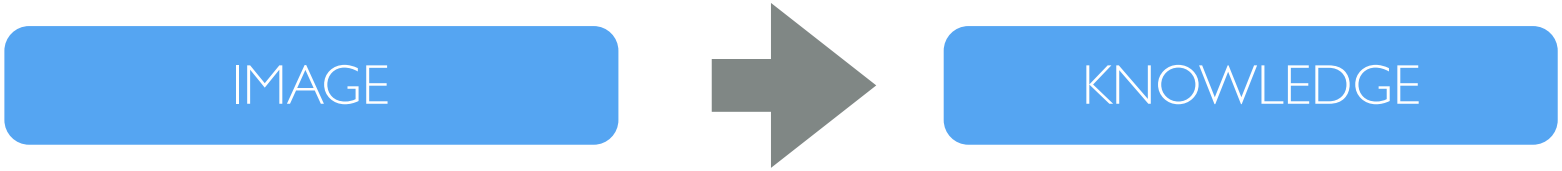
introduction to Computer Vision

Francesca Odone
Department of Computer Science, Bioengineering, Robotics and Systems Engineering
University of Genova

What is Computer Vision about?



- **extracting descriptions of the world from images**
- descriptions of what kind?
  - *qualitative* or *quantitative*
  - *geometric:* shape and position of object or relative distances in the real 3D world
  - *semantic:* what objects do I see?
  - *dynamic:* scene changes, objects velocities, actions, …

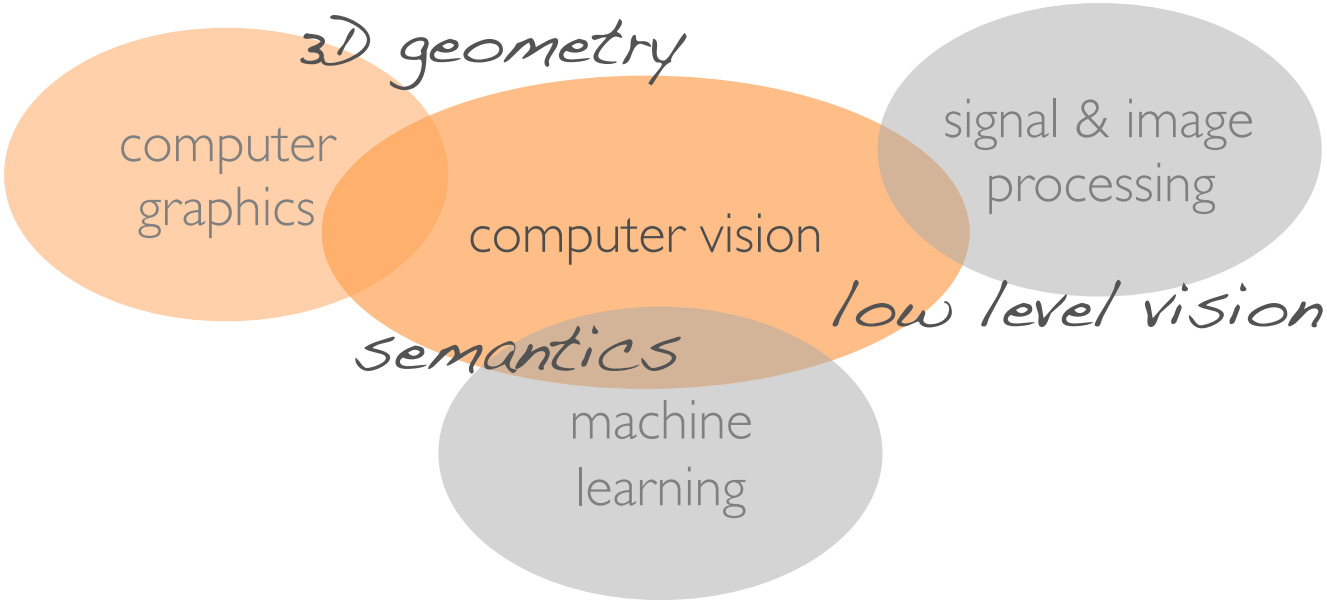
Computer Vision - A brief history

- 1960s - started as a student summer project at MIT
  - the goal of the project was to link a camera to a computer, getting the computer to describe what it saw
- 1970s mainly seen as a part of **artificial intelligence** (emulating **human visual perception**)
  - simplification of the world ("lego-like")
  - pioneering work of David Marr (mainly on) **low level vision**
- 1980s more **sophisticated mathematics**

Computer Vision - A brief history

- 1990s computer vision reaches its maturity
  - **geometric approaches** - optical flow
- 2000s: **machine learning** paves the way to scene understanding
- Today: we are dealing with very large data and **very large datasets** (deep architectures)

Computer Vision - An incomplete view of related disciplines



Outline

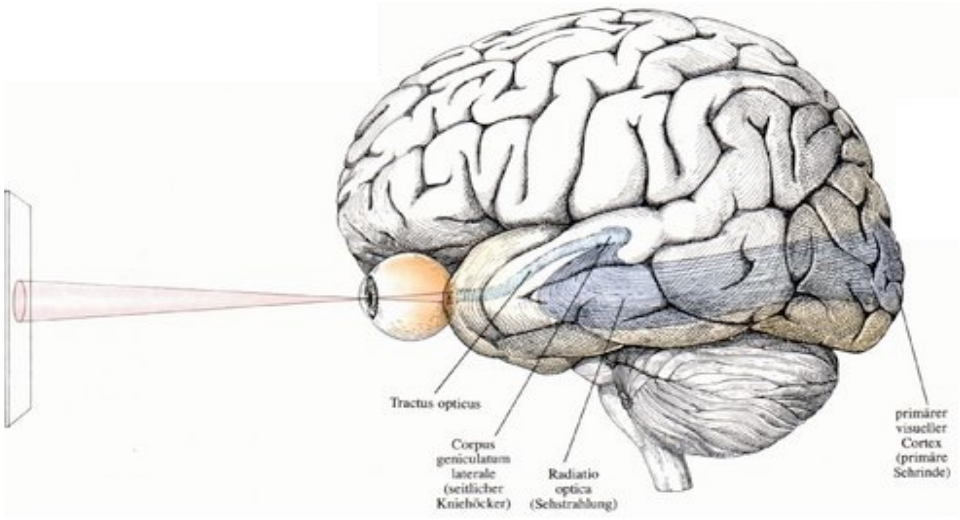
- A few words on human vision
- Digital images and low level elements

- Building blocks: image correlation

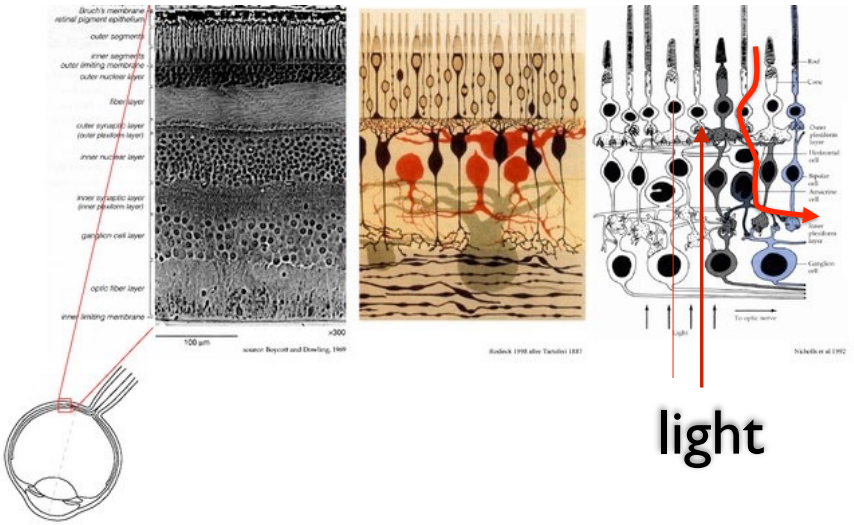
- Depth: computing disparity maps
- Motion: estimating optic flow

- Other ways of analysing motion: motion segmentation (in brief)

The human visual system



Hubel 1985

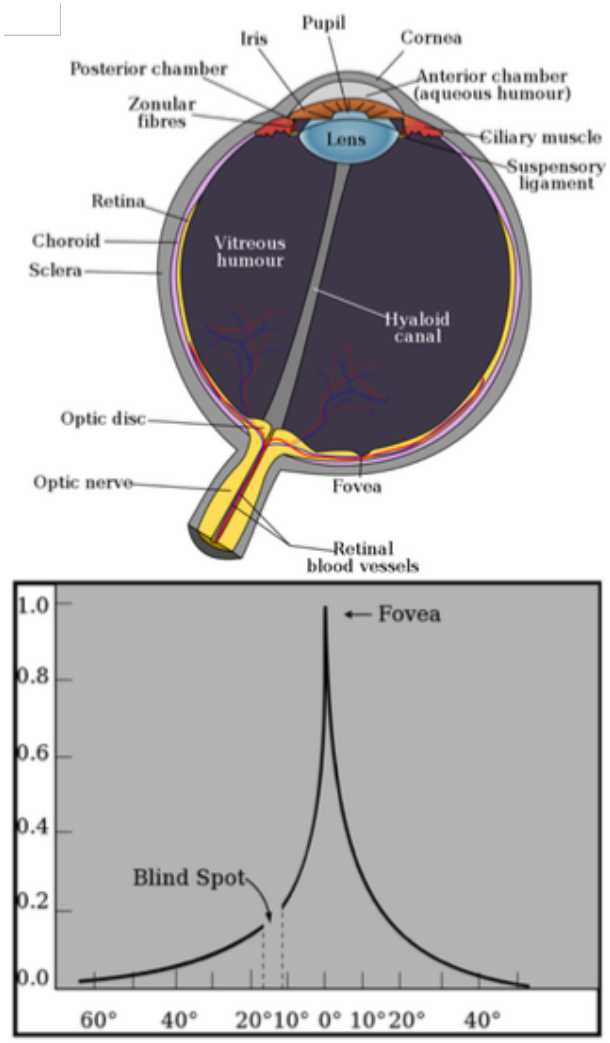


neural signal

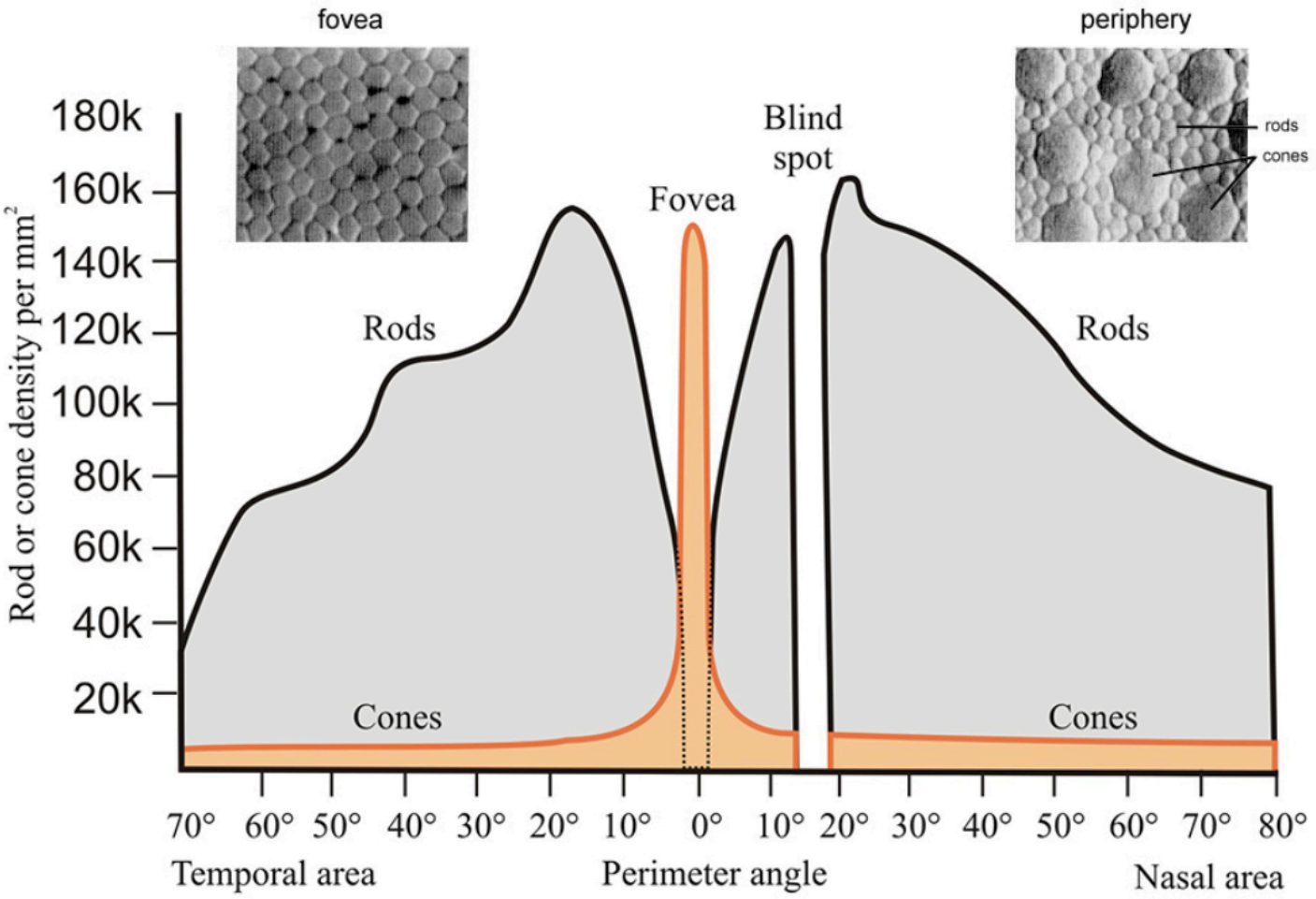
light

The human visual system

- Cones:
  - color vision
  - dense in the center
  - 5-6 millions
- Fovea:
  - central zone of retina
  - only cones, 27 times the density
  - responsible for central vision
- Rods:
  - b/w vision
  - peripheral vision
  - 120 millions

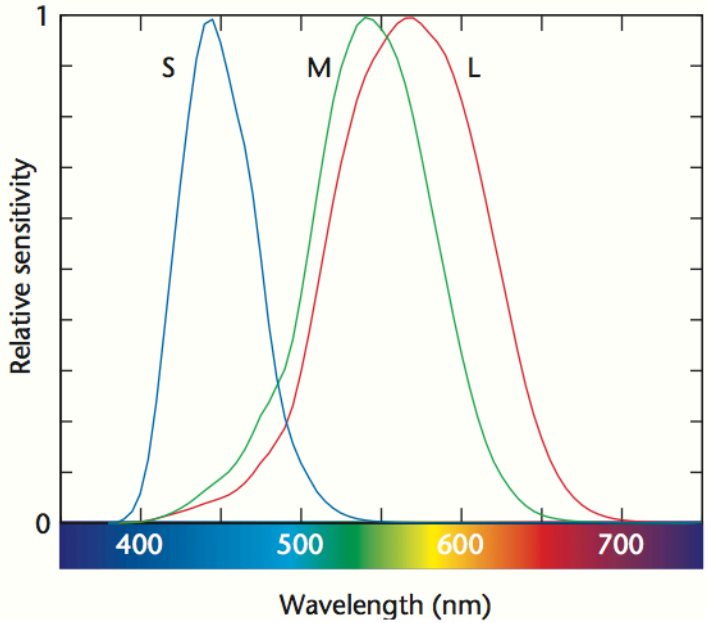


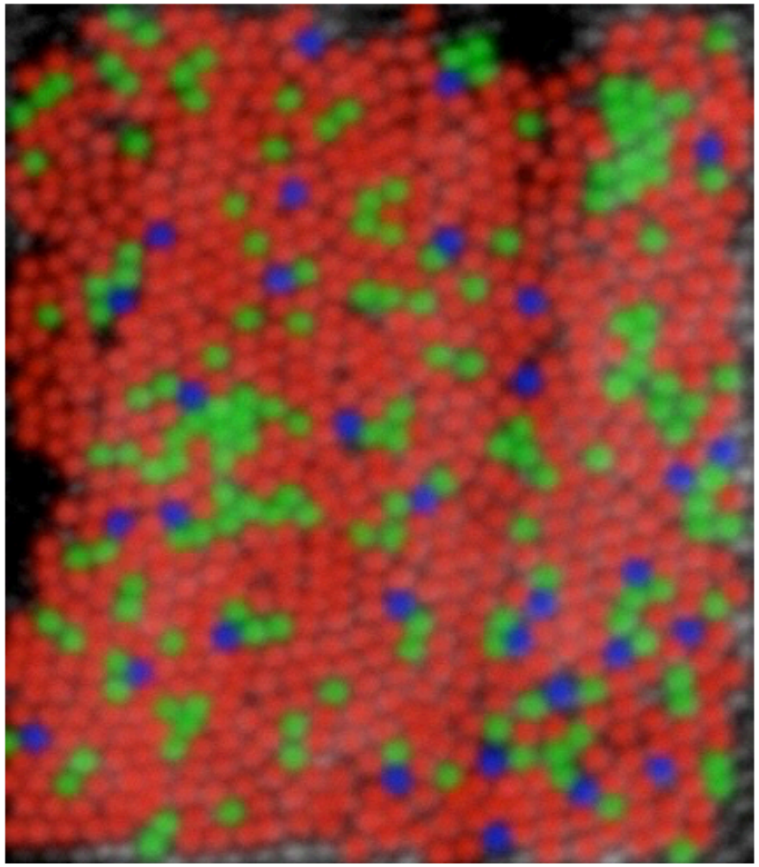
The human visual system



Color vision

- Three different types of cones:
  - present with different densities
  - each type sensitive to a different range of frequencies in the visible spectrum
  - called L, M, S from the zones of spectrum at which they have highest response

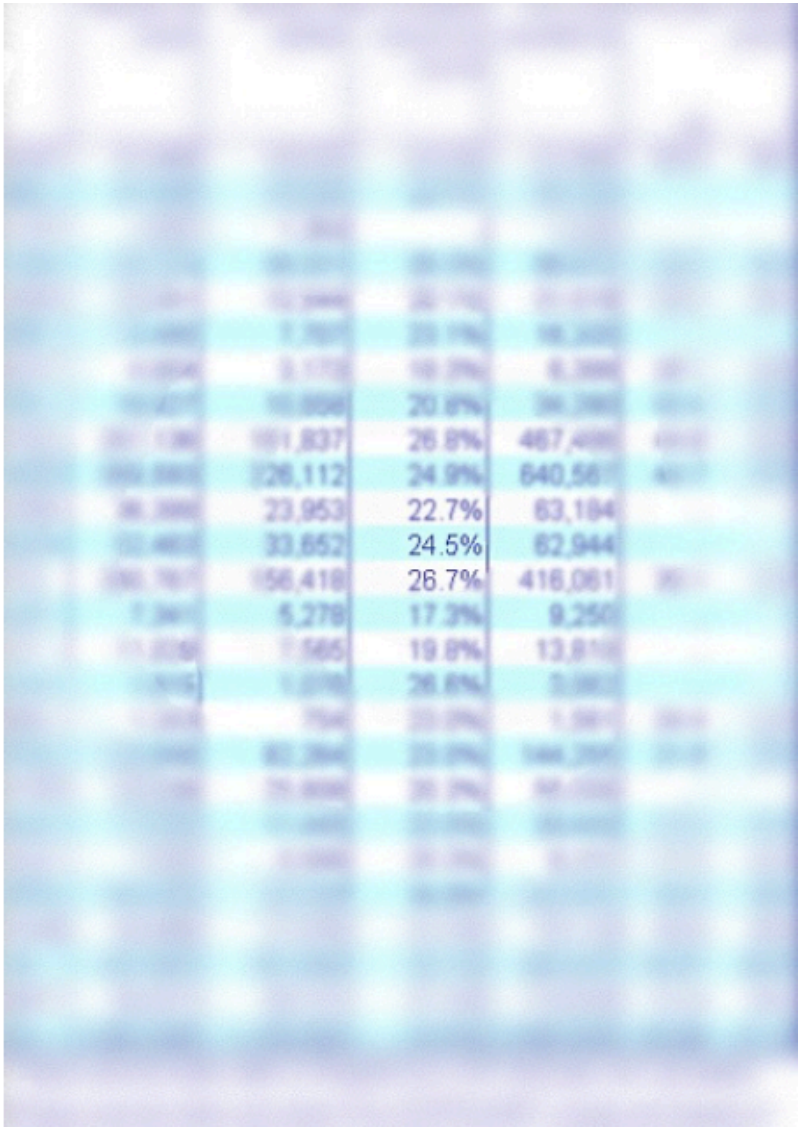




R 63% - G 31% - B 6%

The human visual system

- The eye is not like a camera
- At each instant in time, our vision is sharp only in a very small area around the fovea
- All the rest is badly blurred and vaguely coloured
- Areas outside the fovea provide just contextual information

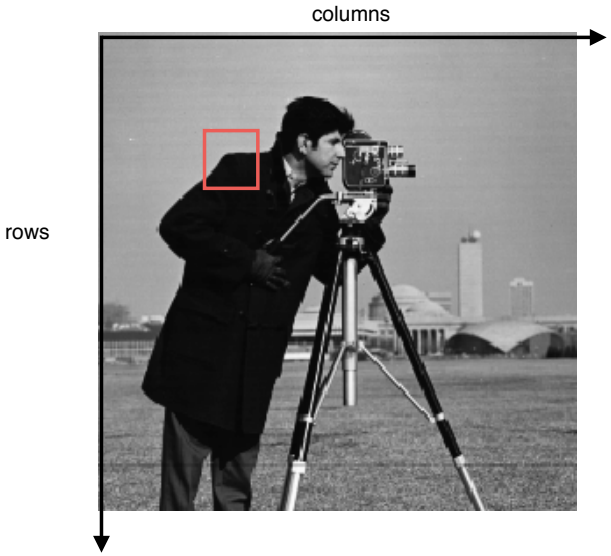


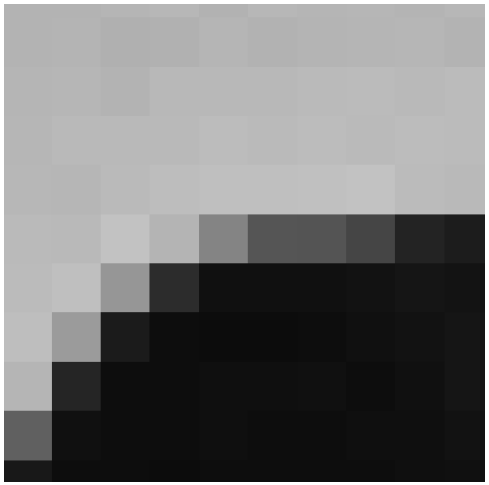
The human visual system

- Vision works as a sequence of *fixations* and *saccades*:
  - Fixation: maintaining gaze on a single location (200-600 ms)
  - Saccade: moving between different locations (20-100 ms)
- Our mental images are a form of integration of several snapshots focused at different spots, as obtained with fixation
- Where do we fixate our gaze?
  - where there's something attractive
  - where we know there's something we want to look at

http://www.cs.ubc.ca/~mikewu/cs547/

A digital image

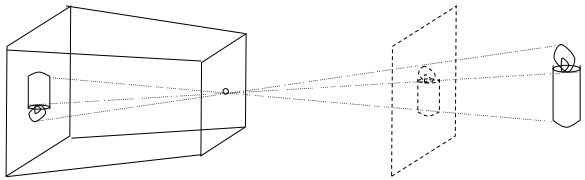


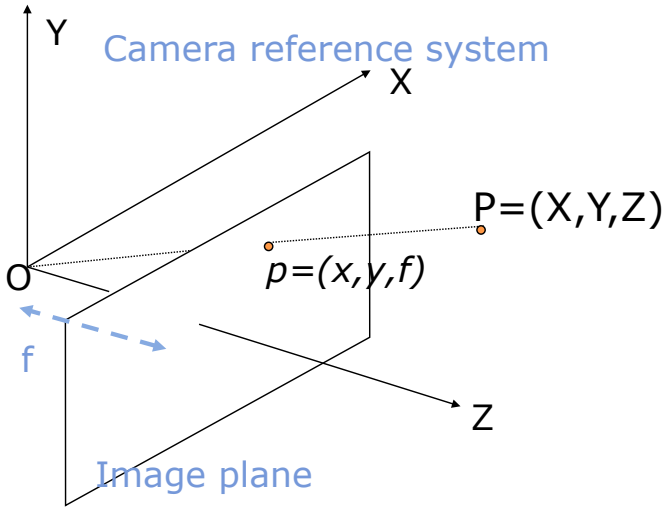


180	183	181	181	179	181	180	180	182	181	183	181	183	184	184	184	184	181	184	183	180
178	178	178	177	182	182	180	177	182	180	183	179	181	181	183	180	181	180	179	180	180
179	178	178	180	177	178	179	179	181	183	178	183	181	182	180	183	181	181	180	181	182
179	179	177	179	177	171	179	180	176	177	181	178	180	181	182	179	183	181	179	177	179
180	179	180	181	180	181	181	182	179	184	184	184	186	187	185	187	182	183	187	186	182
178	179	181	182	180	182	182	185	185	185	188	186	188	186	188	187	190	191	188	186	186
181	178	181	178	186	180	183	182	186	189	191	191	192	194	187	185	178	178	180	184	188
179	177	180	181	180	183	186	185	194	180	132	85	84	69	35	28	25	25	27	29	41
176	177	181	180	183	181	187	191	150	44	16	16	17	18	21	19	19	18	20	19	21
181	181	180	181	181	183	190	155	27	13	12	12	13	16	18	21	18	19	19	19	17
179	179	181	179	186	185	181	37	14	14	15	15	16	14	16	21	19	18	20	19	19
182	180	184	182	183	195	96	16	14	14	15	14	14	15	15	18	20	17	19	18	18
180	180	183	186	191	159	24	13	13	12	13	14	14	14	15	16	20	17	19	19	17
181	183	185	190	185	49	15	16	16	13	12	12	13	14	14	14	19	17	17	18	19
184	184	187	192	104	22	20	17	16	15	12	14	13	15	16	16	18	19	18	18	18
186	189	193	133	24	17	16	16	15	15	16	13	13	15	16	16	16	17	17	16	18
188	192	137	31	16	16	16	17	18	18	15	14	13	14	15	16	17	16	17	16	16
193	126	26	17	17	16	16	16	17	16	16	14	14	13	14	16	16	15	15	16	16
131	24	16	15	16	16	16	16	16	16	17	15	14	12	13	14	15	15	14	16	16
24	17	17	15	16	16	16	17	16	16	17	16	15	15	14	15	15	17	15	15	16
15	15	13	13	15	17	16	15	15	15	14	13	13	13	13	14	14	14	14	15	14

The geometry of image acquisition

At the basis of image acquisition there is a projection from the 3D world to the 2D image





- The pin-hole camera model is a perspective projection
- it is the simplest model of camera and well describes the **geometry** of image formation of a camera system:
  - human vision
  - photo/video cameras

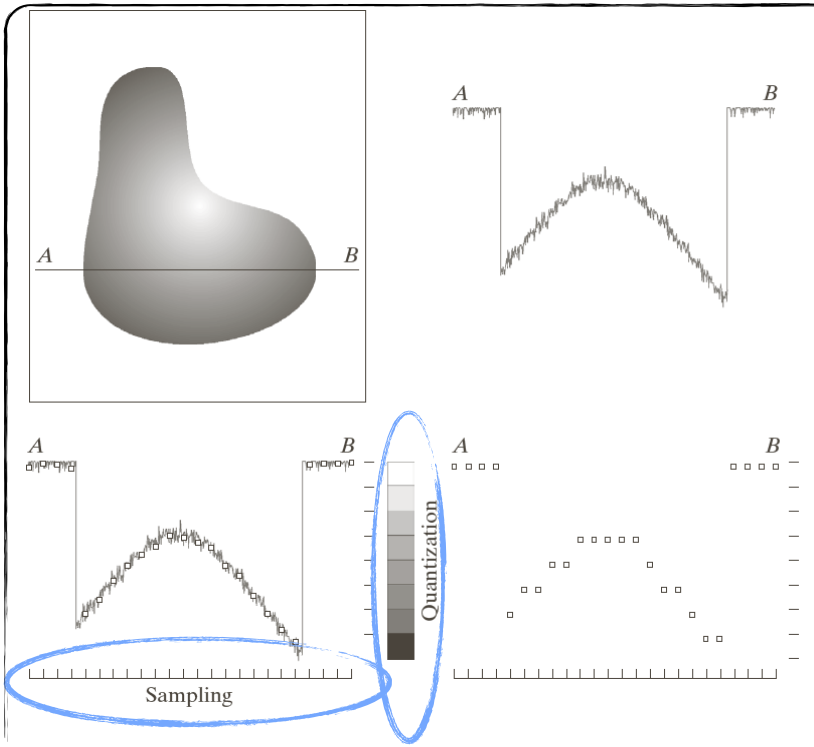
f focal length

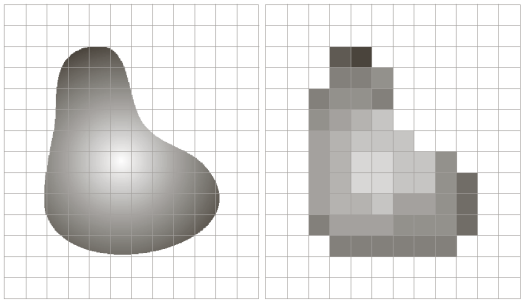
$$x = f\frac{X}{Z}$$

$$y = f\frac{Y}{Z}$$

Sampling and digitalization

during the acquisition process the continuum observations of the world are acquired and stored in digital media



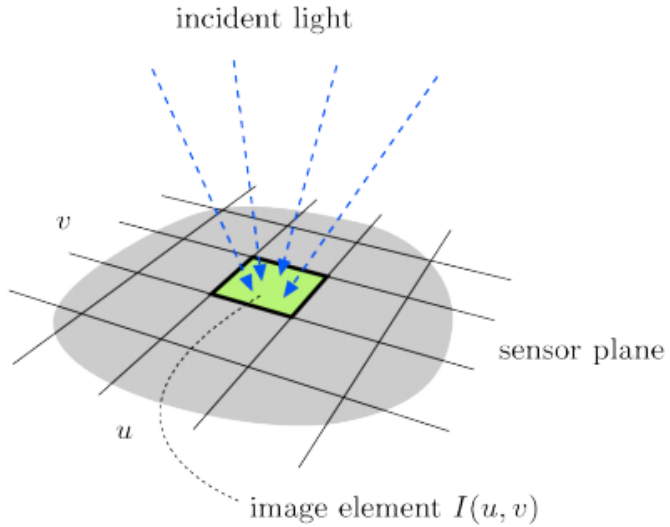


a b

FIGURE 2.17 (a) Continuous image projected onto a sensor array. (b) Result of image sampling and quantization.

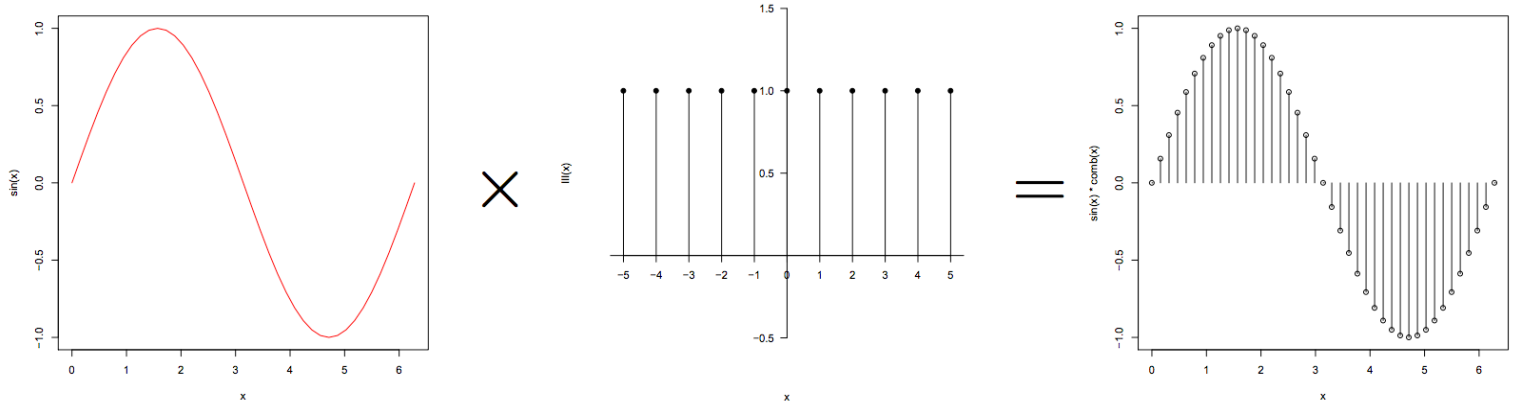
Spatial sampling

- Let us imagine an image to be digitalized is overlaid with a regular grid.
- This grid is referred to as **sampling grid.**
- Each element of the grid will contain a portion (region) of the image. The whole portion will be approximated by a unique (average) value.
- A coarse sampling grid produces an image with fewer details



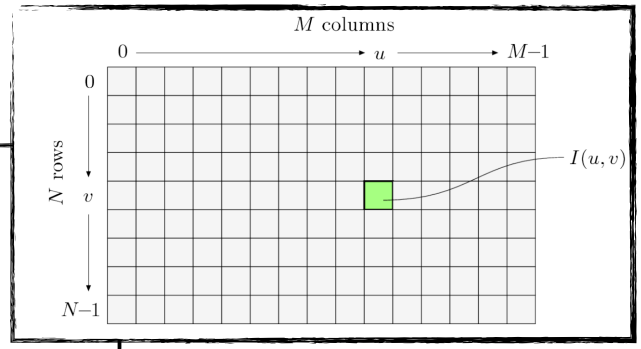
Spatial sampling

- we can think of spatial sampling as a multiplication of a continuous signal with a comb function



the SIZE ...

- The size of the image is given by the number of pixels composing it.
- The size is conventionally expressed by the number of rows times the number of columns of the image matrix, eg 640 x 480.



.... and the RESOLUTION

- An image with a fixed size (in *pixels*) can be visualized at different sizes (in *mm*) on a **support** (paper, monitor, …)
- The *visualization size* is controlled by the **resolution.**
- The resolution depends on the size of the image *and* the size of the support
- It is measured in dots/cm or, more frequently, dots/inches (dpi).

The resolution describes how *dense* are the elements on the support.

Pixels content - information

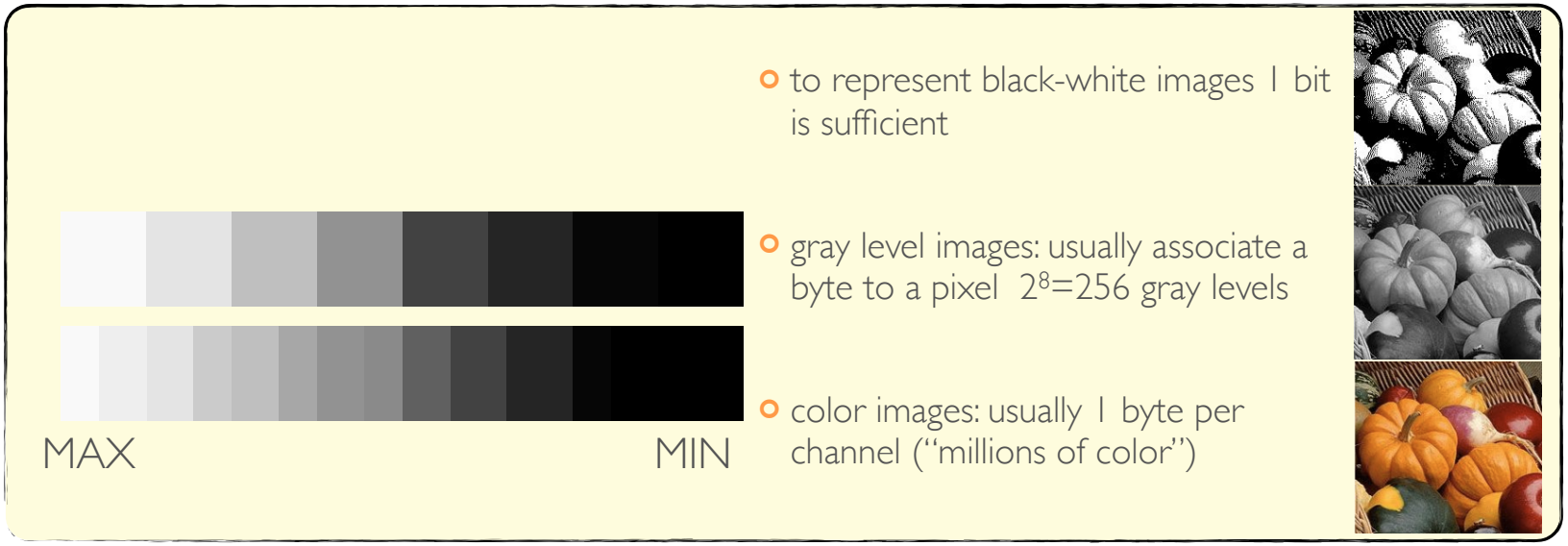
- Pixels contents depend on the image type
  - Gray level pictorial digital images ("black and white photos"): *intensity*
  - Color pictorial digital images: *color* (modeled as triplets, eg RGB)
  - Range images: *depth information*
  - Medical images: *radiations absorbance level*
  - Termal images: *heat*
  - ....

Pixels Content - Dynamic Range

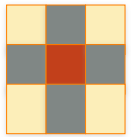
- Total number of distinctive values occurring in the image

this is related to the quantization process...

- it is limited by the number of bit per pixel we may want to use
- it is also limited by the physical dynamic range of the sensor



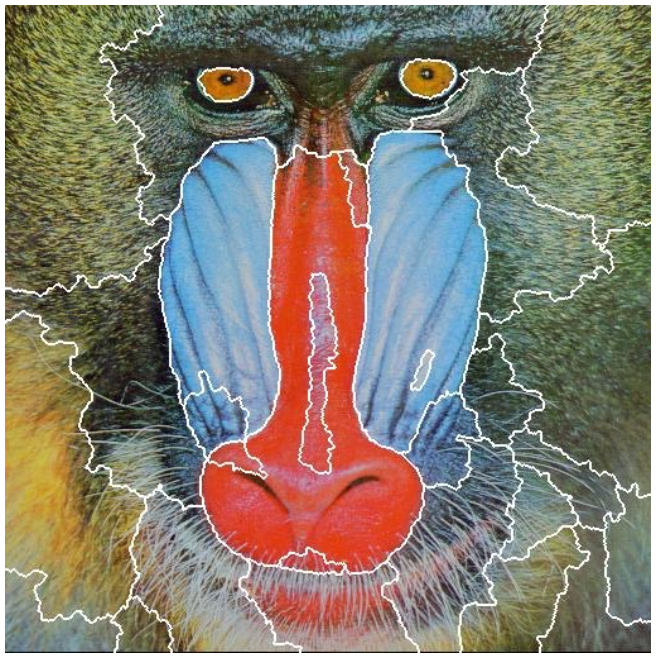
Spatial correlation - pixel neighbourhoods



- A pixel *p* at coordinates *(i,j)* has four horizontal and vertical **neighbors** at coordinates *(*i-1,j) (i+1,j) (i,j-1) (i,j+1)
  - This set is called **4-neighborhood** $N_4(p)$
- The pixel also has four diagonal neighbors:
  (i-1,j-1) (i+1,j-1) (i+1,j-1) (i+1,j+1)
  - The 8 points together form a **8-neighborhood** $N_8(p)$

Connected components

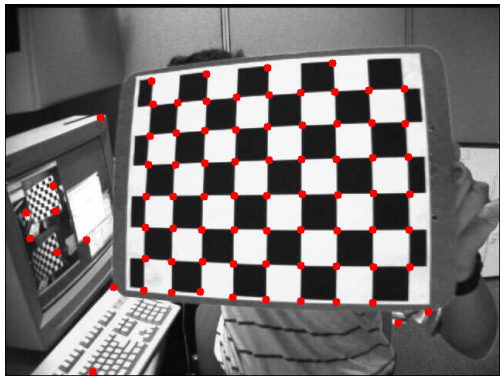
- groups of connected (aggregated) pixels with common properties
- the properties could be a similar color, texture, or motion pattern, …



Keypoints

- "Special" pixels in the image associated with some important properties

EDGES - sharp signal variations



CORNERS - points where the signal varies in at least two directions

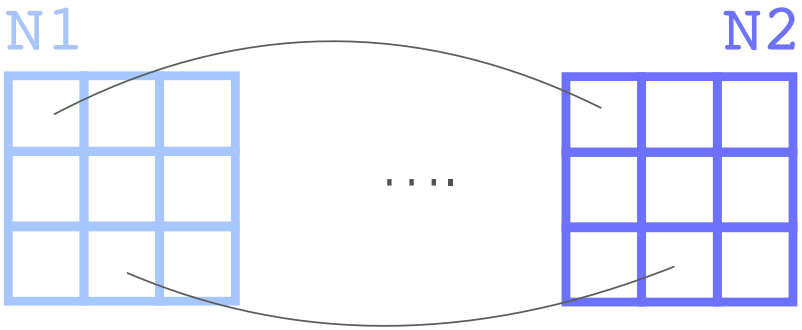
- We have *feature detection algorithms* to find them automatically

From parts to objects

Similarity between image patches

- Let N1 and N2 be two square image patches (or neighbourhoods) of size WxW
  - Goal: to measure how similar they are by means of an appropriate function
  - How: we introduce the so-called correlation methods

$$\phi(N1, N2)$$



Here W = 3

Similarity between image patches

– SSD: Sum of Squared Differences

$$\phi_{SSD}(N1, N2) = - \sum_{k,l=-\frac{W}{2}}^{\frac{W}{2}} (N1(k,l) - N2(k,l))^2$$

– NCC: Normalized Cross Correlation

$$\phi_{NCC}(N1, N2) = - \sum_{k,l=-\frac{W}{2}}^{\frac{W}{2}} \frac{(N1(k,l) - \mu_1)(N2(k,l) - \mu_2)}{W^2 \sigma_1 \sigma_2}$$

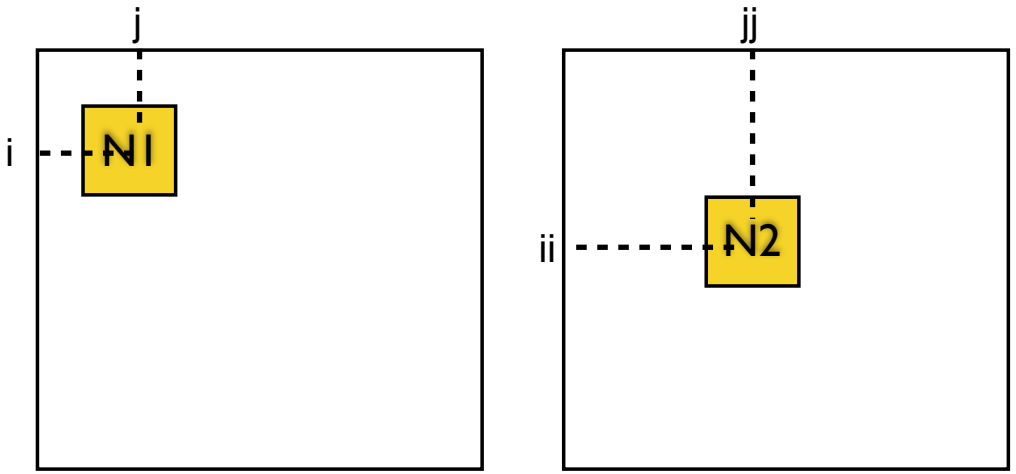
$$\mu_i = \frac{1}{W} \sum_{k,l=1}^{W} Ni(k,l)$$

$$\sigma_i = \sqrt{\frac{1}{W} \sum_{k,l=1}^{W} (Ni(k,l) - \mu_i)^2} \quad \text{with } i = 1, 2$$

NCC: values in the range [-1, 1]

Relative position w.r.t the image

- usually patch N1 belongs to an image I_1, that is, it is a neighbour of a pixel at position (i,j)
- similarly N2 will belongs to an image I_2 as a neighbourhood of a pixel (ii,jj)
- therefore we need to translate the patches to their appropriate position



$$\phi_{NCC}(N1_{i,j}, N2_{ii,jj}) = -\sum_{k,l=-\frac{W}{2}}^{\frac{W}{2}} \frac{(N1(i+k, J+l) - \mu_1)(N2(ii+k, jj+l) - \mu_2)}{W^2 \sigma_1 \sigma_2}$$

The correspondence problem

correlation methods

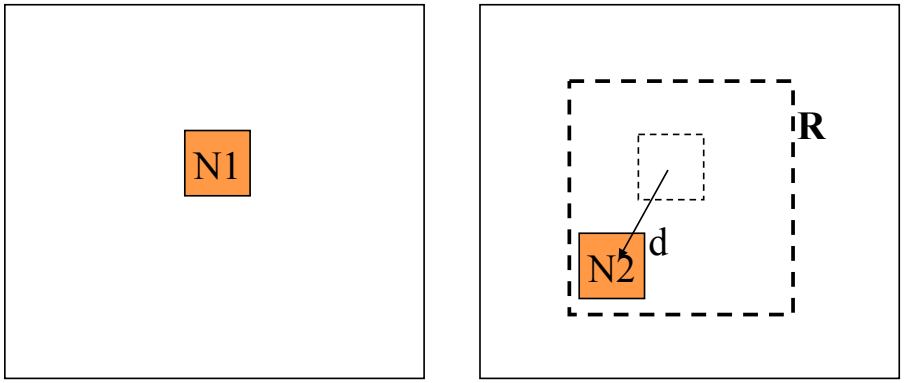
- Now we have all the ingredients to address (in a simple way) an important problem in computer vision: finding correspondences

- Correspondence problem: given **two images**, estimate which parts of the first image correspond to which parts of the second image
  - that is, which parts are projection of the same scene element

- The problem can be addressed as a *dense* or a *sparse* estimation
  - dense correspondence: *for each image pixel*
  - sparse correspondence: *for a set of well defined points*

Where to look for the corresponding feature

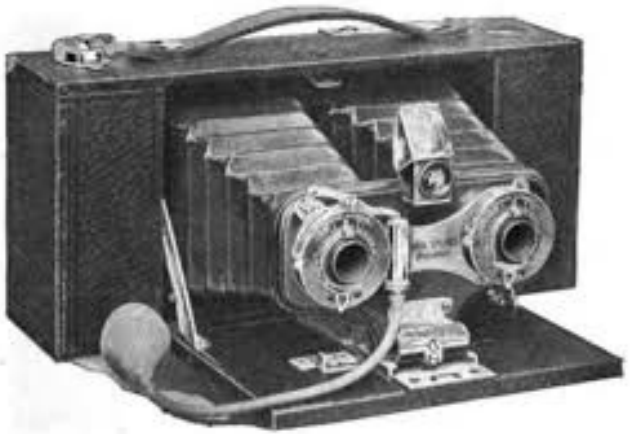
if we have a prior on the corresponding feature position (region **R**)

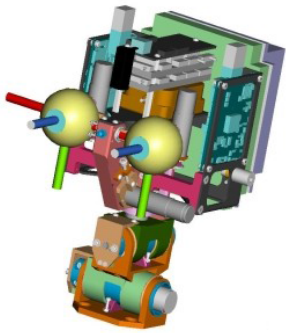


if we don't have a prior on a possible search region we should try all possibilities and this becomes unfeasible as the number of elements grow

Stereopsis

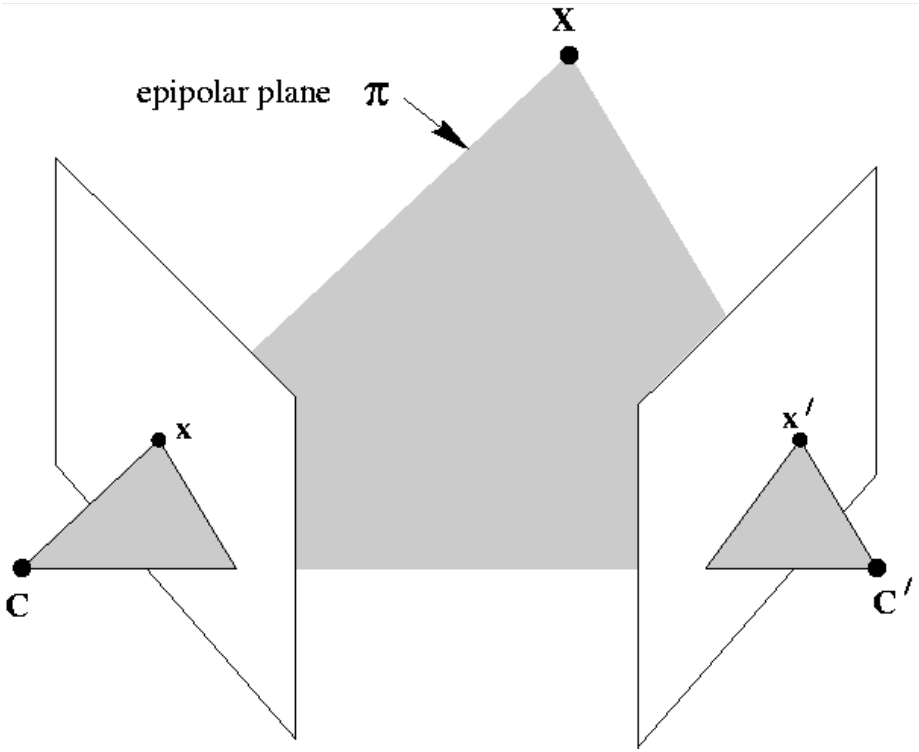
- we refer to stereo vision as the problem of inferring 3D information (structure and distances) from two or more images taken from different viewpoints





Epipolar geometry

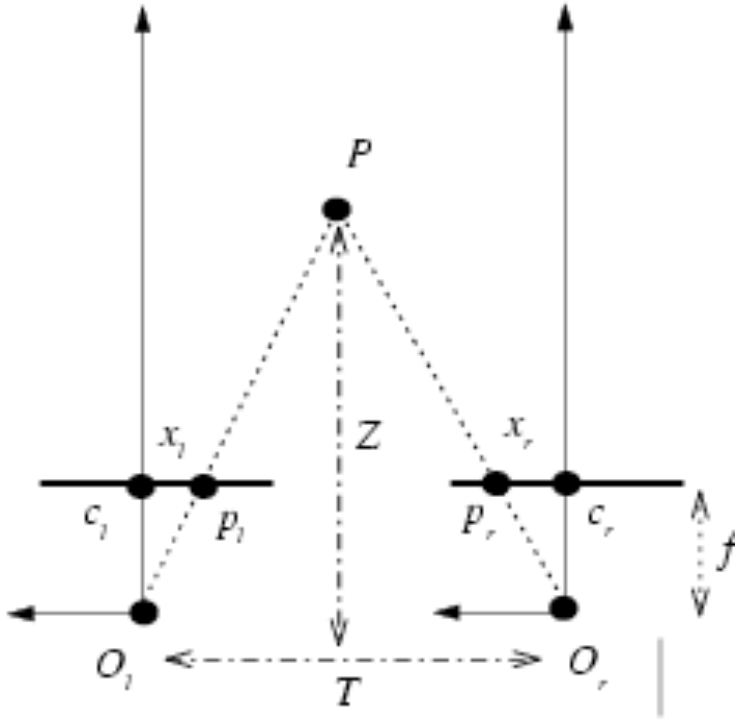
- the geometry of a stereo-system is called epipolar geometry
- it provides a geometrical prior to the algorithms



corresponding points lie on lines (called the epipolar lines)

Stereopsis: depth and disparity

- *Disparity d* is the relative distance between corresponding points (on the image plane)
- *Depth Z* is the distance from a 3D point to the viewing system
- Depth is inversely proportional to disparity



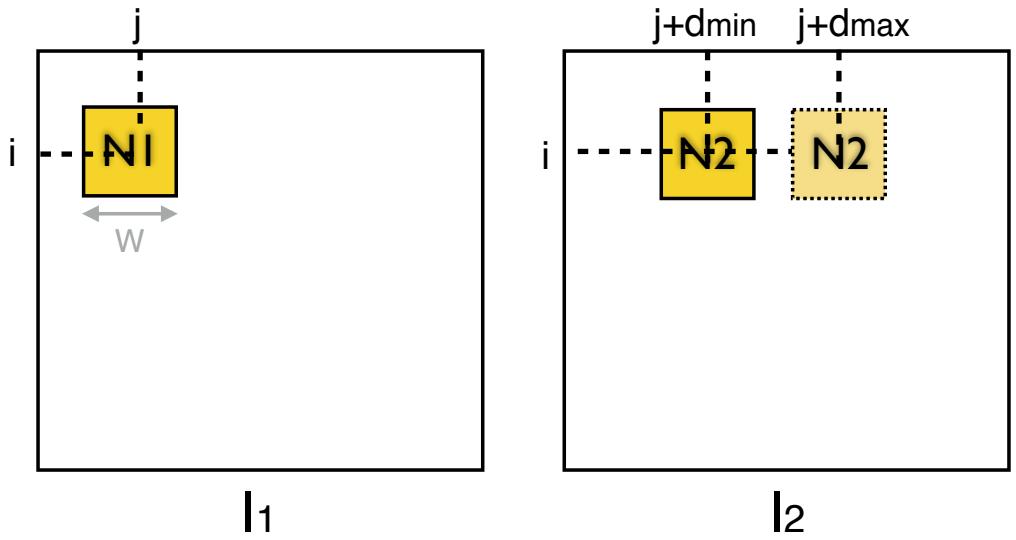
$$Z = \frac{fT}{x_r - x_l} = \frac{fT}{d}$$

Dense stereo correspondences

- we assume we have two *rectified images*, where conjugate points lie on corresponding scanlines of the image ("rows")
- our goal is to obtain a **disparity map** giving the relative displacement for each pixel

- assuming a fixation point at infinity disparity is proportional to the inverse of the distance
- in a standard color coding bright areas correspond to high disparities (closer objects)

Dense correspondences: algorithm sketch



- input:
  - a stereo pair of **rectified** images I_l and I_r
  - size of a correlation window W
  - a search range [d_{min}, d_{max}]

- for each pixel *(i,j)* in I_l
  - for each disparity d in the search range
    - estimate the similarity

$$c(d) = \phi(N1(i,j), N2(i,j+d))$$

    - the disparity of the pixel is

$$\overline{d} = \text{argmax}_{d \in [d_{min}, d_{max}]}\{c(d)\}$$

Dense correspondences: left-right consistency

- correspondences are made more difficult by occlusions (points with no counterpart on the other image)
- let us compute
  - D_{lr}: disparity map from I_l to I_r
  - D_{rl}: disparity map from I_r to I_l
- then $D(i,j)=d$ iff $D_{lr}(i,j)= -D_{rl}(i,j+d)= d$

dense correspondences

Motion perception

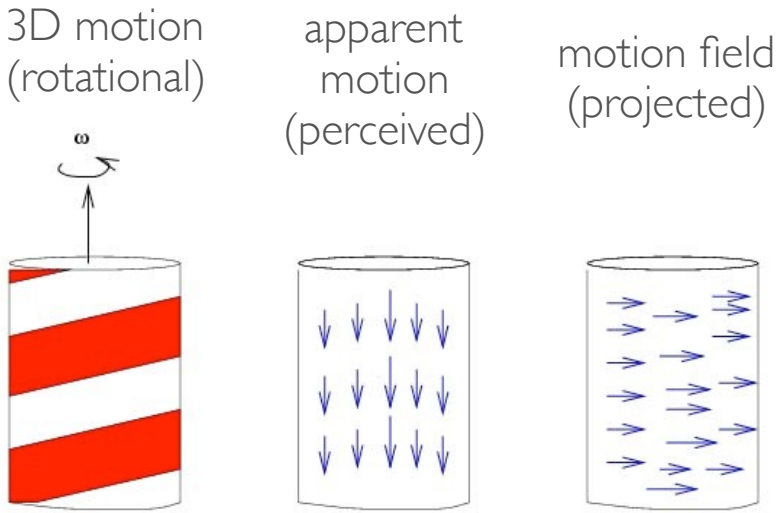
- We are observing a scene with **one** camera acquiring a set of images "close in time"
- **image sequence:** series of N images, or *frames*, acquired at discrete time instants

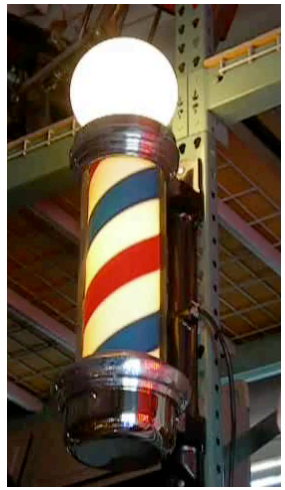
fixed time interval (small)

$$t_k = t_0 + k\Delta t$$

motion field and its estimate

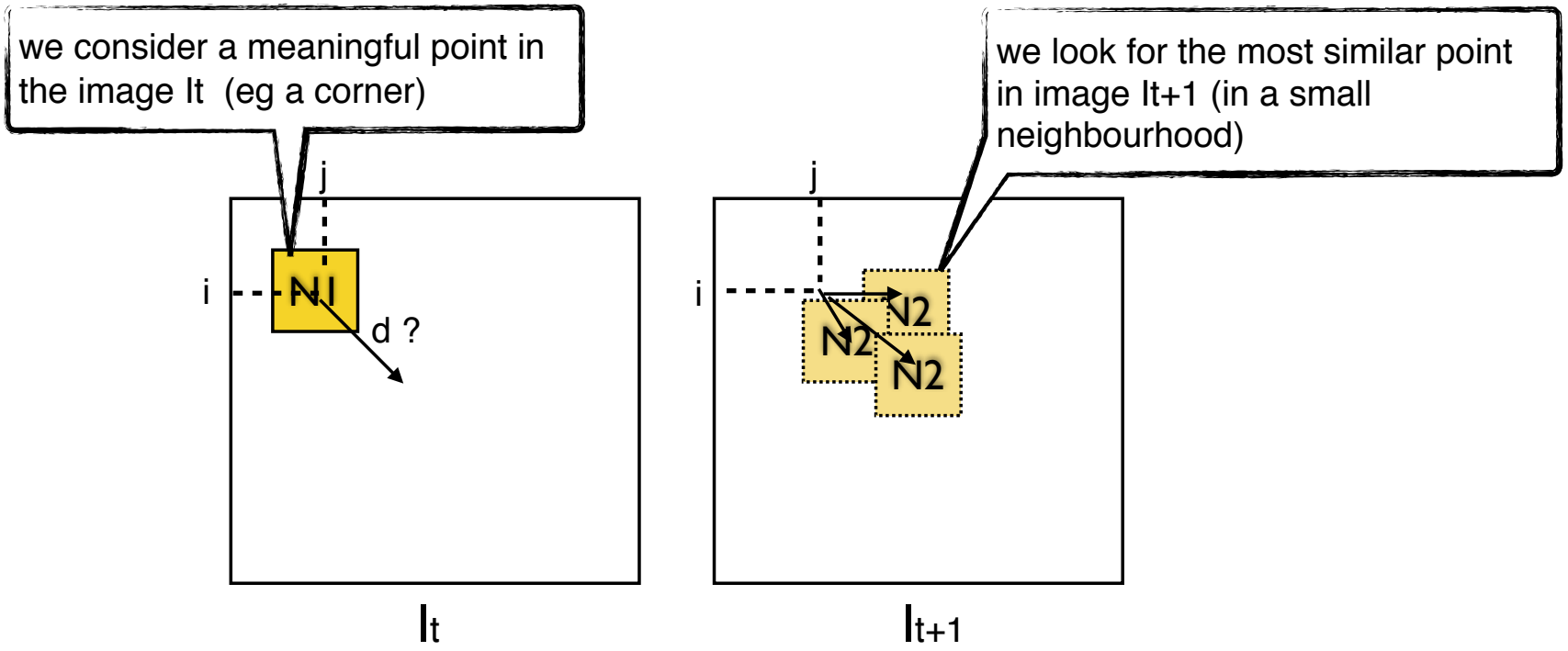
- we may estimate the motion field from images, but what we estimate will be related to the *apparent* motion
- what is the motion field of an object moving in a dark room? or of a uniform object on a similar background?





Correlation-based motion estimation

- Two different images of the same scene, this time acquired by the same camera at adjacent temporal instants
- Prior: in this case we may assume the displacement *d* to be very small



A classical derivative-based algorithm - Lucas Kanade

- We start from an assumption on the *image brightness constancy*

$$\frac{dI}{dt} = 0$$

$$\frac{d(I(x,y,t))}{dt} = \frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0$$

$$(\nabla I)^\top \mathbf{u} + I_t = 0$$

image brightness constancy equation

A classical derivative-based algorithm - Lucas Kanade

- The optical flow is a vector field subject to the constraint

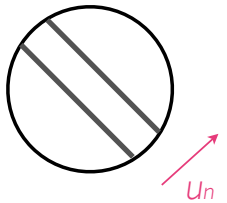
$$(\nabla I)^\top \mathbf{u} + I_t = 0$$

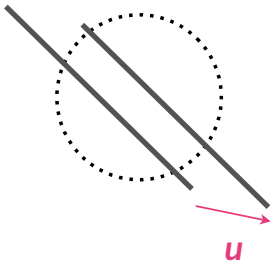
- Notice that **one** constraint is not enough to compute the optical flow (**2** unknowns)

The aperture problem

- the image brightness constancy equation allows us to determine the optical flow component parallel to the spatial image gradient
- Analytically

$$u_n = \frac{(\nabla I)^\top \mathbf{u}}{||\nabla I||} = \frac{-I_t}{||\nabla I||}$$





A classical derivative-based algorithm - Lucas Kanade

- many algorithms start from the idea of adding constraints to the underdetermined system obtained by the brightness constancy equation
- we will see a simple way of doing so: the Lucas-Kanade algorithm
  - assumption: **u** is constant in a small neighbourhood of a point

A classical derivative-based algorithm - Lucas Kanade

- the assumption allows us to obtain a system of equations with one equation for each point in the neighbourhood

$$(\nabla I(\mathbf{x}_i, t))^\top \mathbf{u} + I_t(\mathbf{x}_i, t) = 0 \quad \mathbf{x}_i \in N$$

- we then obtain a linear system **Au=b** with

$$A = \begin{bmatrix} \nabla I(\mathbf{x}_1, t)^\top \\ \nabla I(\mathbf{x}_2, t)^\top \\ \cdot \\ \cdot \\ \cdot \\ \nabla I(\mathbf{x}_m, t)^\top \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} -I_t(\mathbf{x}_1, t) \\ -I_t(\mathbf{x}_2, t) \\ \cdot \\ \cdot \\ \cdot \\ -I_t(\mathbf{x}_m, t) \end{bmatrix}$$

m elements in the *N* neighbour.

A classical derivative-based algorithm - Lucas Kanade

- The linear system may be solved with the pseudo-inverse

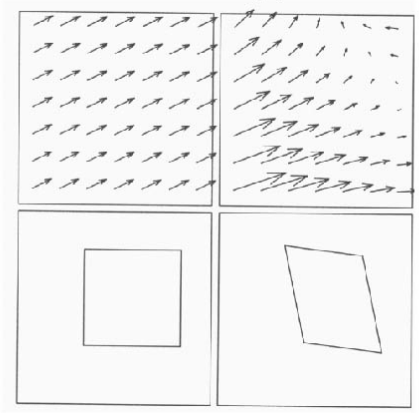
$$\mathbf{u} = A^{\dagger}\mathbf{b} \quad \text{with} \quad A^{\dagger} = (A^{\top}A)^{-1}A^{\top}$$

- notice that the inversion of the matrix will be ill-posed if the matrix is not full rank
- the matrix is full rank in the proximity of corners (points who do not suffer from the aperture problem)
  this is why often times Lucas Kanade is implemented as a *sparse* algorithm (after a corner detection stage)

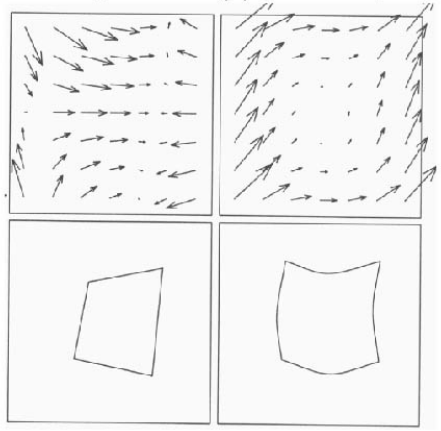
other parametric models

- the hypothesis of a locally constant (translational) optical flow can be extended with more complex models (affine, projective, quadratic, ...)

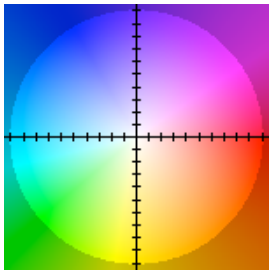
(a) Modello traslazionale (b) modello affine



(c) modello proiettivo (d) modello quadratico



Optic flow - example

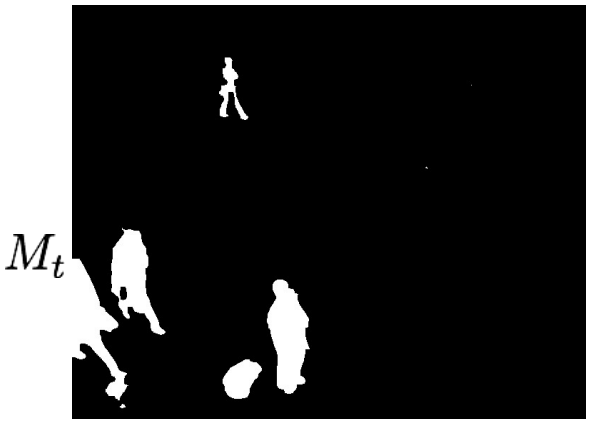


MAIN DIRECTIONS
color coding

motion segmentation

- A different problem of motion analysis is motion segmentation
- **motion segmentation:** what regions of the image plane correspond to different moving objects?
- in the case the camera is still, motion segmentation is called **change detection**

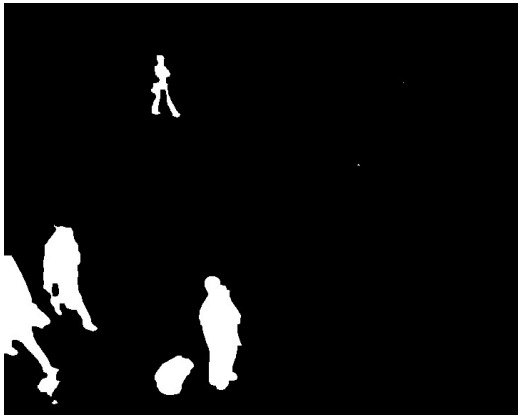
motion segmentation: change detection



- Motion segmentation aims at identifying image regions undergoing a uniform 2D motion field
- Change detection: given a sequence of images taken by a fixed camera, find the regions of the image, if any, corresponding to the different moving objects
- Pixel classification: still or moving?

motion segmentation: change detection

I_t



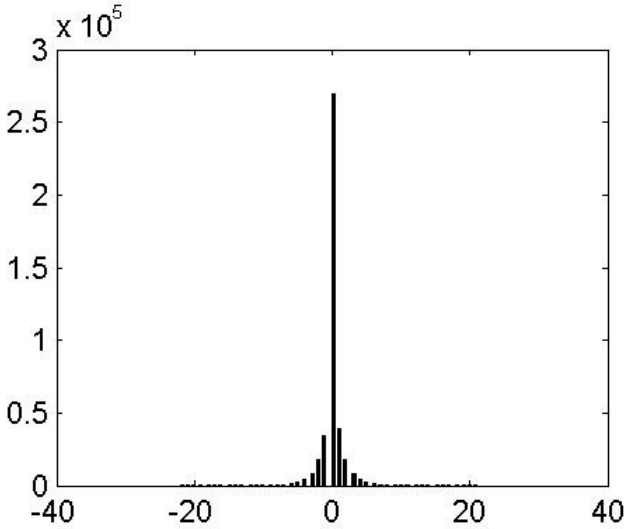
M_t

- in the case the camera is still motion segmentation can be implemented as a difference operation
- assuming we have a reference image

$$M_t(x,y) = \begin{cases} 1 & \text{if } |I_t(x,y) - I_{ref}(x,y)| > \tau \\ 0 & \text{otherwise} \end{cases}$$

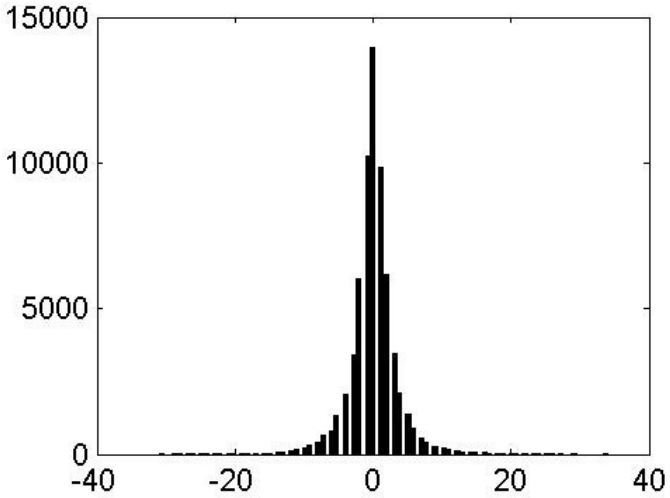
- the thresold must be chosen considering a trade-off between false positives and false negatives

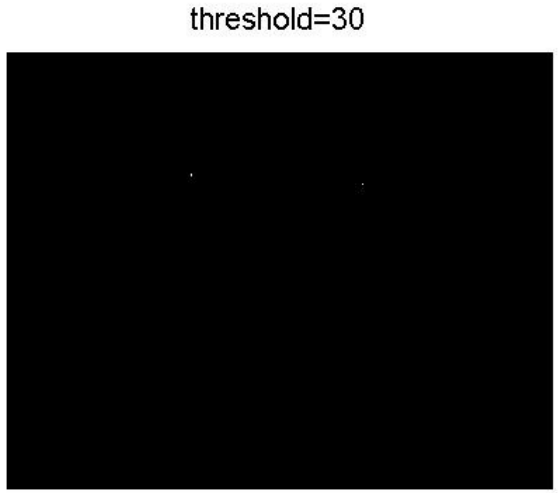
change detection



threshold=10

change detection





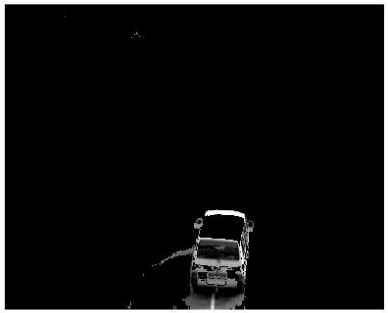
change detection: what is a reference image?

- the simplest way of detecting moving objects is to compare consecutive (or close) frames

frame 1

frame 2

|I1-I2|>th



Artifacts
Difficult to detect slow motion
Noisy localization

change detection: what is a reference image?

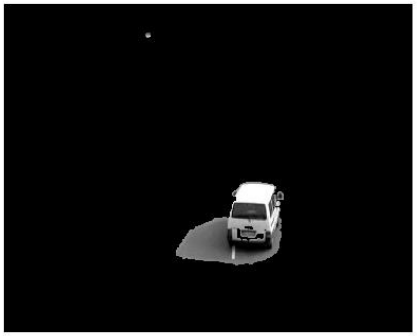
- the reference image or, more in general the *background model*, is a "picture" of the empty scene, containing all the parts which are not moving

sfondo

frame

|I-B|>th

after a cleaning procedure



change detection: what is a reference image?

- the reference image or, more in general the *background model*, is a "picture" of the empty scene, containing all the parts which are not moving
- the simplest way of computing it is to average the first N frames (assuming that at the beginning the scene is stationary)

$$I_{ref} = B = \frac{1}{N} \sum_{t=1}^{N} I_t$$

Background modeling: a running average

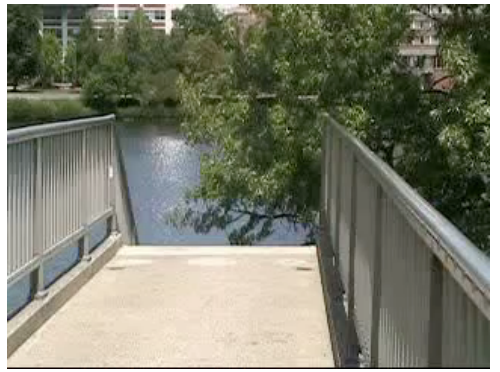
$$B_t(i,j) = \begin{cases} \alpha B_{t-1}(i,j) + (1-\alpha) I_t & \text{if } (i,j) \text{ is classified as static} \\ B_{t-1}(i,j) & \text{otherwise} \end{cases}$$

notice: the background model is updated at each frame

- it is quite robust to moving objects in the scene
- it incorporates stable changes (at a speed which is proportional to α)
- it is simple and computationally efficient

CONS

- it does not deal with repetitive (and uninteresting) motion



motion segmentation: moving cameras

- We need to consider the presence of a dominant motion (the ego motion)
- A possible approach is to first obtain a rough estimate of the ego motion and then compensate it to generate a synthetic datum *as if* the camera were still
- Much harder and computationally intensive (as an alternative we often look for semantic info: people detection, car detection, …)

- change detection may be applied during fixation periods (eye/camera is still) but in this case we cannot use a reference background
- for small eye movements we can eliminate ego motion by thresholding and then cluster optical flow values

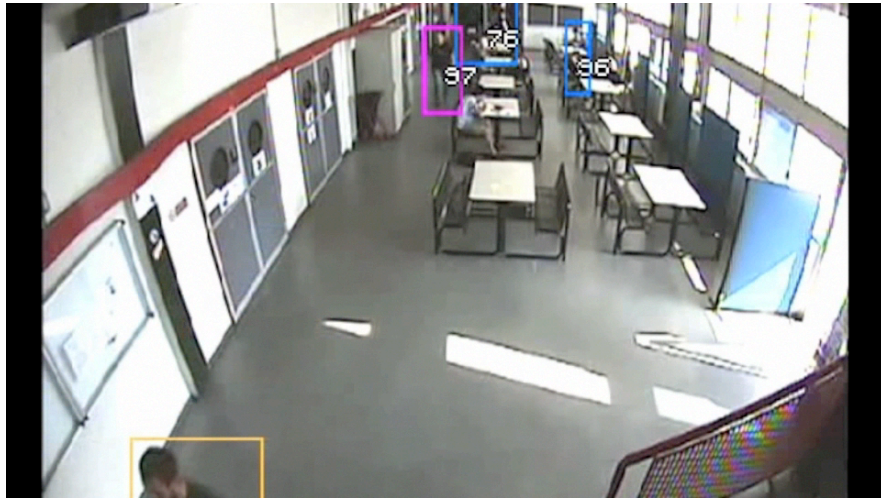
blob detection & object tracking

- After a segmentation procedure we obtain larger portions of coherent pixels

  pixel ⟶ blob (connected component of moving pixels)

- Blob: a region of the image associated with a coherent information, it can be described through its size, shape, color, texture, …

- After motion segmentation, we can perform blob similarity between blobs detected at time *t* and *t+1*



Wrapping up

- We have introduced a selection of *basic* computer vision algorithms

- They can be seen as building blocks for
  - human-robot interation: locating interacting agents
  - understanding the surrounding environment
  - locating interesting (closest disparity) points to control fixation
  - …

- To bridge a semantic gap between pixels and semantic meaning we need to know more (see next classes on machine learning)