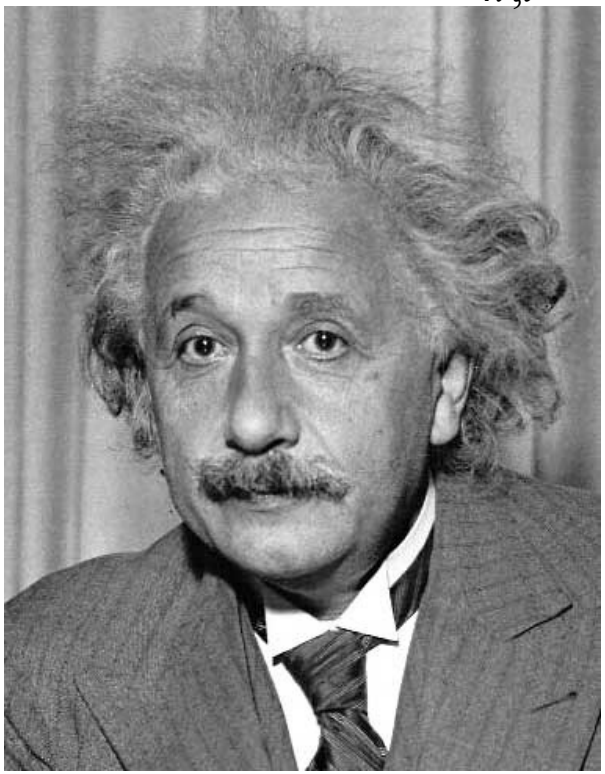
Slide: Hoiem

Template Matching with filters

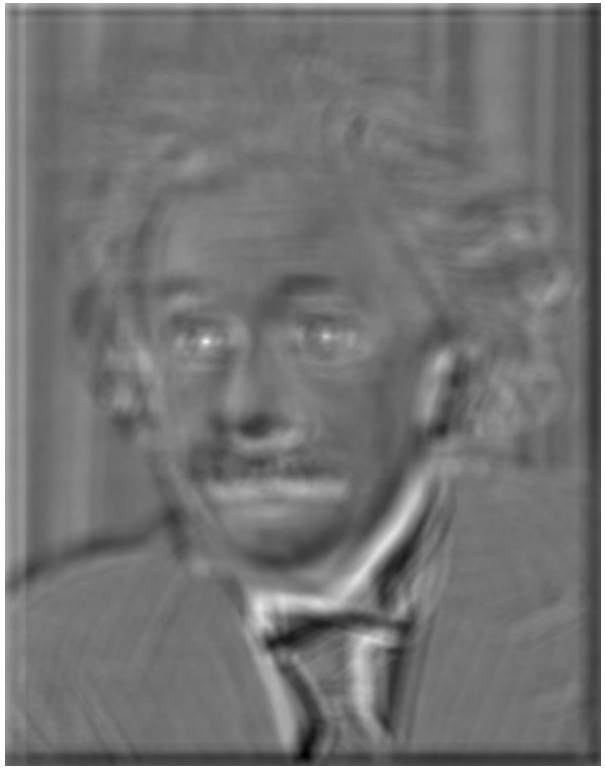
- Goal: find  in image
- Method 3: Convolutional filtering the image with zero-mean eye

$$h[m,n] = \sum_{k,l} (g[k,l] - \bar{g})(f[m+k,n+l] - \bar{f})$$

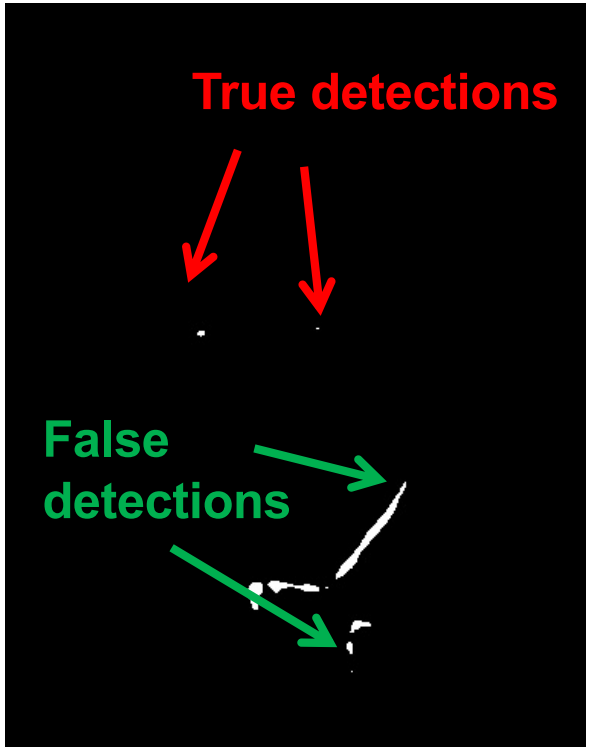
mean of g



Input



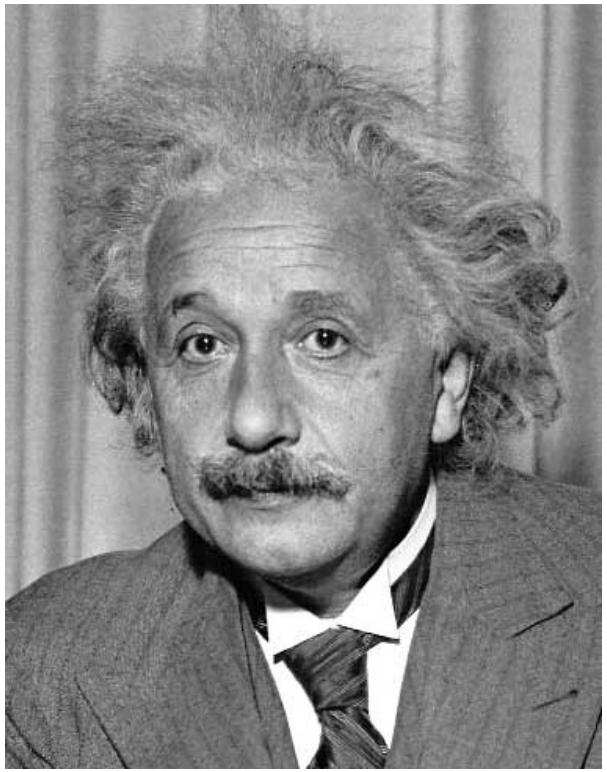
Filtered Image (scaled)

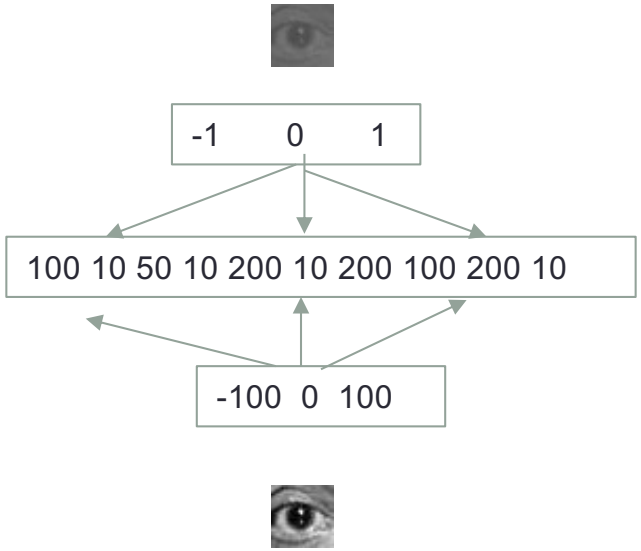


Thresholded Image

Slide: Hoiem

How to make it independent of the template variance?





Template Matching with filters

- Goal: find  in image
- Method 4: Normalized cross-correlation

mean template

mean image patch

$$h[m,n] = \frac{\sum_{k,l}(g[k,l]-\bar{g})(f[m-k,n-l]-\bar{f}_{m,n})}{\left(\sum_{k,l}(g[k,l]-\bar{g})^2\sum_{k,l}(f[m-k,n-l]-\bar{f}_{m,n})^2\right)^{0.5}}$$

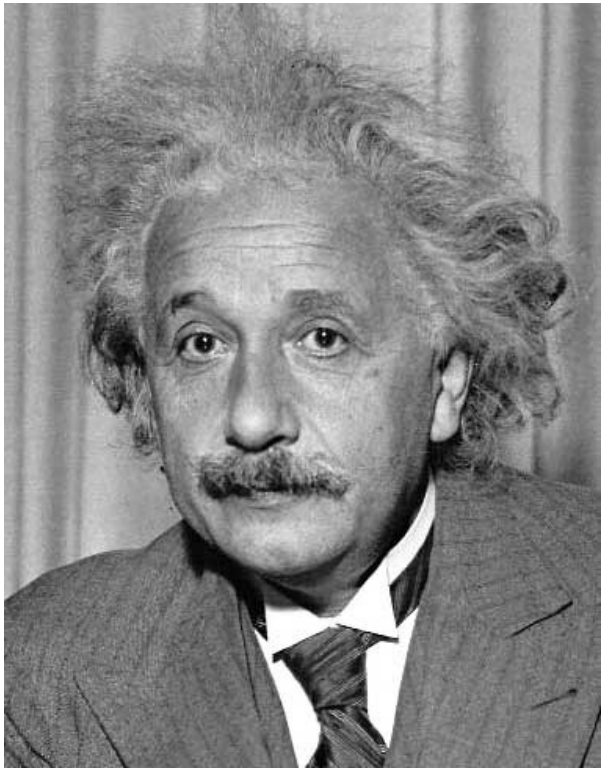
Skimage:

```
result = match_template(image, template)
ij = np.unravel_index(np.argmax(result), result.shape)
x, y = ij[::-1]
```

Slide: Hoiem

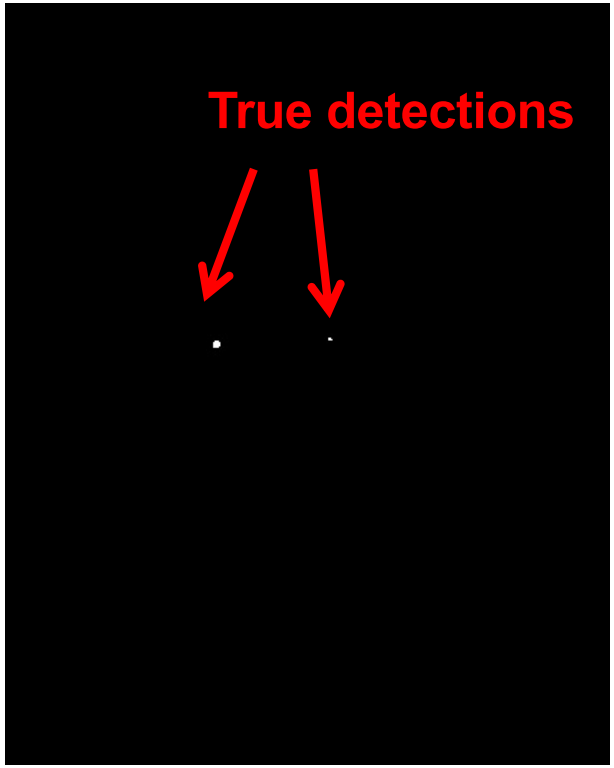
Template Matching with filters

- Goal: find  in image
- Method 4: Normalized cross-correlation



Input

Normalized X-Correlation

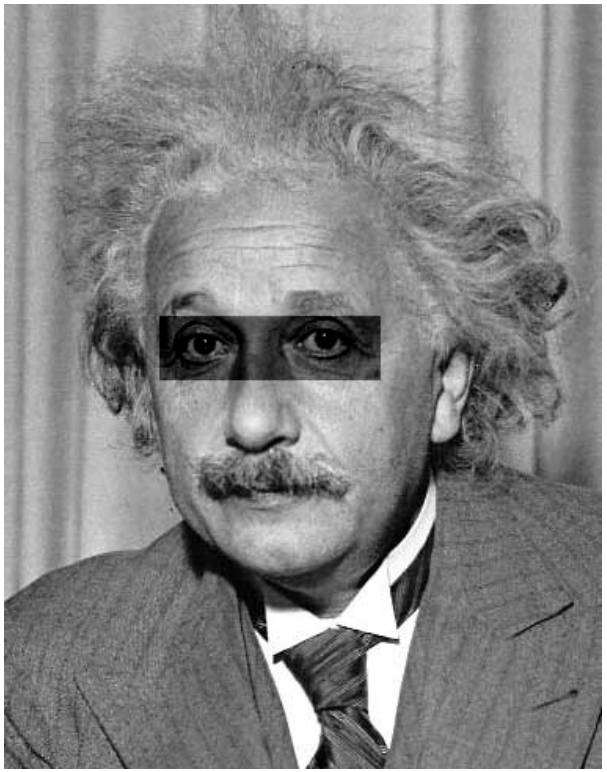


Thresholded Image

Slide: Hoiem

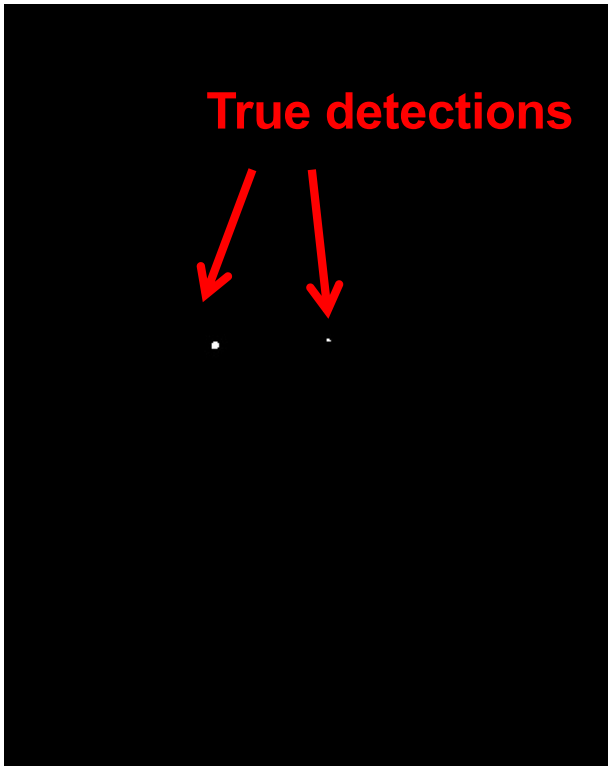
Template matching with filters

- Goal: find  in image
- Method 4: Normalized cross-correlation



Input

Normalized X-Correlation



Thresholded Image

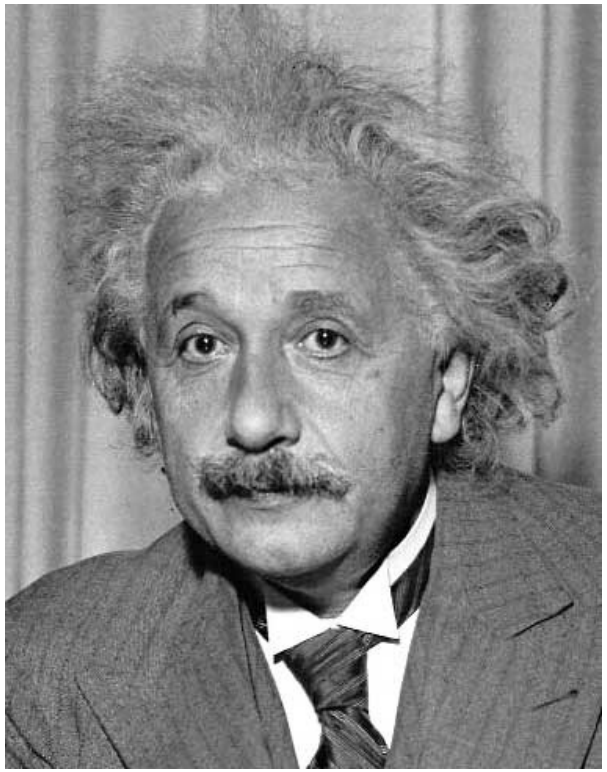
Slide: Hoiem

Template matching

Similarity or distance measure between two patches:

Distance	Properties
Sum of Squared Differences	Fast, but sensitive
Correlation	Less sensitive to illumination changes
Zero-mean correlation	Less sensitive to mask and image values
Normalized X-Correlation	Less sensitive to mask and image variance

How to make it independent of the template variance?



When will not the template matching work?

Today's plan

- Template Matching
- What are and why we need image descriptors?
  - A particular kind of image descriptor (HOG)
- Pedestrian detection with HOG
- Image retrieval and classification

Why we need a descriptor?

To solve real world problems (image retrieval, image classification, etc.), we need to find a connection between:

- a matrix of pixels (raw representation),
- what humans see in an image (face, smile, emoticon).

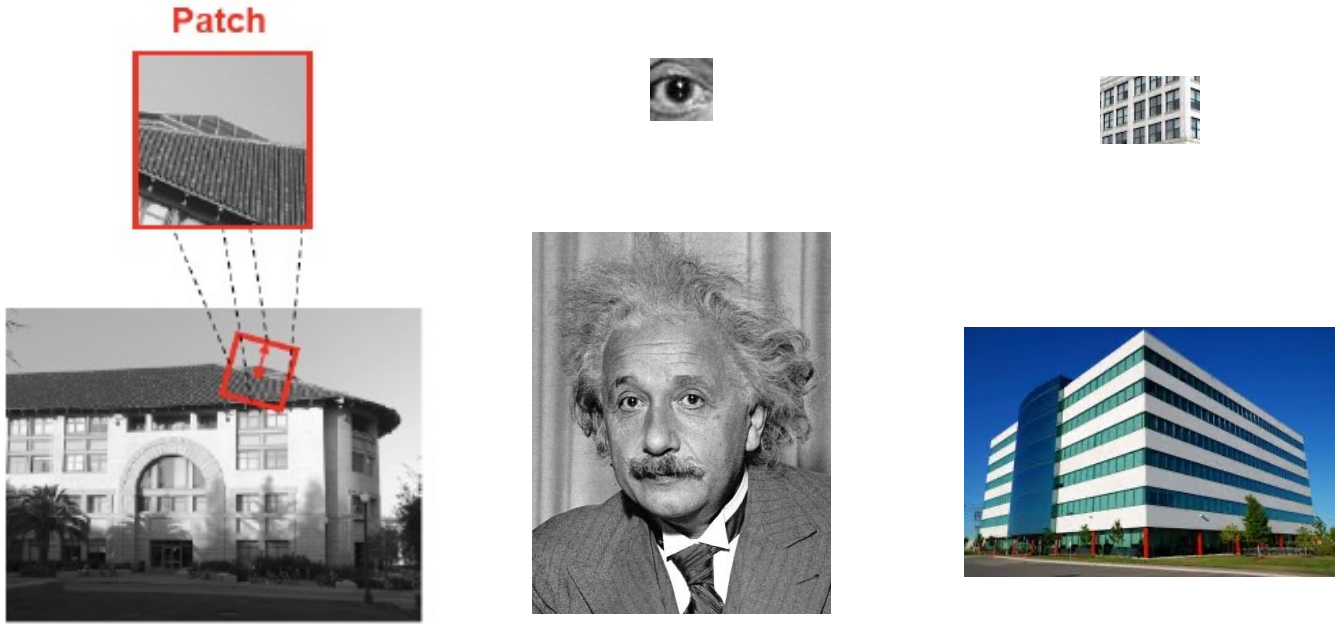
shape
color
....

face
smile
emoticon

Image descriptors allow to describe and represent the image/object by quantities (colour, shape, regions, textures and motion) closer to the visual characteristics perceived by humans.

How to choose the descriptor?

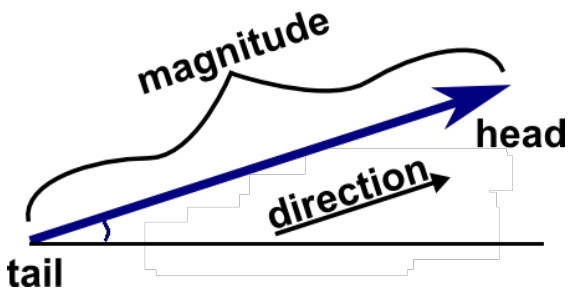
Can we discriminate objects based on their local shape and appearance?



Is the gradient structure characteristic of local or global shape and appearance?

Histogram of gradient (HOG)

Remember what is the image gradient....

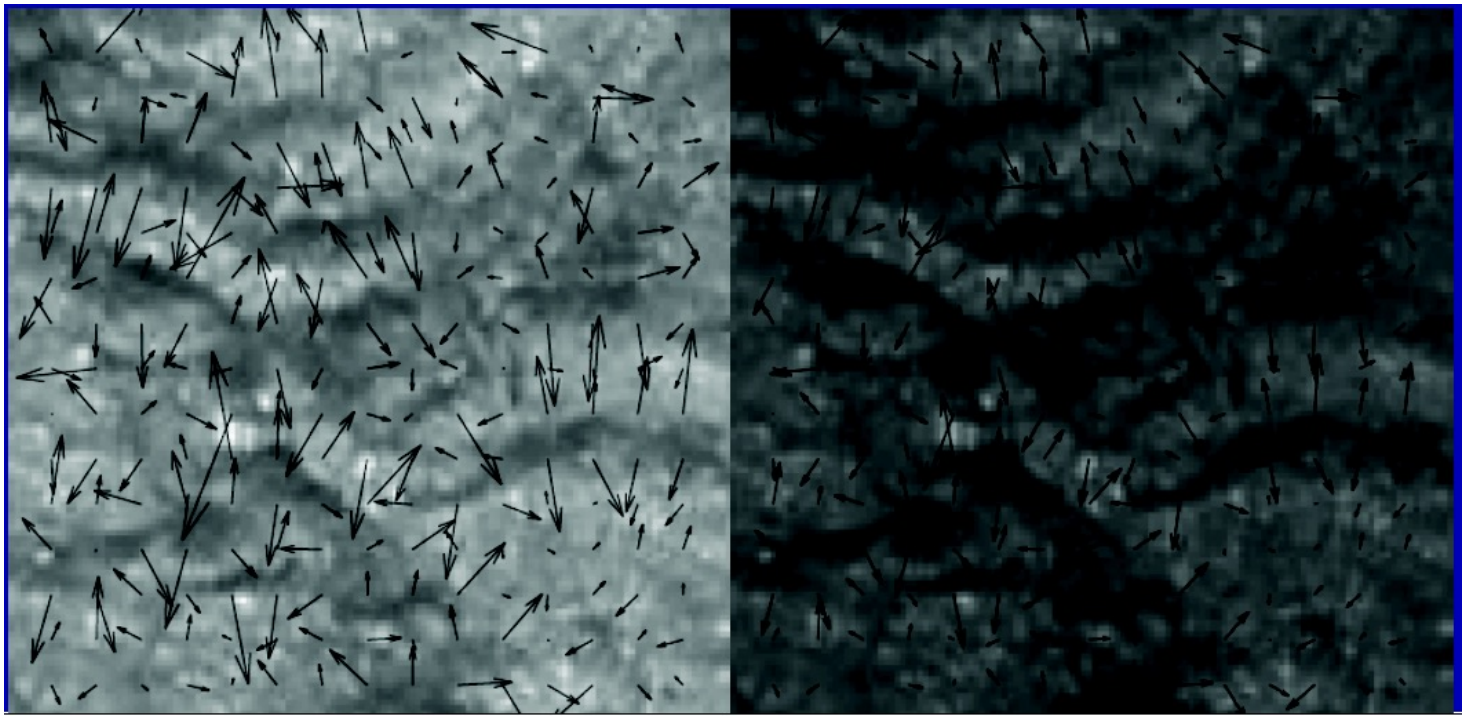


- The image gradient at each pixel is a vector.

- As a vector, it has a magnitude and a direction.

Histogram of gradient (HOG)

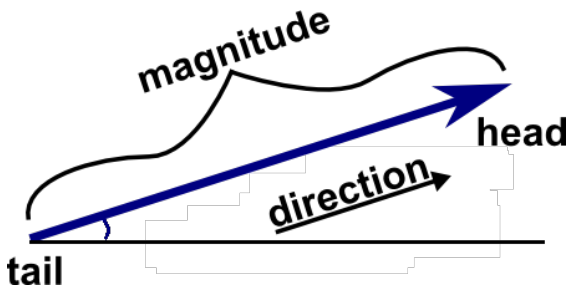
Would the gradient magnitude be useful?



Gradient magnitude is affected by illumination changes!

But the direction isn't!

Histogram of gradient (HOG)



$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

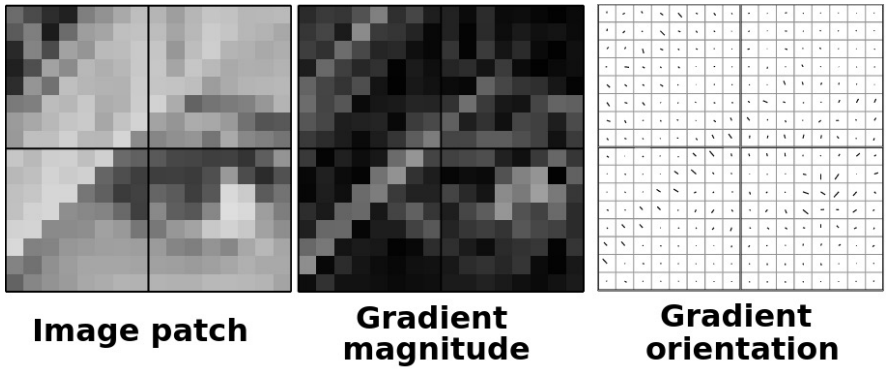
$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

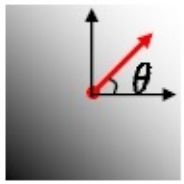
What could be the histogram of gradient

...knowing that the gradient is characterized by the two quantities?

Histogram of gradient (HOG)

Gradient of an image patch:





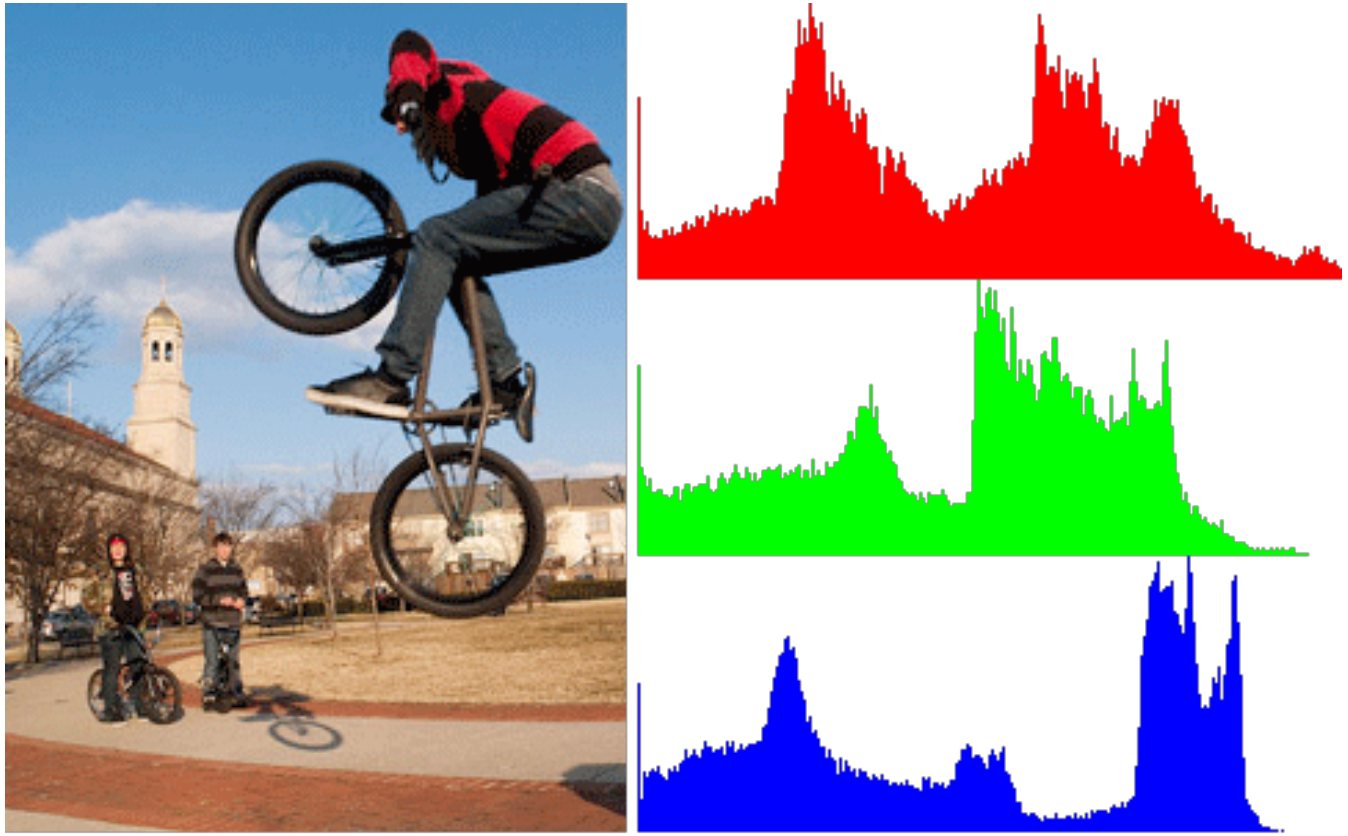
$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

How to obtain the overall orientation of the pixels gradient?

Histogram of gradient (HOG)

Remember what is the histogram of colour images...

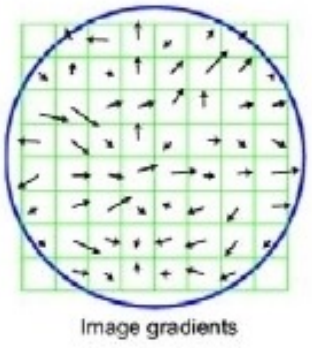


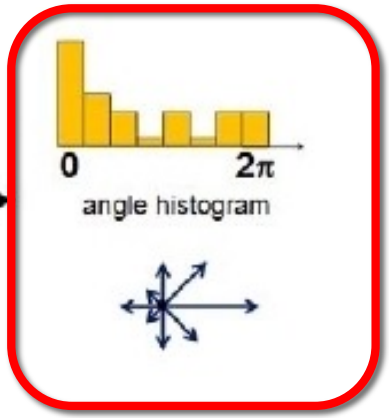
Histogram of gradient (HOG)

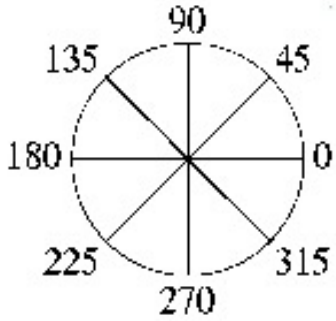
Histogram of gradient orientations

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$



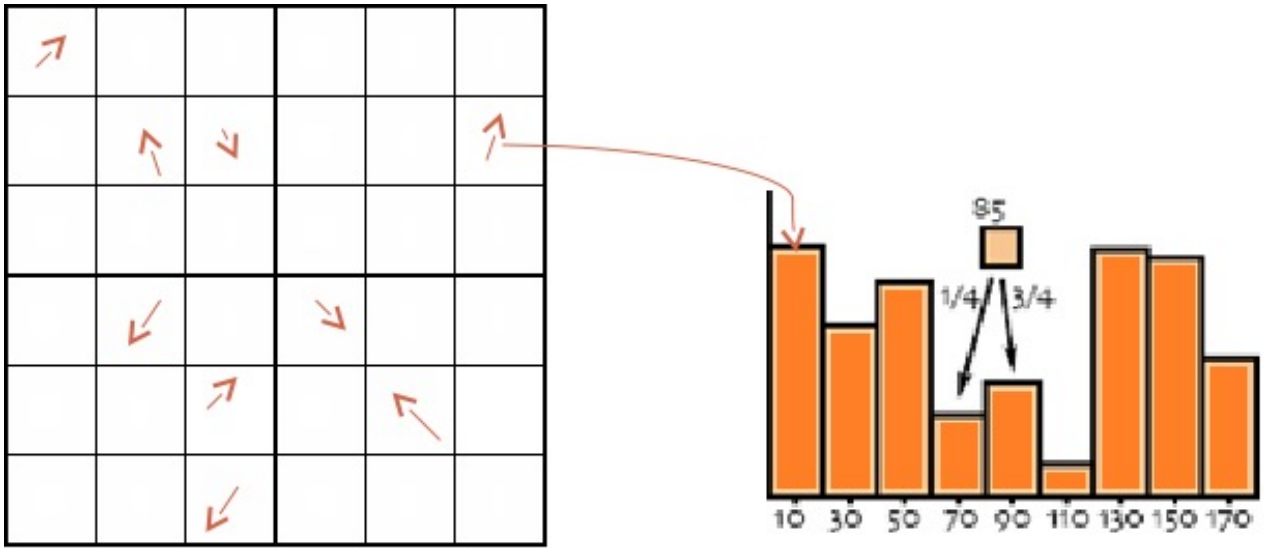




- The gradient orientation is an angle
- Count occurrences of gradient orientation in a patch
- Quantize to 8 bins, each bins cover 45 degrees
- Visual representation of the histogram

Histogram of gradient (HOG)

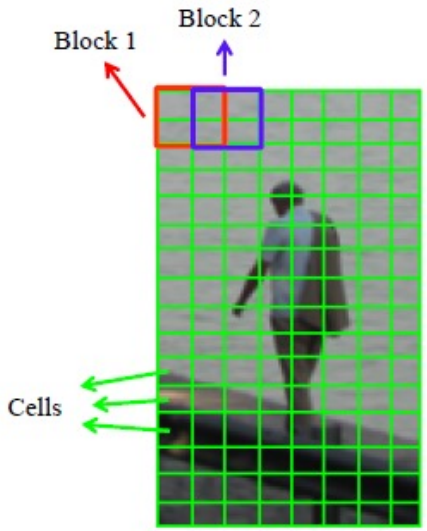
Histogram of gradient orientations



- From 0 to 180 degrees, 9 bins, 20 degrees per bin
- θ= 85 degrees
- To which bin it contributes?

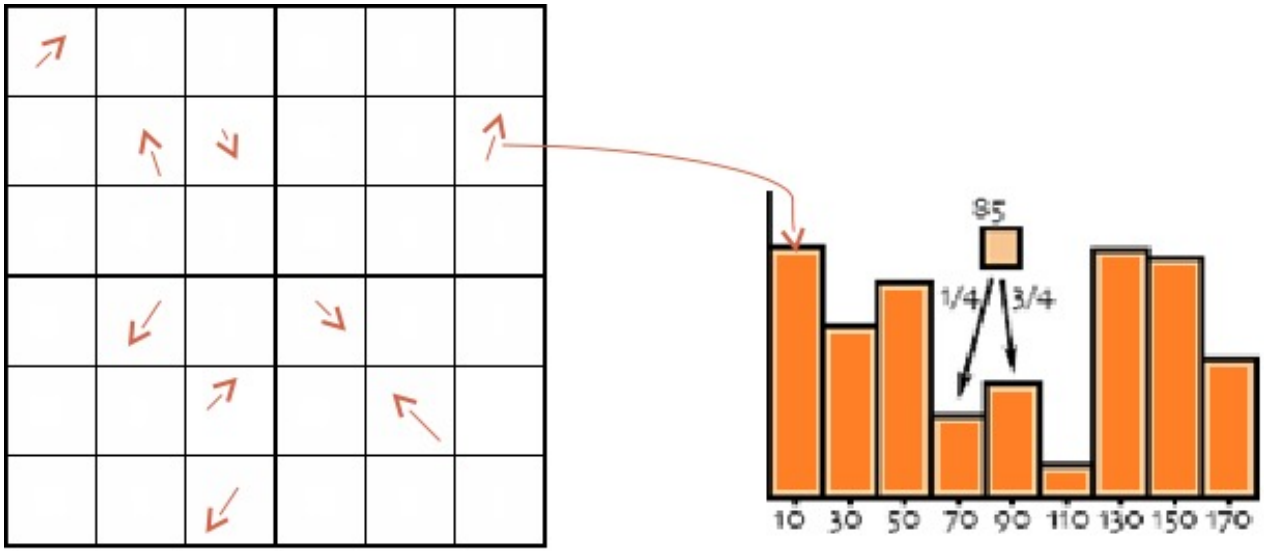
Histogram of gradient (HOG)

- Divide the image into small connected regions called **cells**.
- Compute a local histogram for each cell weighted by gradient magnitude
- Simply concatenate the histogram of the cells.



Histogram of gradient (HOG)

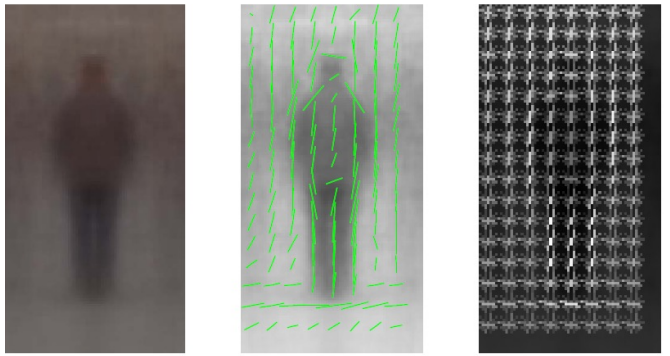
Histogram of gradient orientations



- Compute the distance to adjacent bin centers (from Bin 70 and Bin 90 are 15 and 5 degrees, respectively).
- Divide the distance by the size of the bins (distance/binsize): 5/20=1/4, 15/20=3/4
- Weight the contribution by the gradient magnitude

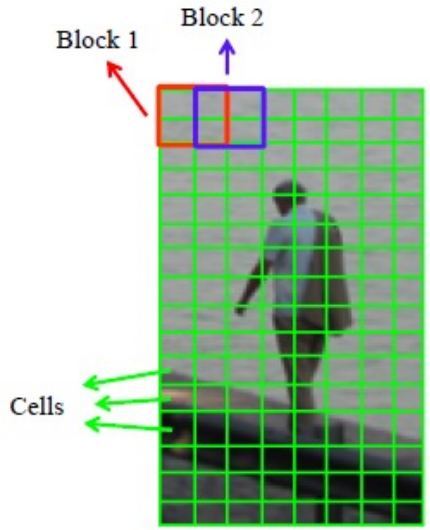
Contrast - normalization

- Gradient strengths vary over a wide range owing to local variations in illumination and foreground-background contrast.
- **How to achieve invariance to changes in illumination or shadowing?**
- Compute a measure of intensity across a larger region than a cell (a block)
- Normalize all cells within the block with this intensity value
- L_2 normalization: $L_2 = \sqrt{(||v||^2_2 + \varepsilon^2)}$, ε is a regularization parameter.



Histogram of gradient (HOG)

- For a 64x128 image
- Divide the image in cells of 8x8 pixels (8x16 cells)
- Group cells into blocks of 2x2 cells (16x16 pixels) of 50% overlap
- Total number of blocks: 7x15= 105
- Quantize the gradient orientation into 9 bins
- Concatenate histograms: 105 x 4 x 9 = 3780 feature vector



Compute gradient in practice

- Convolve the image with discrete derivative mask:
  - $D_x=[-1, 0, 1]$, $D_y=[1, 0, -1]^T$
- Angles: $\tan^{-1}(D_y/D_x)$
- Magnitude: $\sqrt{(D_y^2 + D_x^2)}$

Compute gradient in practice

-1	0	1

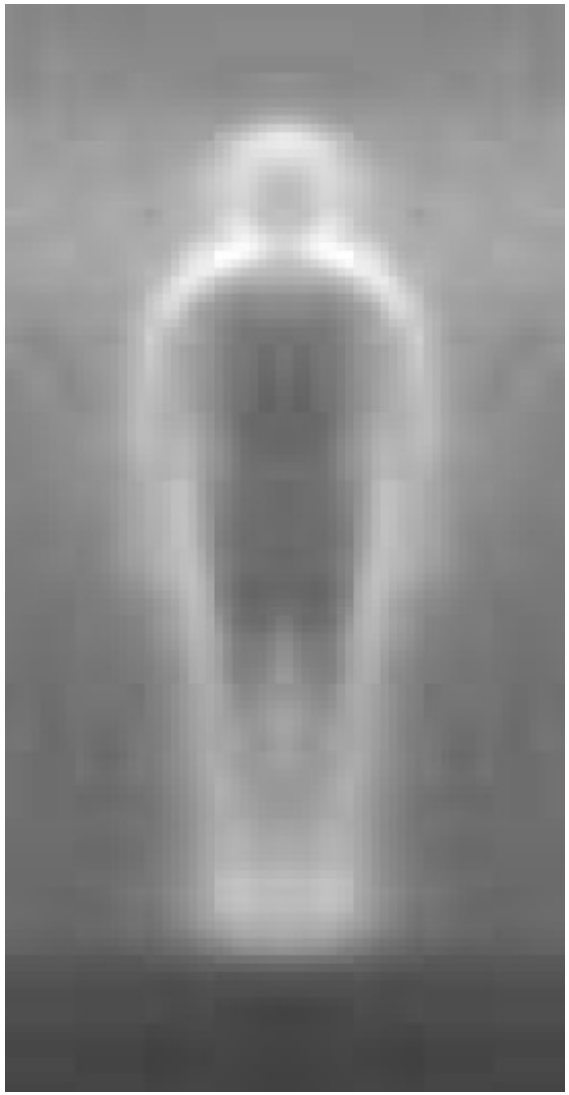
centered

-1	1

uncentered

1	-8	0	8	-1

cubic-corrected



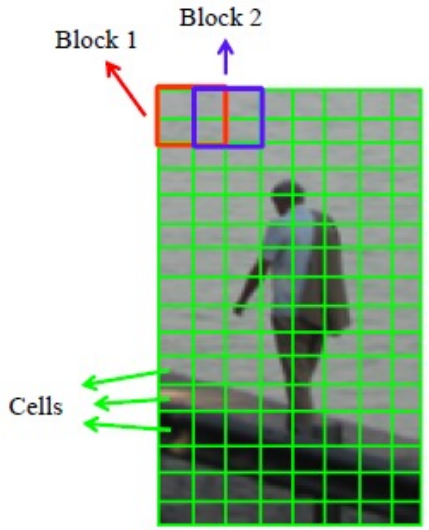
0	1
-1	0

diagonal

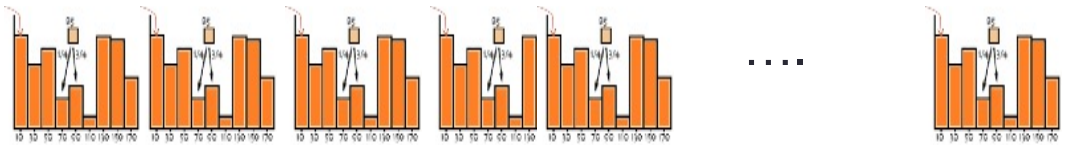
-1	0	1
-2	0	2
-1	0	1

Sobel

Histogram of gradient (HOG)



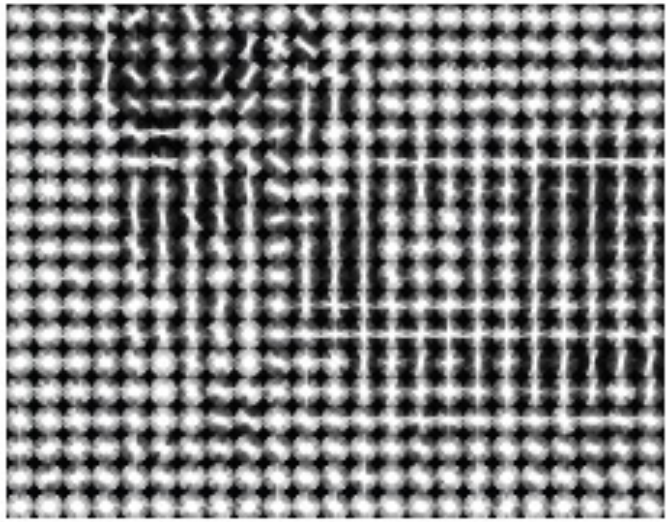
Image/patch descriptor:

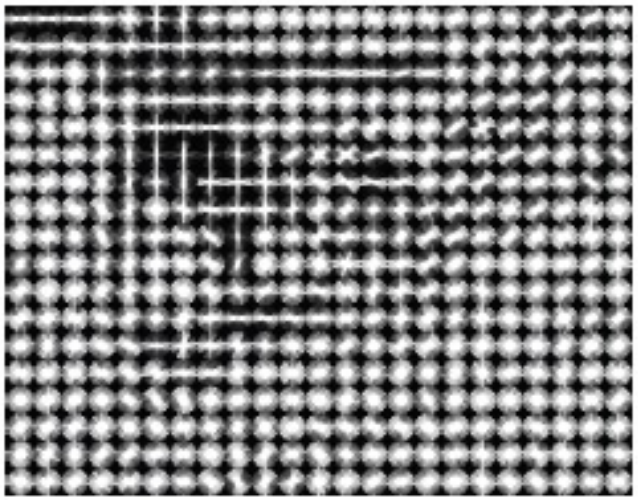


- Concatenated histograms: 105 x 4 x 9 = 3780 feature vector

Histogram of gradient (HOG)

Can we say that the HOG is able to describe local shape and appearance?





Does the HOG descriptor carry information about the gradient or edge positions?

Today's plan

- Template Matching
- What are and why we need image descriptors?
  - A particular kind of image descriptor (HOG)
- Pedestrian detection with HOG
- Image retrieval and classification

Pedestrian detection

Pedestrian detection

Why is the problem difficult?

- Wide variety of articulated poses
- Variable appearance/clothing
- Complex backgrounds
- Unconstrained illumination
- Occlusions
- Different scales

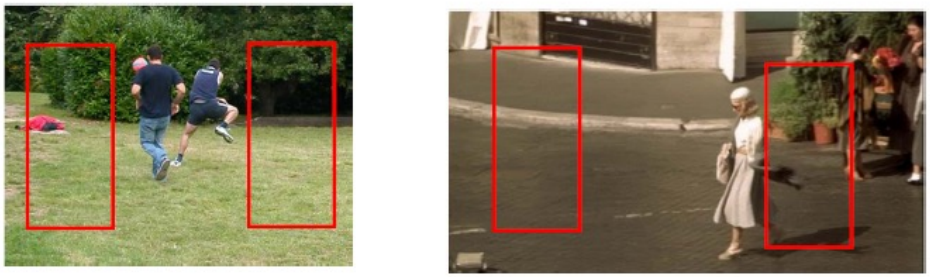
Pedestrian detection

- Trasform the detection problem into a binary (yes/not) classification problem:

  "Is a given window representing a pedestrian or not?"

  - Positive data – 1208 positive window examples

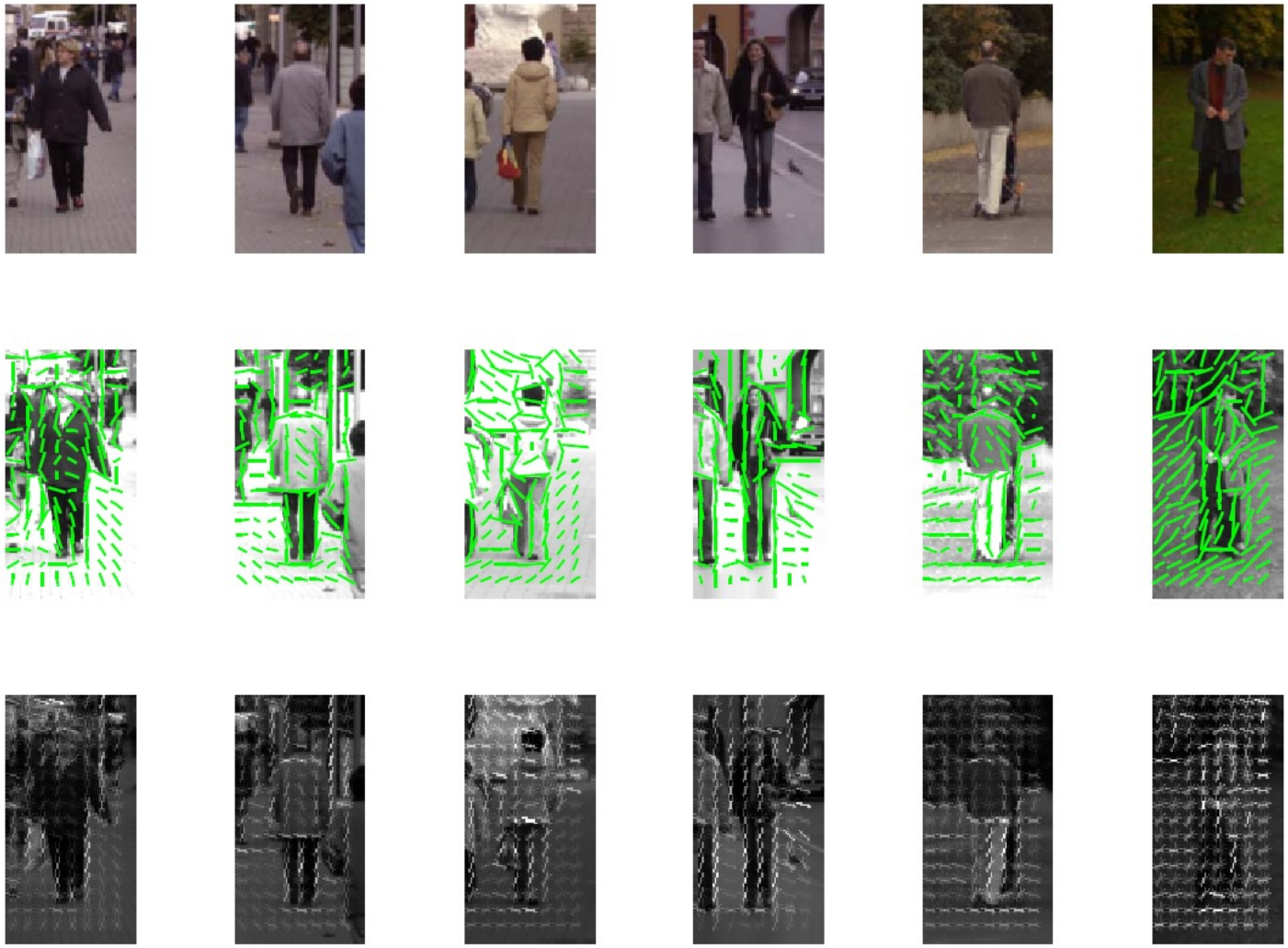
  - Negative data – 1218 negative window examples (initially)



Dalal and Triggs, CVPR 2005

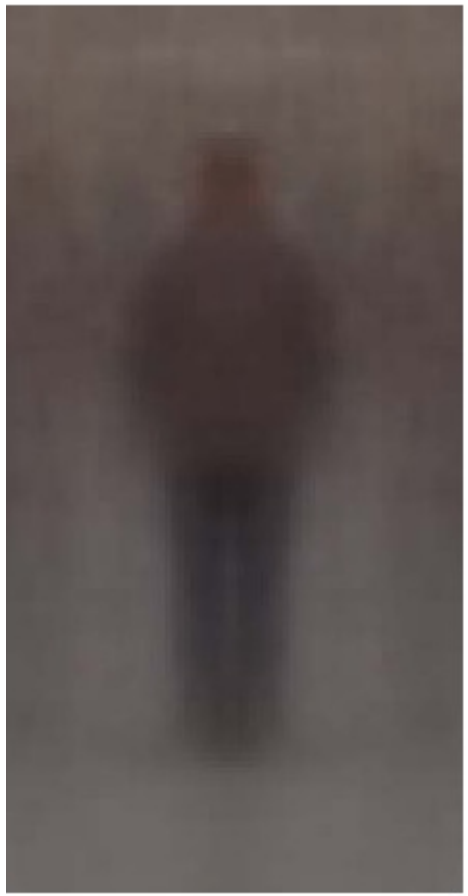
Pedestrian detection

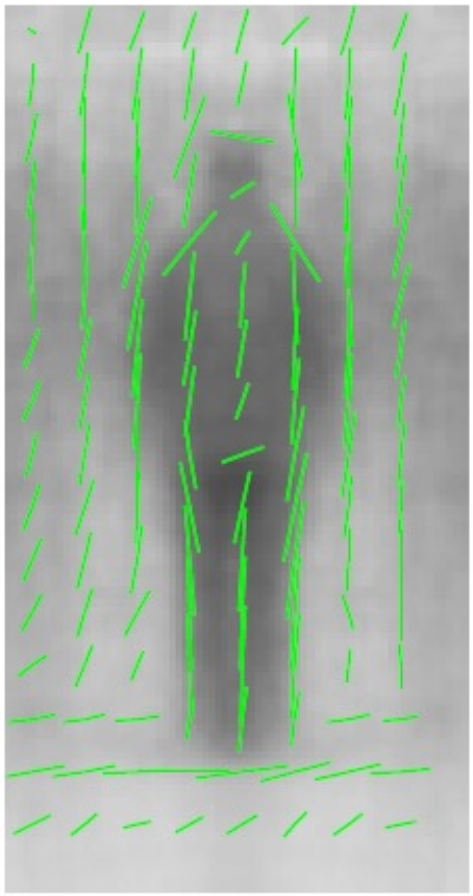
Compute HOG descriptor for all training samples



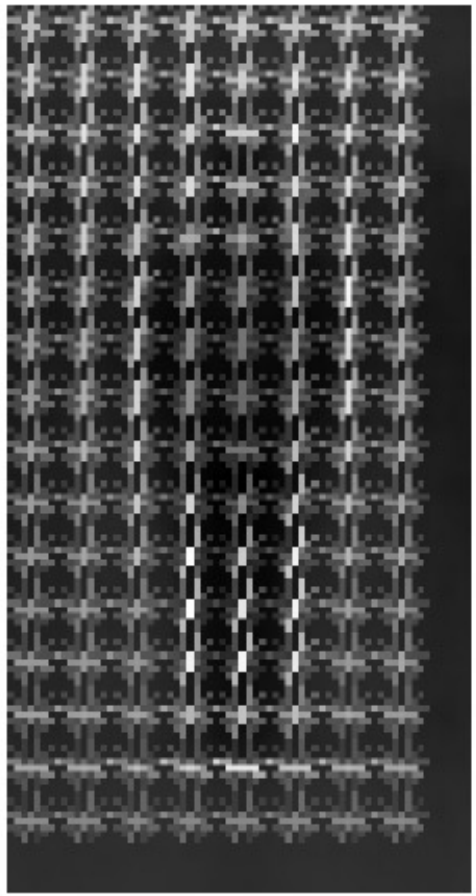
Pedestrian detection

Averaged positive examples



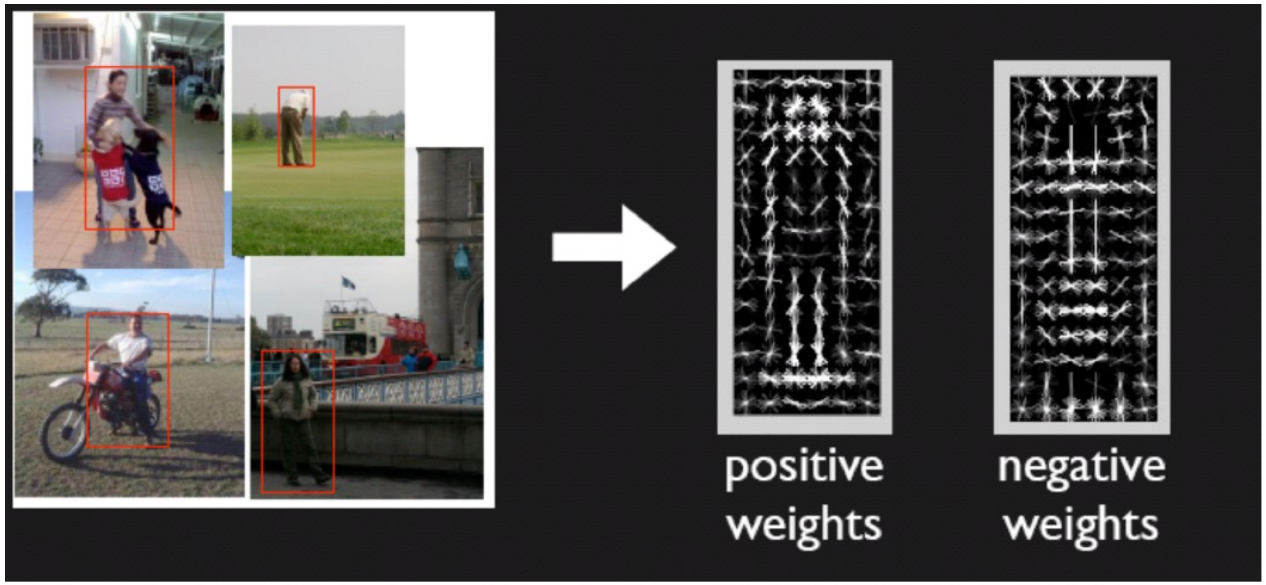


Predominant direction



Histograms of gradients

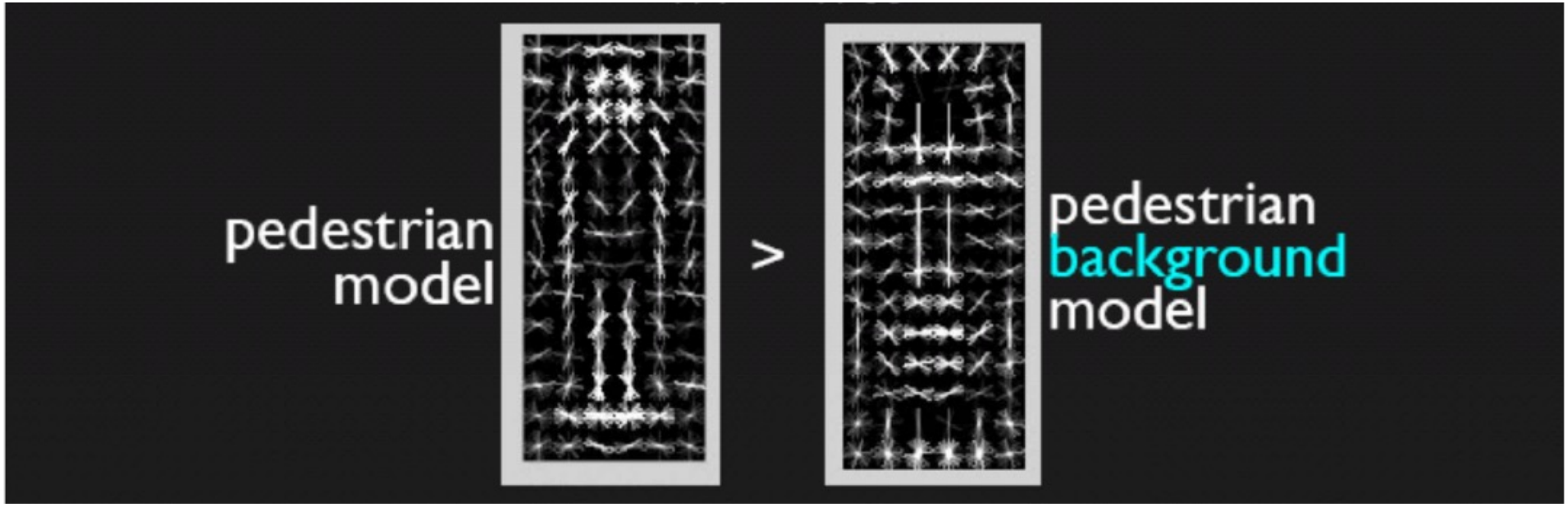
Apply correlation with a pedestrian template



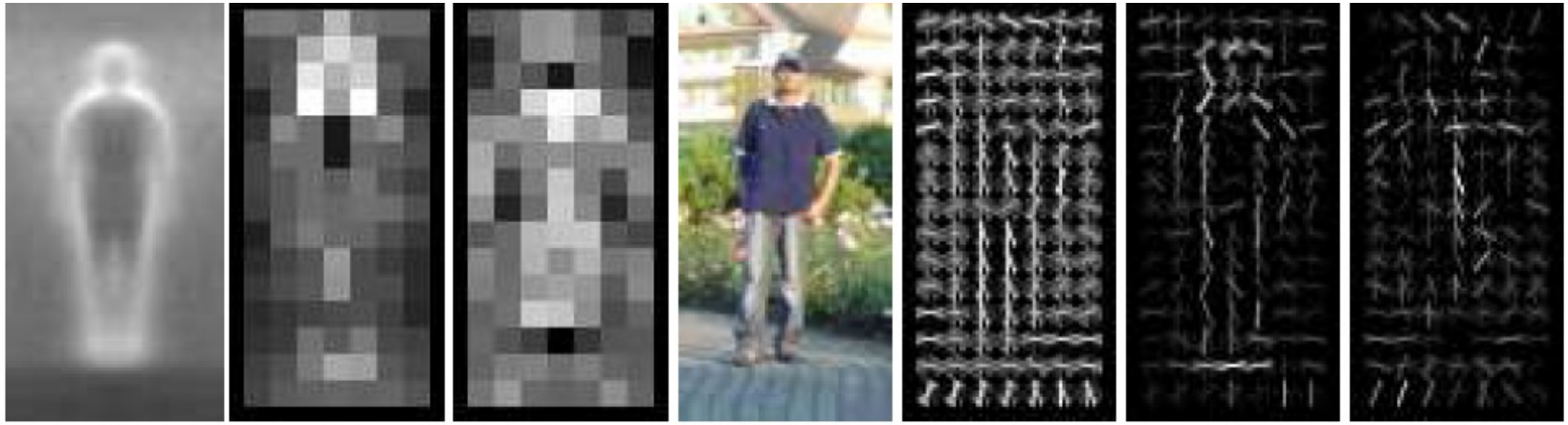
Slide from Deva Ramanan

- Positive weights show edge orientations highly correlated with the pedestrians images.
- Negative weights show edge orientations non correlated with image regions containing a pedestrian (horizontal edges in the region of the legs)

Meaning of negative weights



Template matching based on HOG for pedestrian detection



Descriptor extraction:

(a) Get the average gradient image over query and test examples,
(b) Extract HOGs for the query image,
(c) Extract HOGs for the dataset of images.

HOG-based Retrieval:

(d) For each dataset image, extract the region bottom left and compare to the query HOG
(e) Apply **the sliding window** technique all over the image and compute the HOGs correlation
(f) Apply the maximum over the correlation resulting image.

(g) Decide if the correlation is high enough -> Pedestrian vs. No-pedestrian.

Pedestrian detection results

Today's plan

- Template Matching
- What are and why we need image descriptors?
  - A particular kind of image descriptor (HOG)
- Pedestrian detection with HOG
- Image retrieval and classification

Problem 1: Image Description

Given an image, how can we automatically construct an image description in order to describe and discriminate objects and images (e.g. between 'building' and 'nature' images)?

Which **visual characteristic**s of the image can be used for this goal?

Problem 1: Image Description

Given an image, how can we measure difference between both images: 'building' and 'nature'?

Which **visual characteristic**s of the image can be used for this goal?

Problem 2: Image Retrieval

Given a 'building' image (query), how to retrieve other 'building' images in a database?

Definition: Given an image (query image), the image retrieval consists of sorting the rest of images according to the similarity to the query image.

Given a set of images of buildings, forests, roads, etc., and a query image of a *building* what kind of images do you expect to retrieve first?

Algorithm for image retrieval

Given a 'building' image (query), how to retrieve other 'building' images in a database?

Algorithm:

1. Define the image descriptor
2. Extract the image descriptors of the database images

3. Given a query image, extract its descriptor
4. Sort the database images according to the similarity with the query image.

General approach for the retrieval problems

Let's suppose for now that the descriptor is simply the mean colour....

Building

Building

Nature

Nature

'Building' or 'Nature'?

Algorithm

- Represent each image of the dataset by its descriptor
- Store the descriptors of the samples

R	G	B	
20	30	200	
34	166	111	
12	220	222	
25	244	30	

General approach to retrieval problems

Let's suppose for now that the descriptor is simply the mean colour....


Building


Building


Nature


Nature


'Building' or 'Nature'?

Algorithm

- Represent each image of the dataset by its descriptor
- Store the descriptors of the samples
- Sort according to the similarity to the query descriptor

R	G	B	Order
20	30	200	3
34	166	111	1
12	220	222	4
25	244	30	2
233	**55**	**211**	

Do we explicitly extract what the image represents (building, tree, person)?

How to choose the descriptor?

The **descriptor (or feature vector)** should describe the image in a way that is invariant to all the image changes that are suitable to our application (e.g. color, illumination, noise etc.)





What do all these buildings have in common?

What does distinguish them from images of natural lands?

Image retrieval

Given an image (query), find all similar images in the database.

Image descriptor:

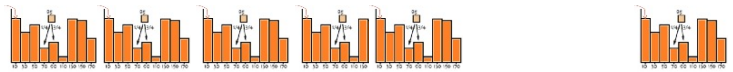
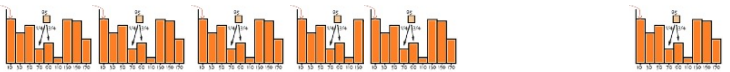


Image descriptor:



K-Nearest Neighbors for retrieval

- The query is an unlabelled vector in our feature space
- Retrieve the k-closest neighbours as the relevant items to a query
  - k is a user-defined constant
  - Database images do not necessarily have labels.

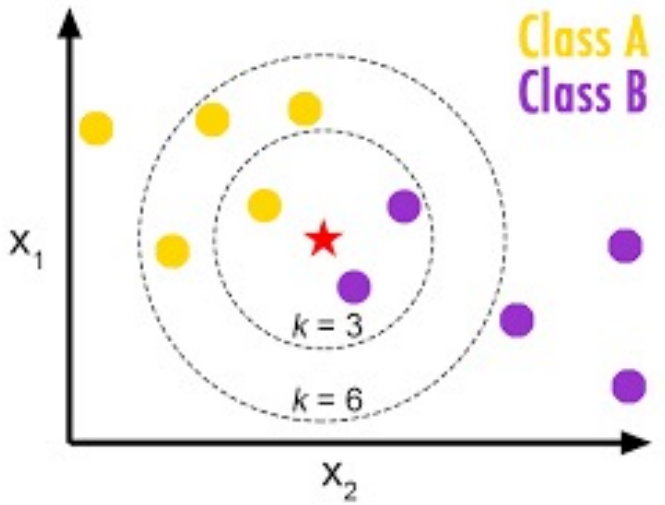
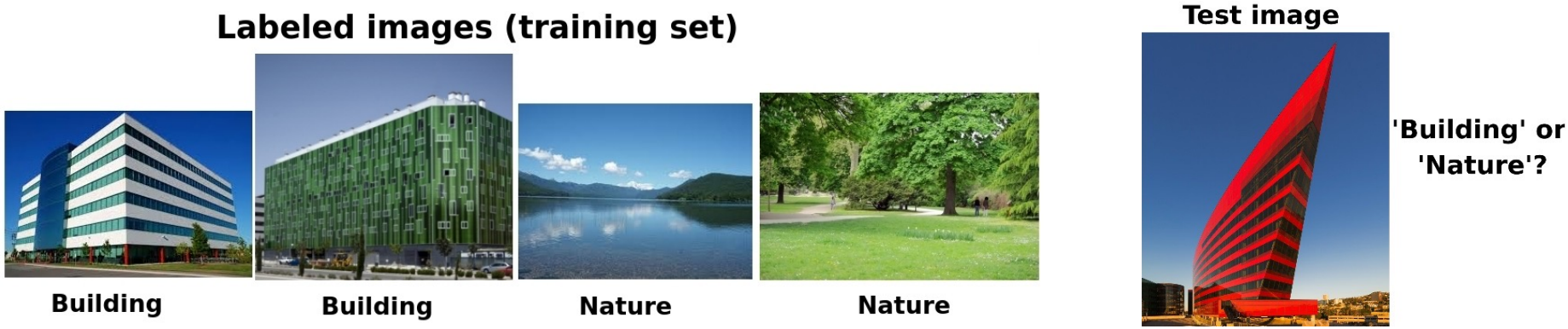


Image descriptors

General approach to classification problems

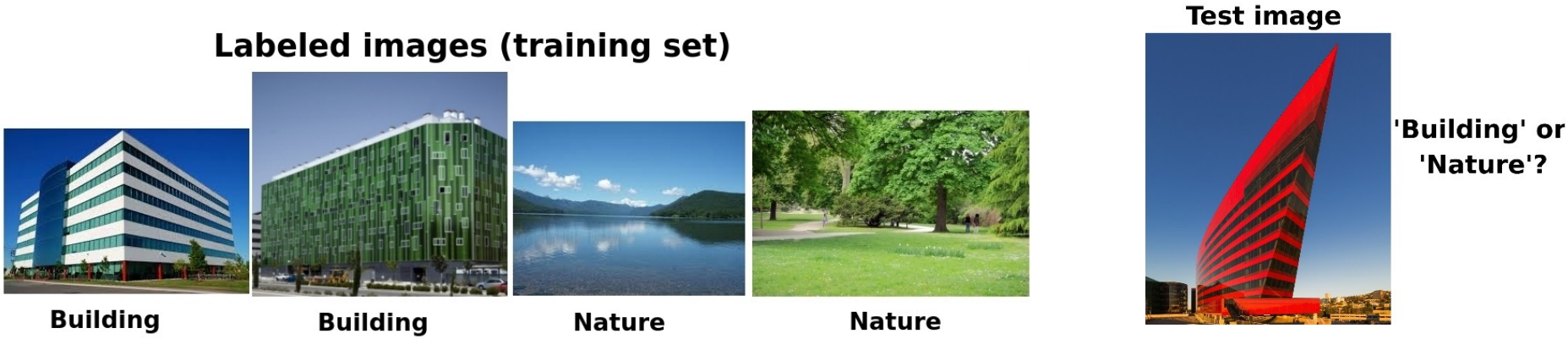


In the classification problem we have classes with their corresponding labels!

Training phase:

- Represent each image of the training set by its descriptor
- Store the descriptors and class labels of the training samples (labelled images)

Algorithm for image classification



Training:

1. Define image descriptors
2. Use training set to extract their descriptors
3. Train a model

Test:

4. Given a test example extract its descriptor
5. Apply the model and compare with the training examples to decide its label

General approach to the classification problem

Let's suppose for now that the descriptor is simply the mean colour....

Training phase:

- Represent each image of the training set by its descriptor
- Store the descriptors and class labels of the training samples (labelled images)

R	G	B	Label
20	30	200	1
34	166	111	1
12	220	222	0
25	244	30	0

- **Model**: if G>200, -> label 1. Otherwise, label 0.

General approach to the classification problem

Let's suppose for now that the descriptor is simply the mean colour....

Labeled images (training set)

Building Building Nature Nature

Test image

'Building' or 'Nature'?

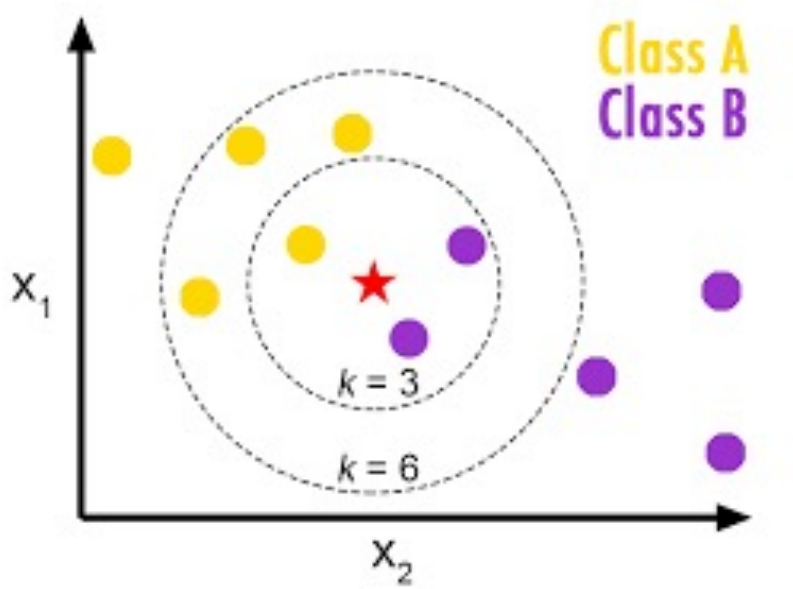
Test phase:

- Compute the descriptor of the test image
- Apply the model/classifier to compare the descriptor of the test image to the descriptors of the training images in order to determine its class membership.

R	G	B	Label
20	30	200	1
34	166	111	1
12	220	222	0
25	244	30	0
233	**55**	**211**	**?**

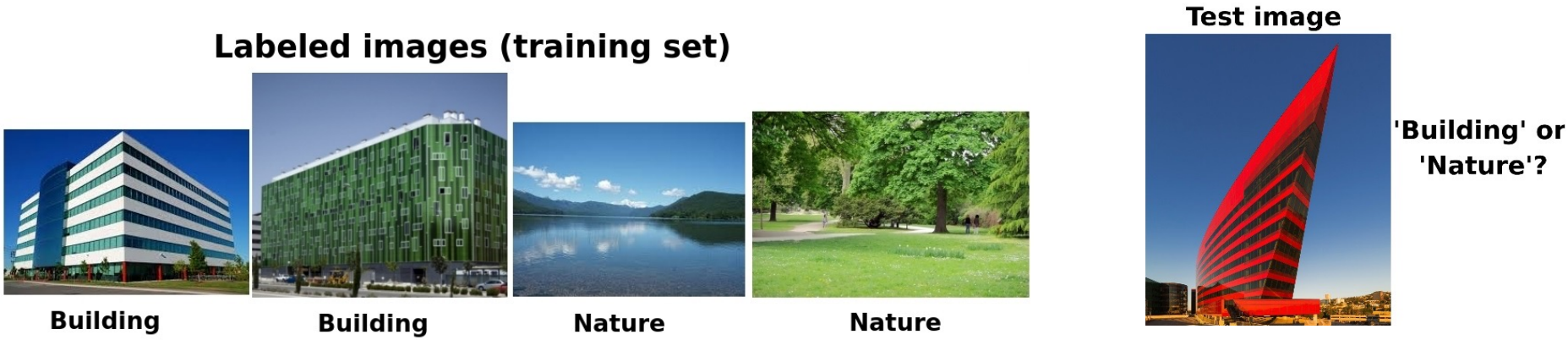
K-Nearest Neighbors for classification

- The feature vector of the image is a point in our feature space
- The image is classified by assigning the label which is most frequent among the k training samples nearest to the test point.
  - k is a user-defined constant (How to choose k?)
  - Can the classification change for different k?



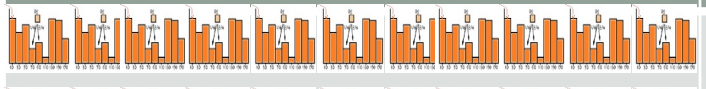
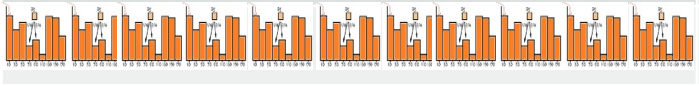
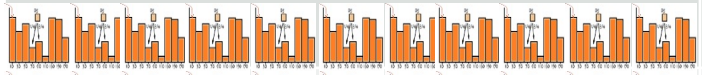
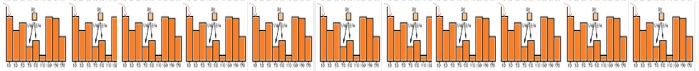
General approach to the classification problem

Can we use HOG as image descriptor?



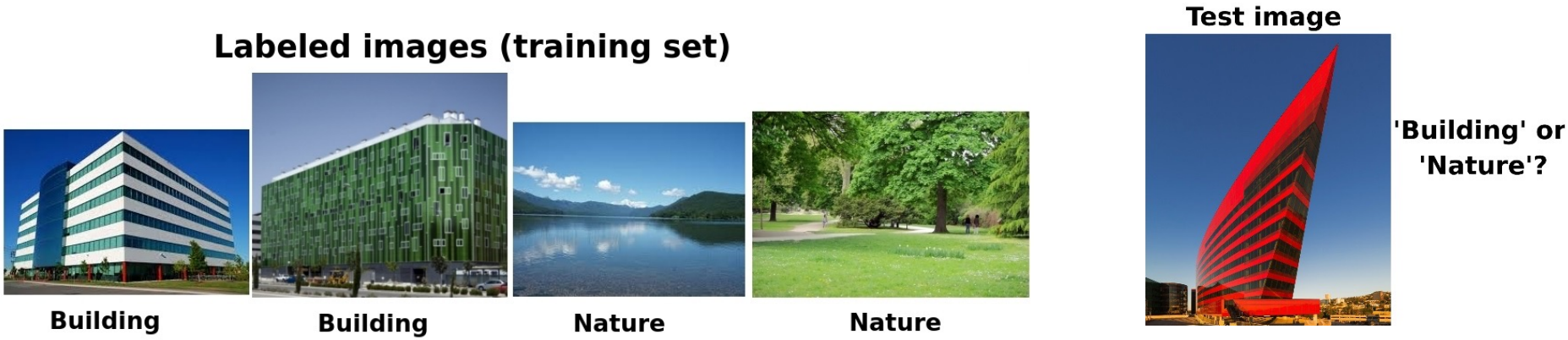
Training phase:

- Represent each image of the training set by the statistics of its HOGs
- Store the descriptors and class labels of the training samples (labelled images)

HOG descriptor	Label
	1
	1
	0
	0

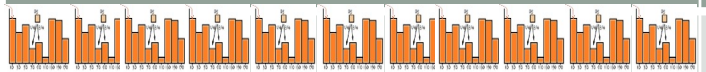
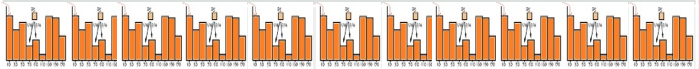
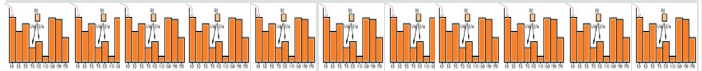
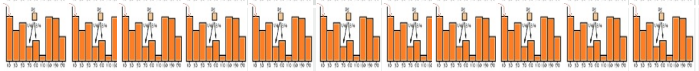
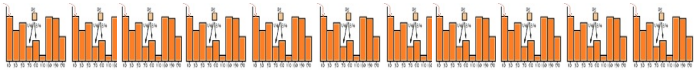
General approach to the classification problem

Can we use HOG as image descriptor?



Test phase:

- Compute the HOG –based descriptor of the test image
- Apply the model/classifier to compare the descriptor of the test image to the descriptors of the training images in order to determine its class membership.

HOG descriptor	Label
	1
	1
	0
	0
	?

HOG image descriptor for object detection, retrieval and classification

- Extract features which are discriminant for your problem
  - each image is represented by a high-dimensional feature vector.
- Use a machine learning algorithm for object detection, retrieval and classification
  - this always requires to first extract features from all images of the training set.
- Make predictions (classification) on the test images to detect the looked for object/class.
  - K-nn
  - There are other classifiers: e.g. Support vector machine classifier, decision trees, etc.

Let’s do some test

- Go to socrative.com
- Room: ComputerVision2122