UniGe | MaLGa

Dealing with more than one image

Francesca Odone francesca.odone@unige.it

UniGe | MaLGa

Useful concepts from previous classes

- Family of geometric transformations
- Local features and feature matching

What you will learn

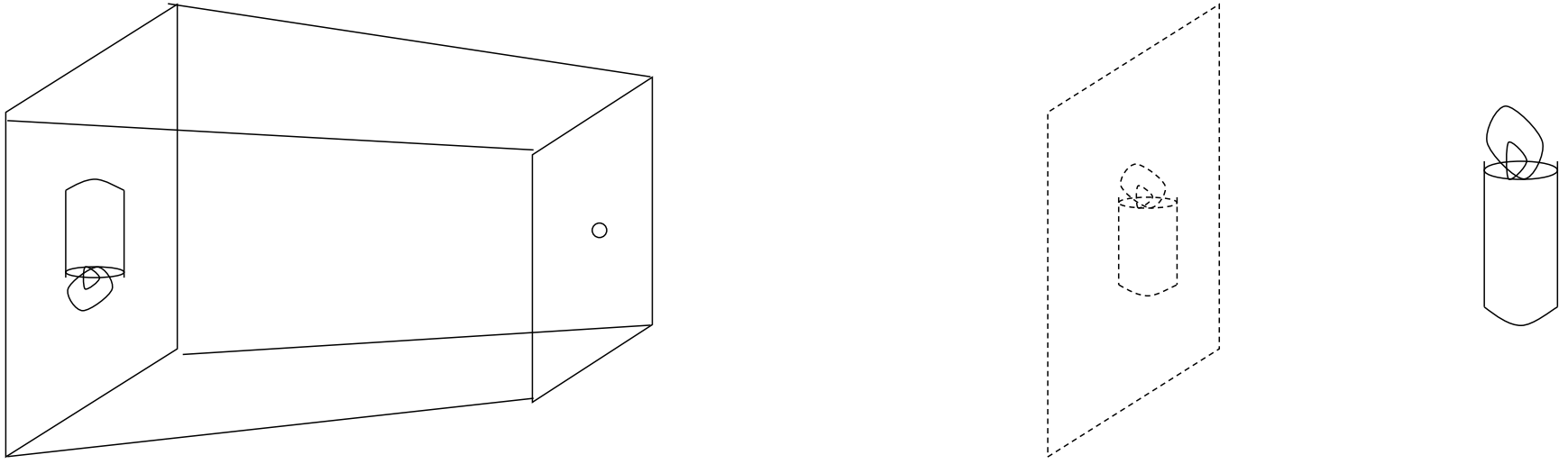
- The geometry of image formation
- Stereopsis and 3D reconstruction from images
- Motion analysis

UniGe | MaLGa

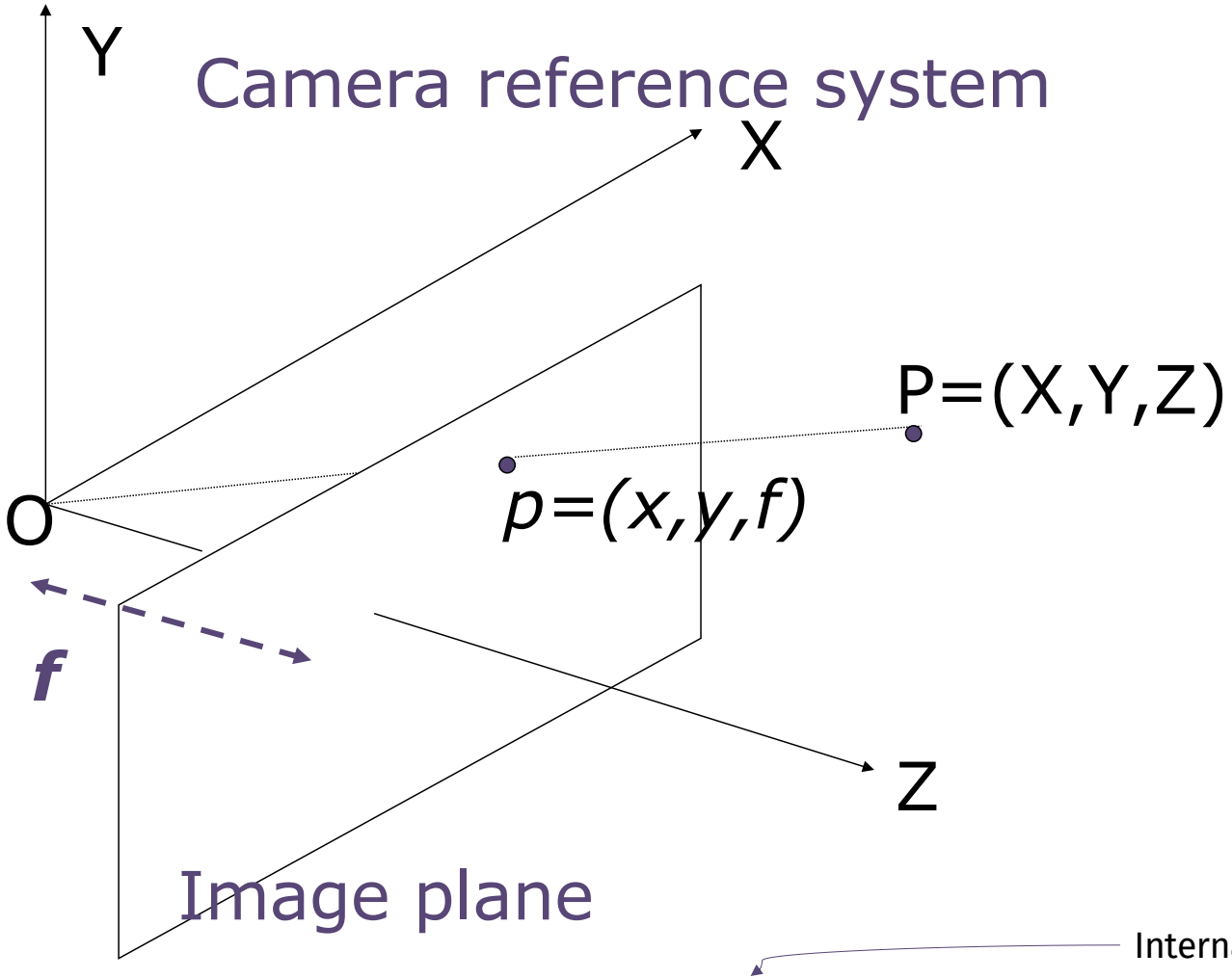
Foreward: image formation (geometrically speaking..)

THE GEOMETRY OF IMAGE FORMATION

Camera obscura



GEOMETRY: PERSPECTIVE OR PINHOLE MODEL



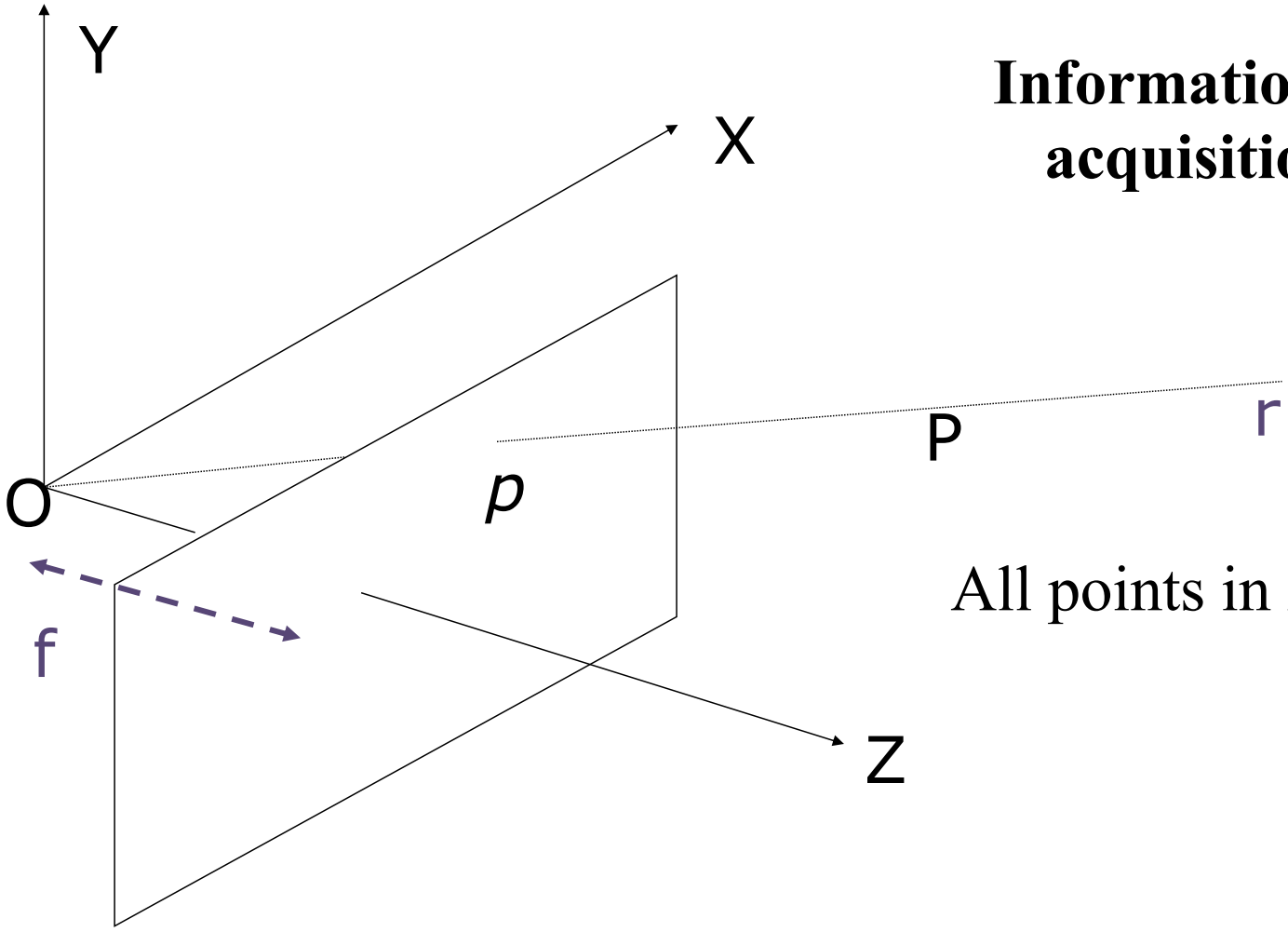
$$x = f\frac{X}{Z}$$

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

Internal parameter of the camera

f focal length

GEOMETRY: PERSPECTIVE OR PINHOLE MODEL

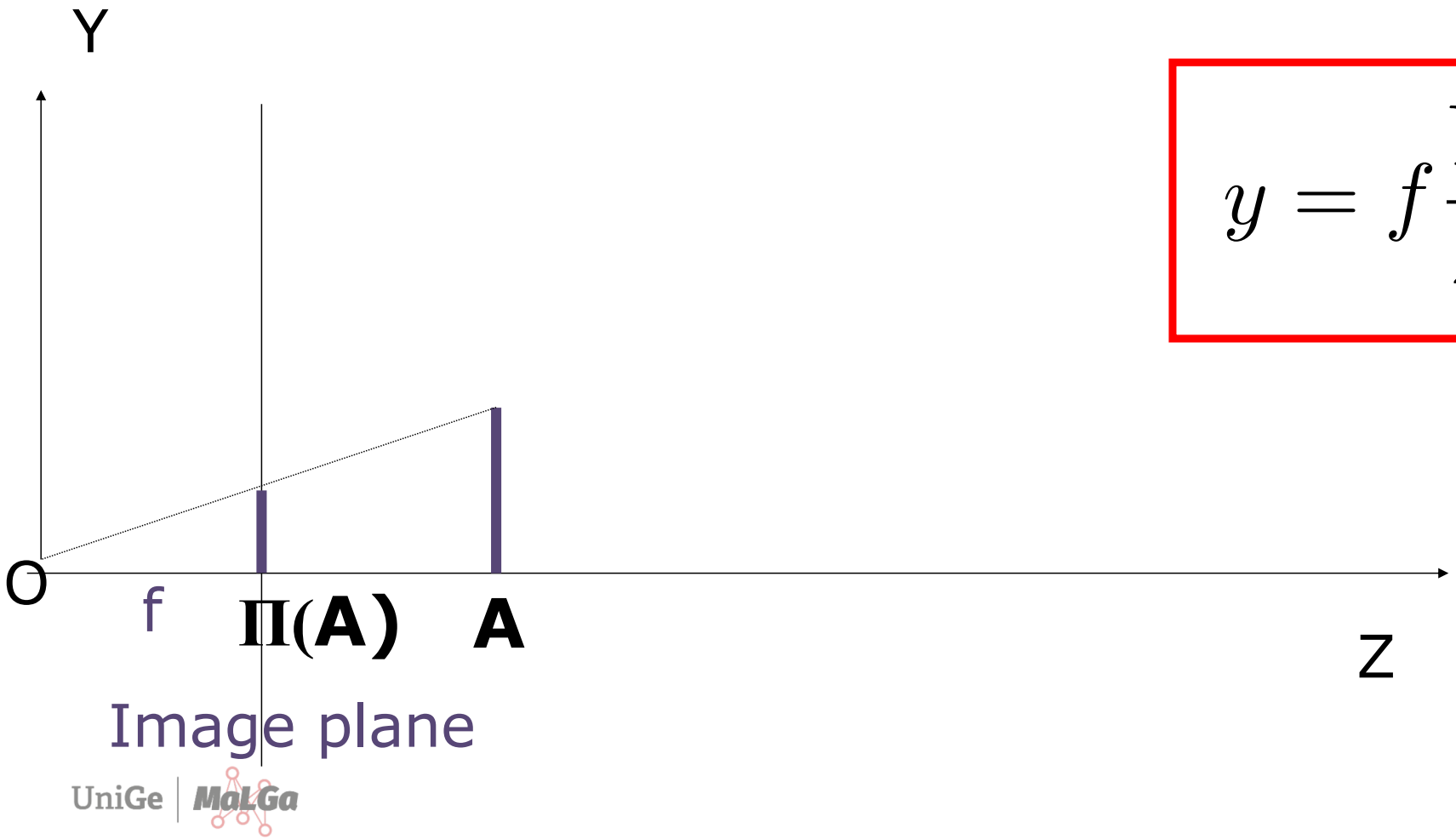


Information loss of the acquisition process

All points in *r* are projected in *p*

GEOMETRY: PERSPECTIVE OR PINHOLE MODEL

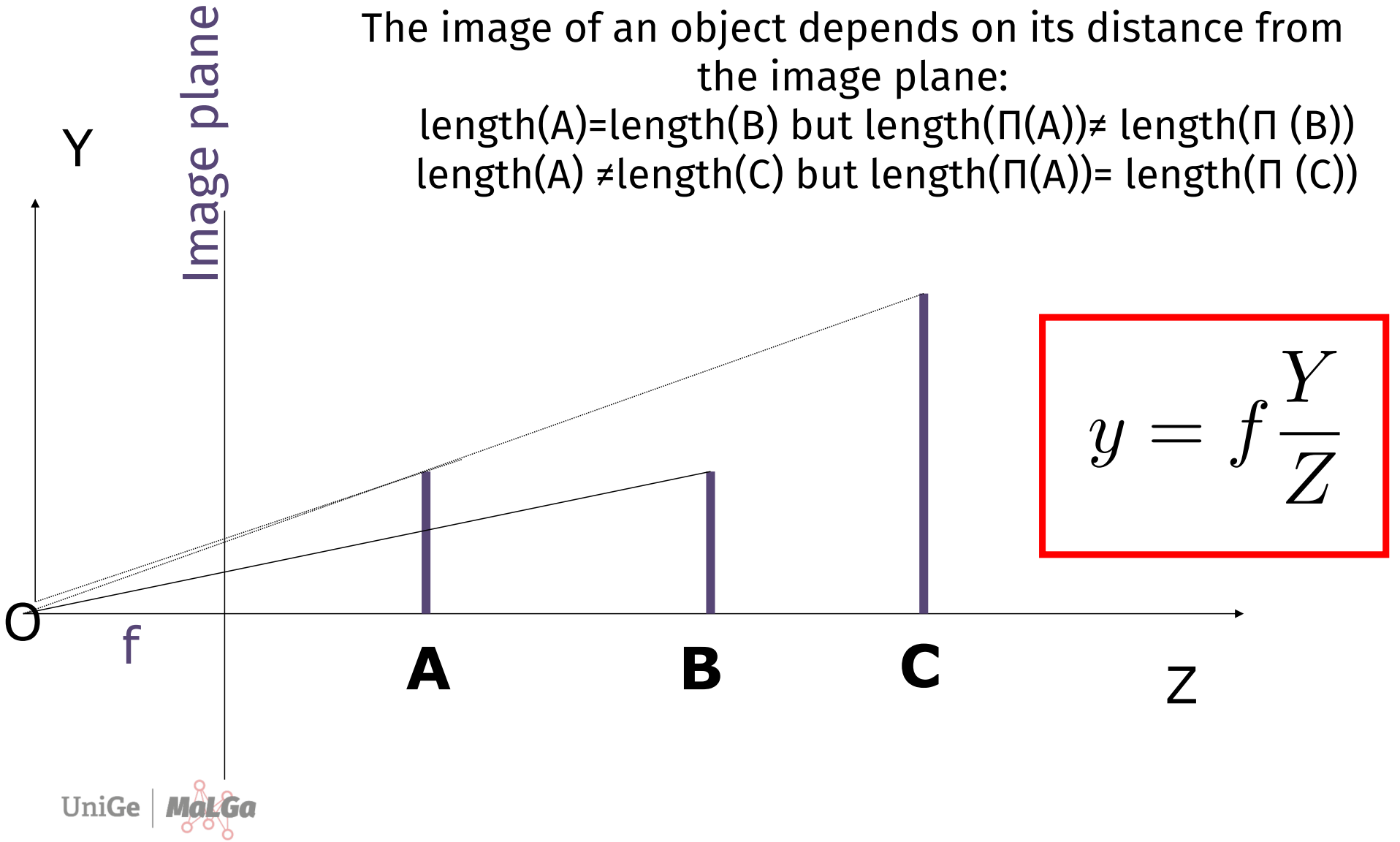
Perspective projection does not preserve distances: $\text{length}(\Pi(A)) \neq \text{length}(A)$



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

GEOMETRY: PERSPECTIVE OR PINHOLE MODEL

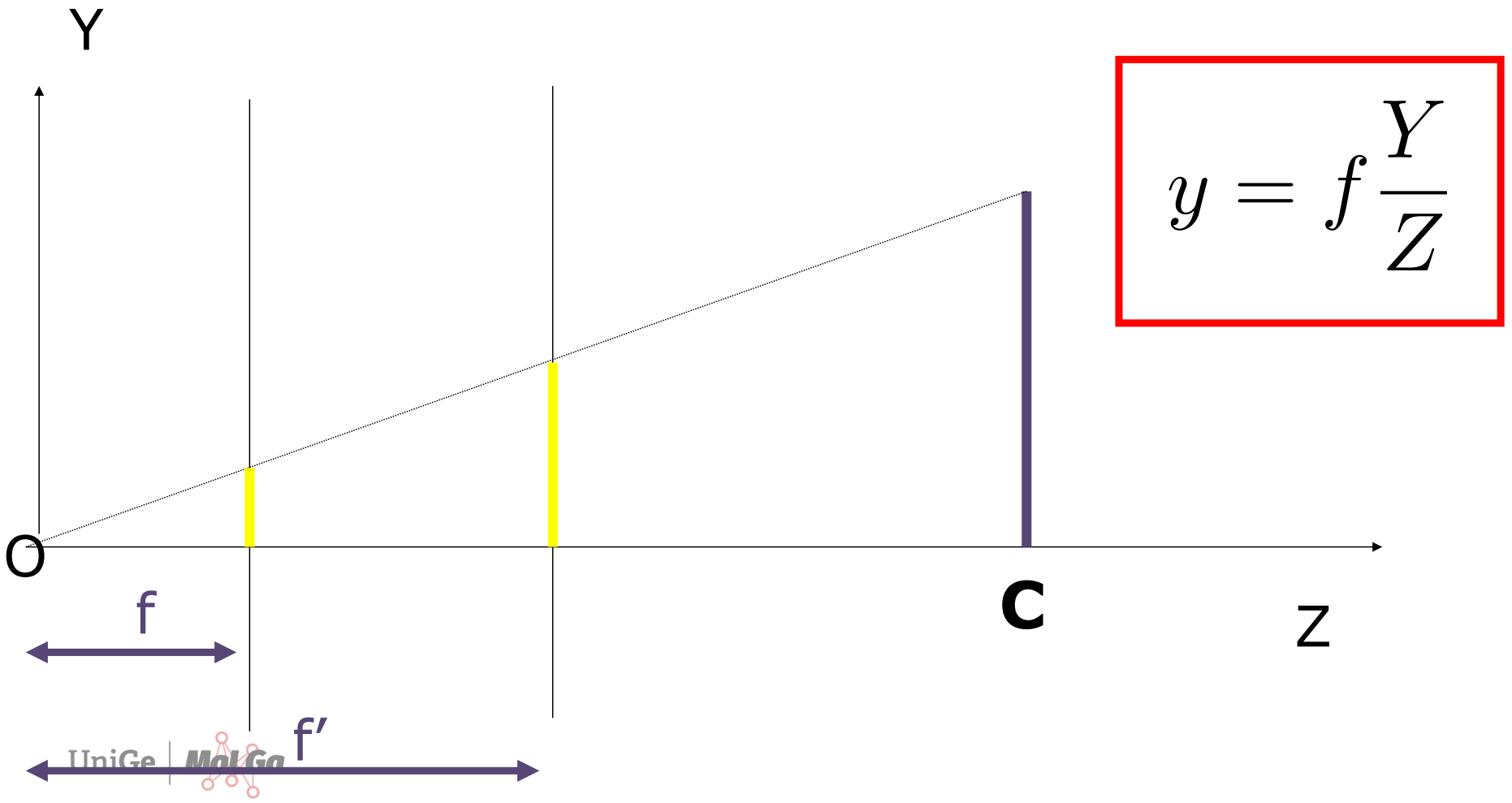
The image of an object depends on its distance from the image plane:
length(A)=length(B) but length(Π(A))≠ length(Π (B))
length(A) ≠length(C) but length(Π(A))= length(Π (C))



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

GEOMETRY:
PERSPECTIVE OR PINHOLE MODEL

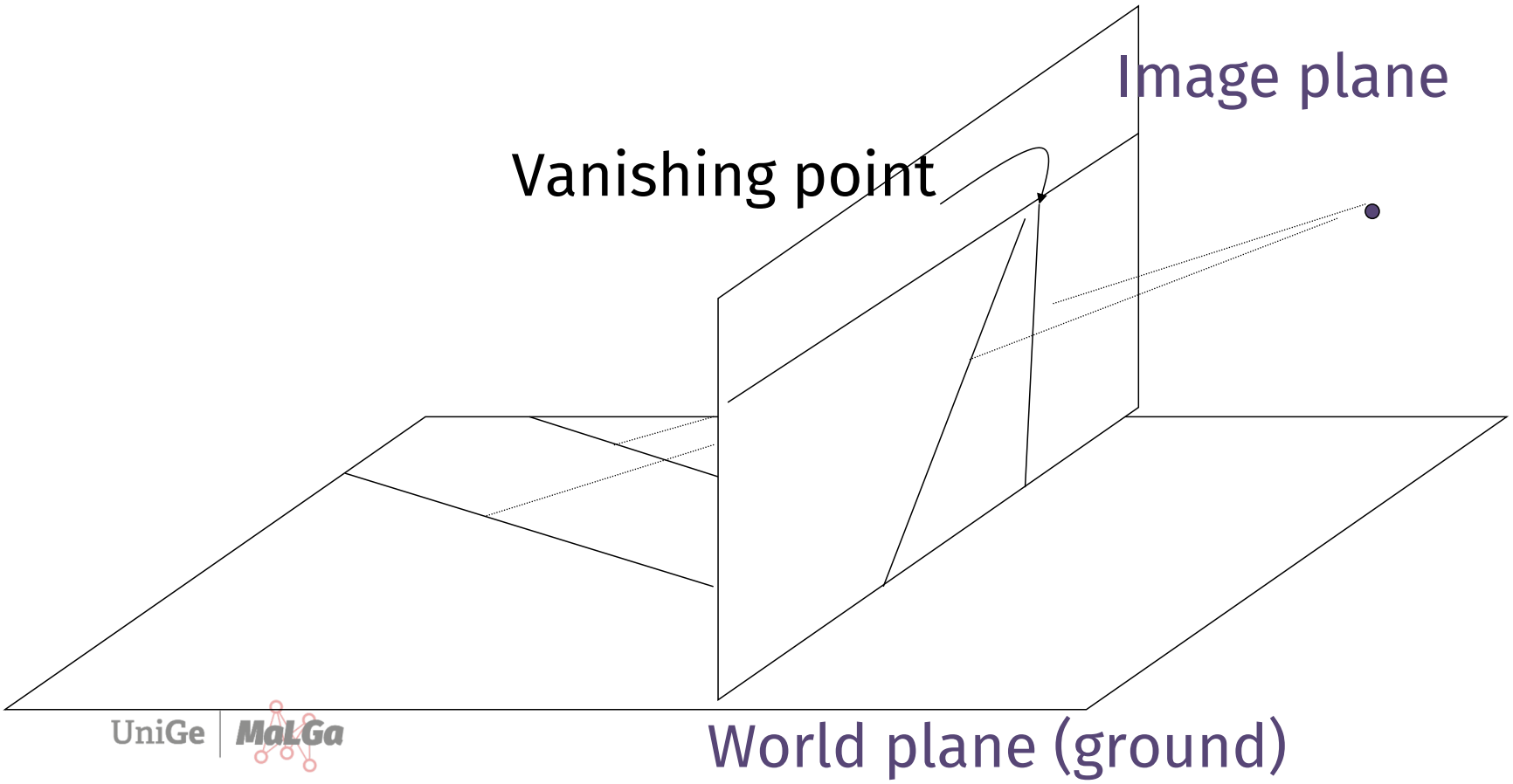
And also depends on the focal length



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

GEOMETRY: PERSPECTIVE OR PINHOLE MODEL

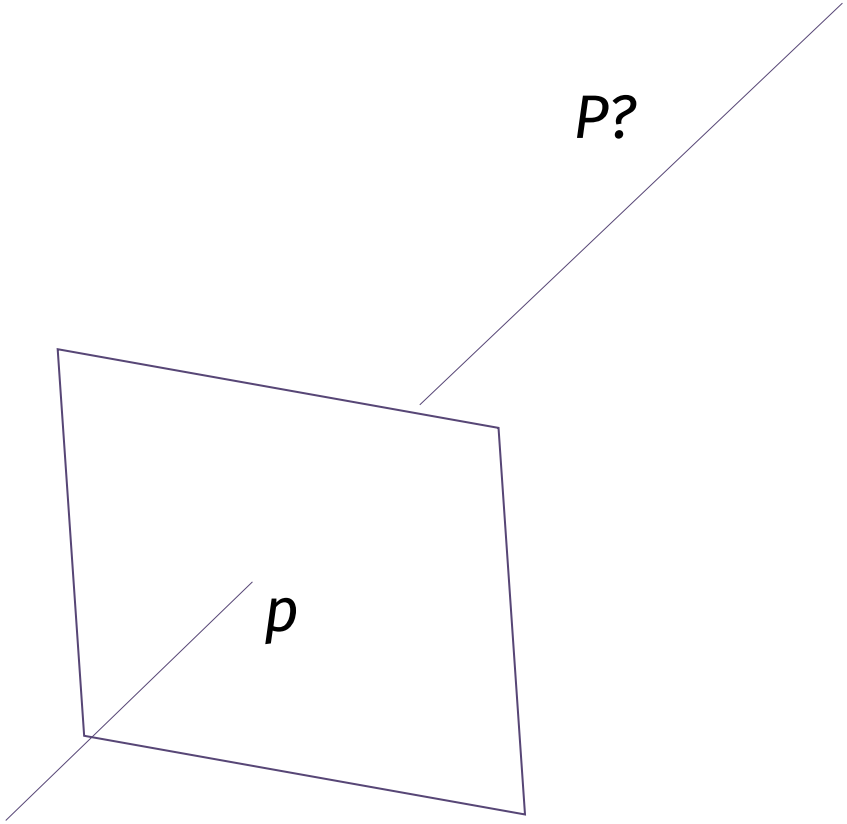
Perspective projection does not preserve parallelism



UniGe | MaLGa

Depth Estimation

Perspective projection is a lossy transformation

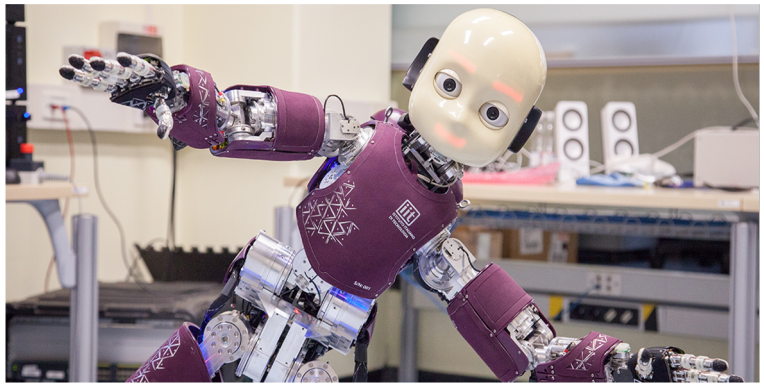


Introduction

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

we consider an acquisition system with 2 cameras

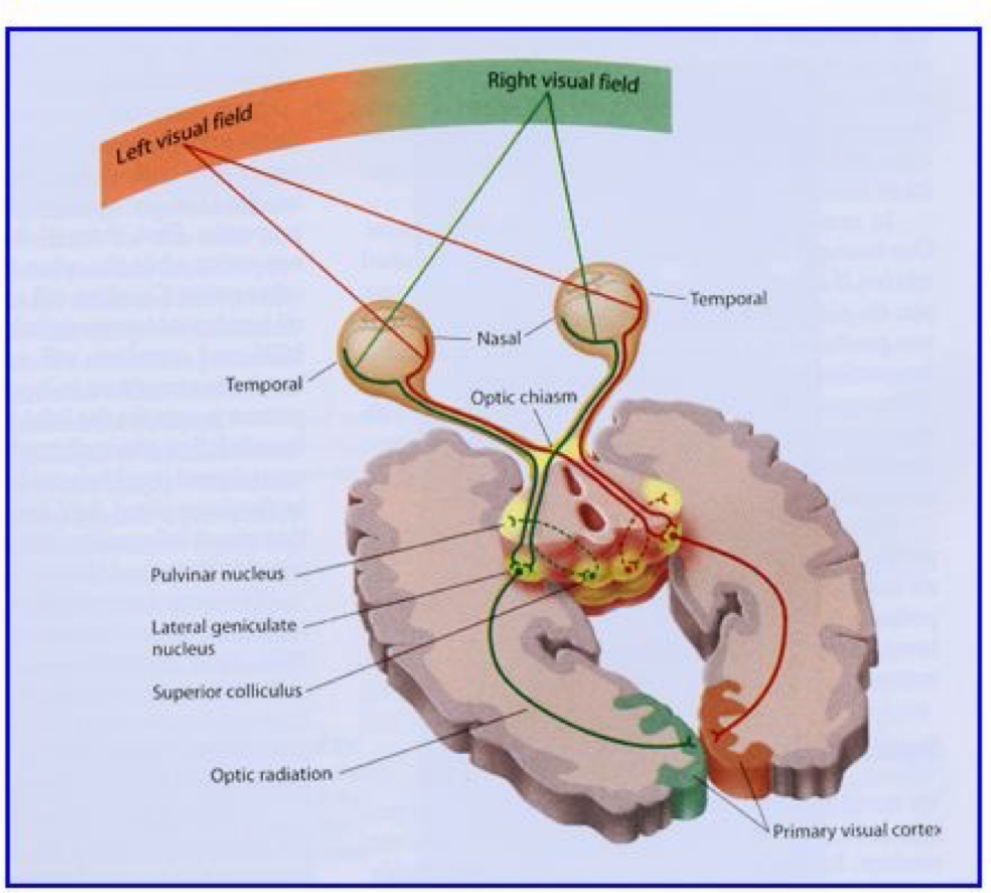
- “explicit” system: a stereo rig



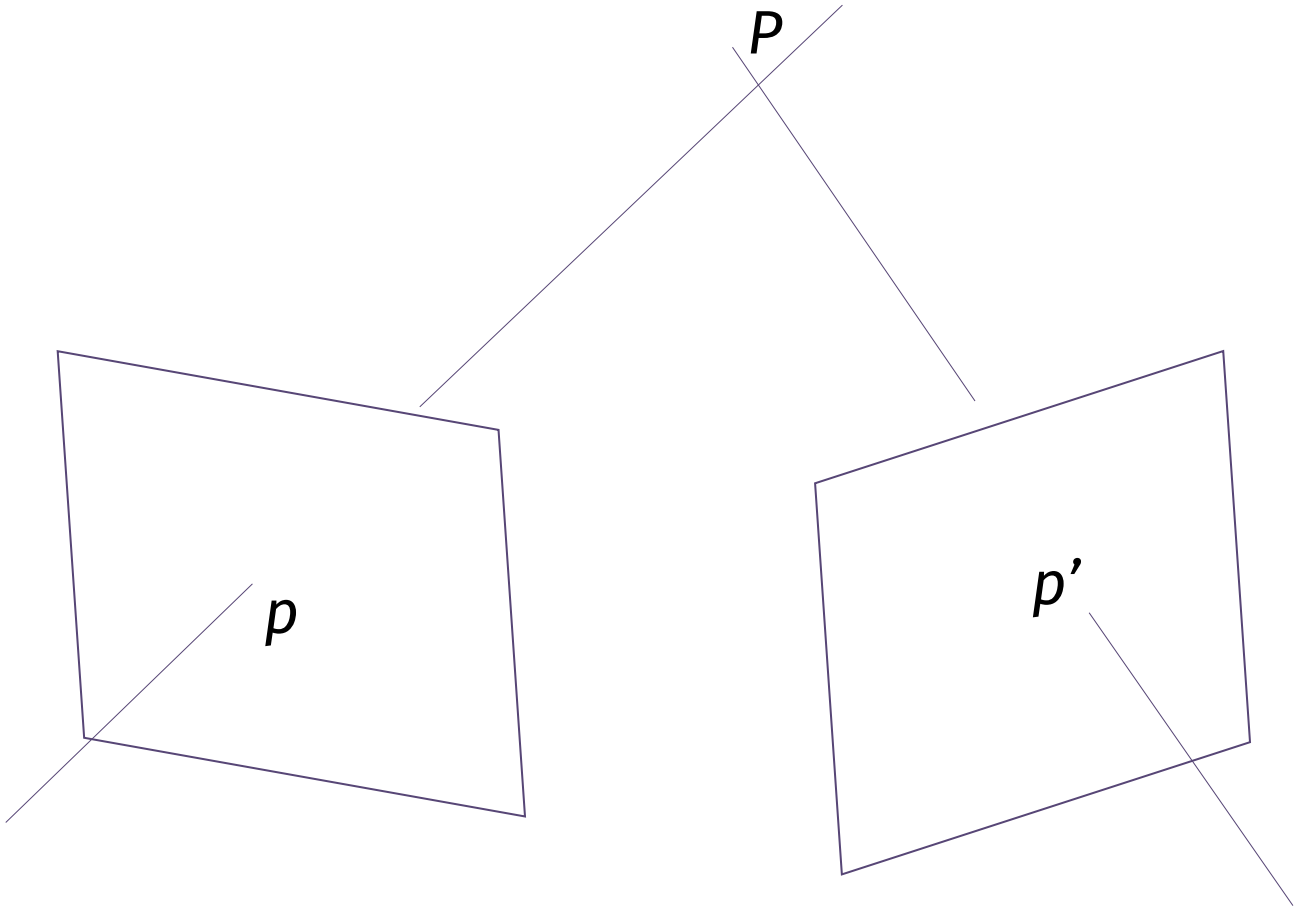
- “implicit”: one moving camera

Stereopsis: perception of depth

our 3D perception of the world is due to the interpretation that the brain gives of the computed difference in retinal position, called *disparity* between corresponding items



With a second view…



You first need to solve a correspondence (matching) problem to find the pair (p, p’)

A simple stereo system

Relationship between depth and disparity

We can recover depth information

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

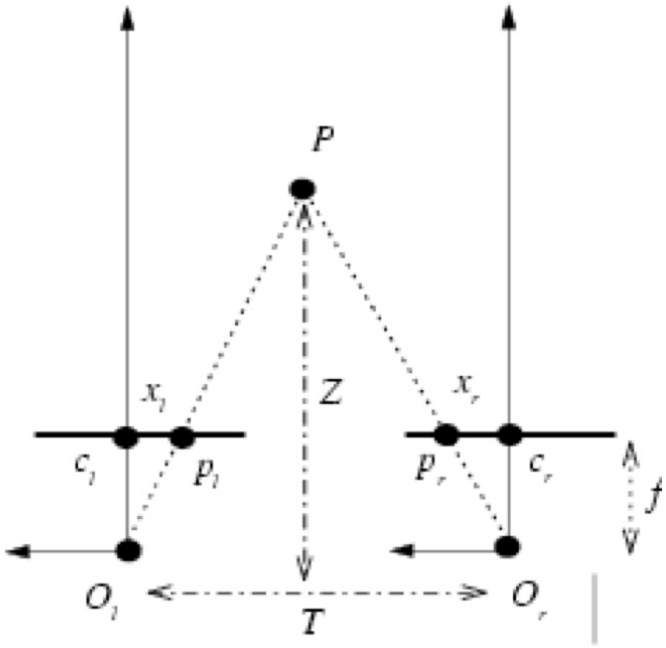
focal length

baseline

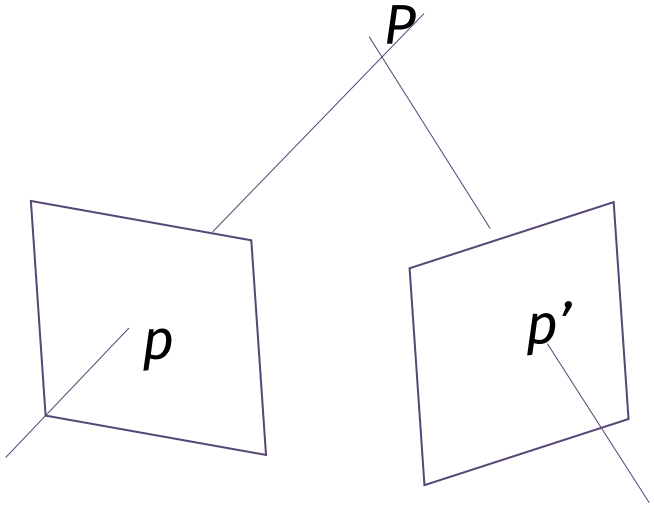
disparity

Depth is inversely proportional to disparity

Special case:
fixation point at infinity



The problems of stereopsis



stereo covers two main problems:

- finding **correspondences** between image pairs (p,p')
- **reconstructing** the 3D position of a point P given its corresponding projections on the images

step 1: produces 2.5 disparity maps (or depth maps)

step 2: produces a 3D point cloud

Correspondence problem

the correspondence problem involves two decisions:

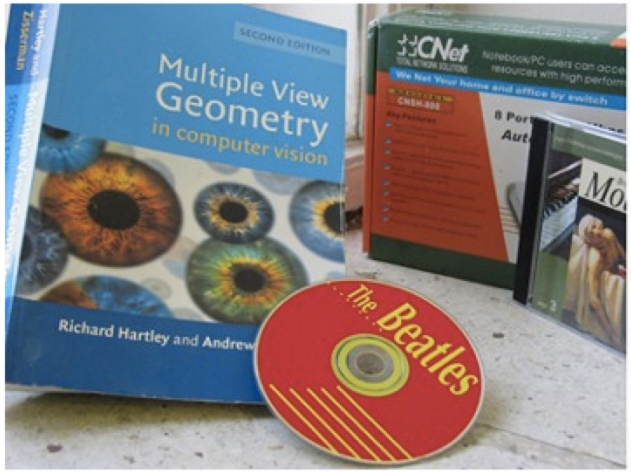
1. which image elements to match
2. which feature description + similarity measure to adopt

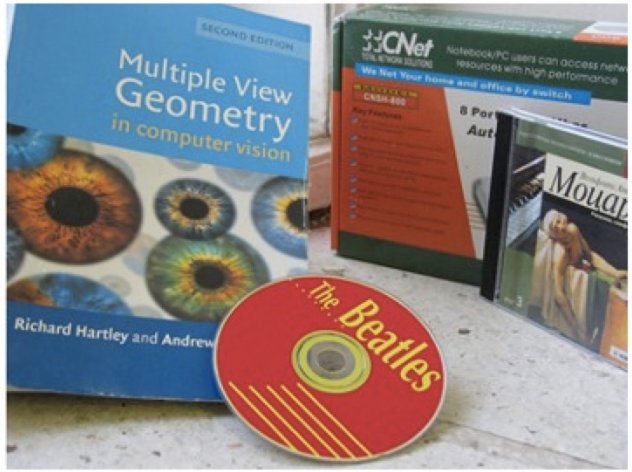
(1). can be solved in two ways:

- use all pixels in the image (obtaining *dense correspondences or disparity maps*)
- use subsets of pixels meeting some requirements (obtaining *sparse correspondences)*

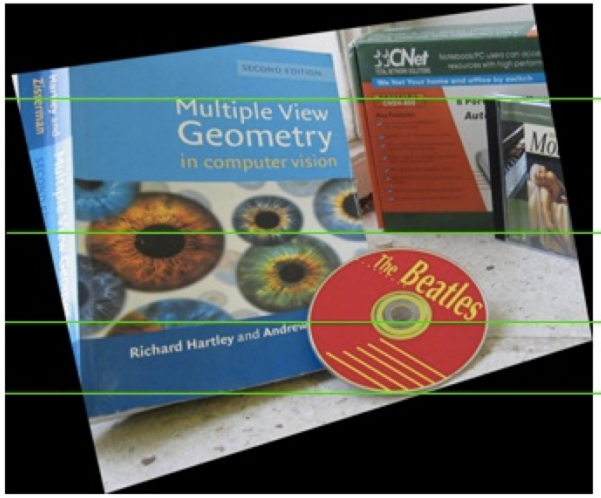
Computing disparity maps

Two words on input images





A stereo pair



A rectified stereo pair

Equiv to fixation point at infinity

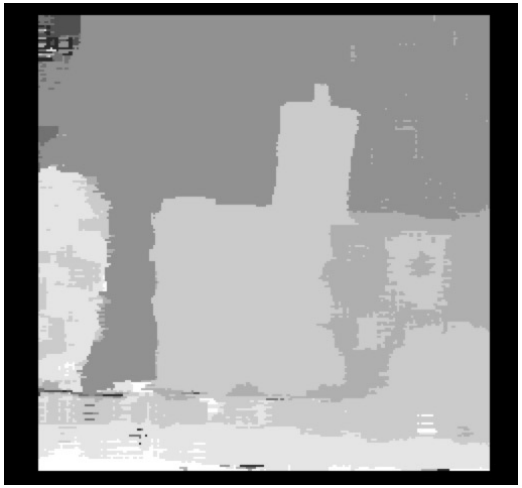
It's a geometric transformation of
the most general type (a projectivity)

Dense correspondences

we consider the so-called *correlation methods* applied to image patches (neighbourhoods of pixels)

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

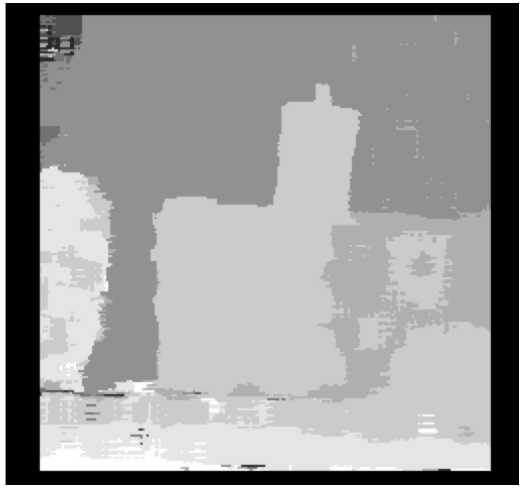
input:

- a stereo pair of rectified images *Il* and *Ir*
- size of a correlation window W
- a search range [dmin, dmax]

for each pixel *pl* of *(i,j)* coordinates in *Il*

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

Examples



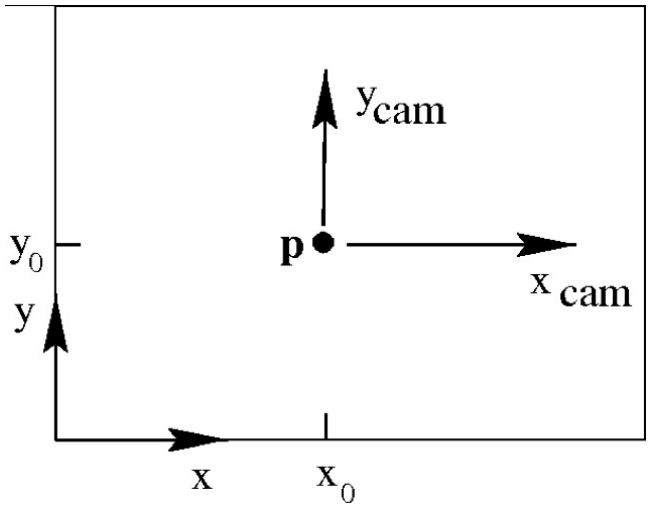
Parameters of a stereo system

How to relate points in the world with pixels in the image

Intrinsic/internal parameters

- characterize the mapping of an image point from camera to pixel coordinates in each camera

- (translation and scaling)



$$K = \begin{bmatrix} \alpha_x & 0 & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}$$

all entries in pixel coordinates

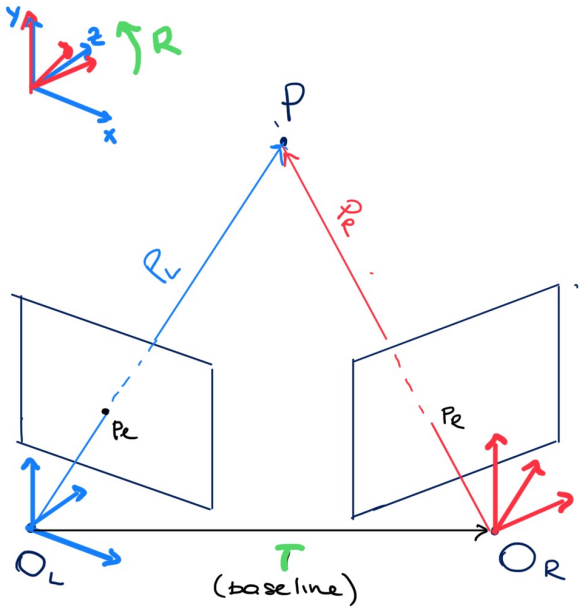
$$\alpha_x = f m_x$$
$$\alpha_y = f m_y$$

number of pixels per unit distance

$$x_0 = m_x p_x$$
$$y_0 = m_y p_y$$

Parameters of a stereo system

How to relate points in the world with pixels in the image



Extrinsic/external parameters

- describe the relative position and orientation of the two cameras (R, **T**)

$$\mathbf{P}_r = R(\mathbf{P}_l - \mathbf{T})$$

- without loss of generality we may assume the left camera is our global (world) reference frame. Then (R, **T**) describe the

System calibration: sketch

Internal and external parameters can be estimated by camera calibration procedures

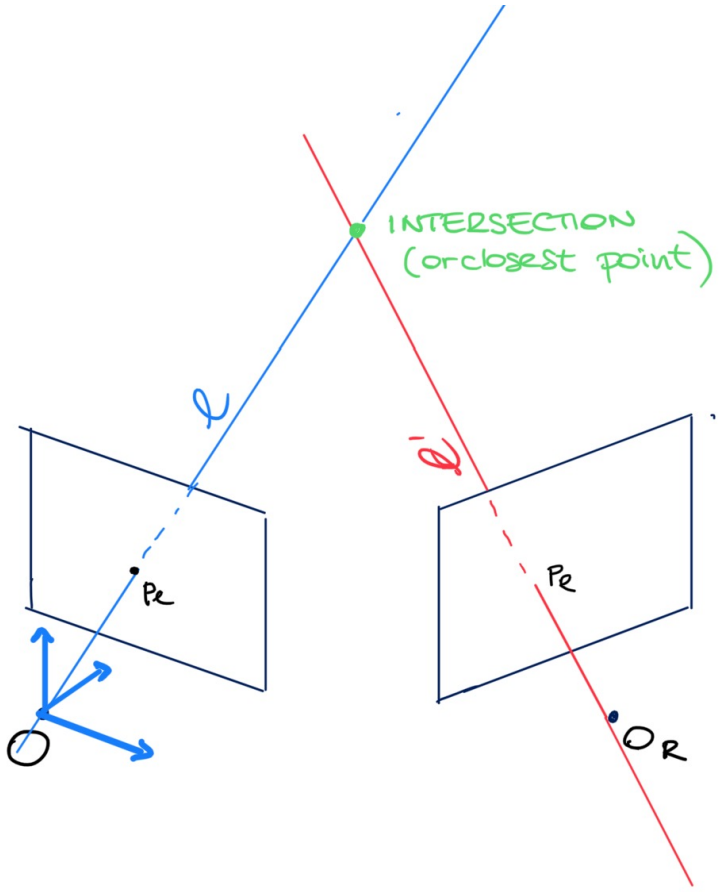
External parameters can be obtained starting from point correspondences (it requires some further exploration on the geometrical aspects of a stereo system, the so-called **epipolar geometry**)

2D to 3D: points triangulation (sketch)

Given a pair of corresponding points in 2D (metric coordinates on a common reference frame) (p_l, p_r)

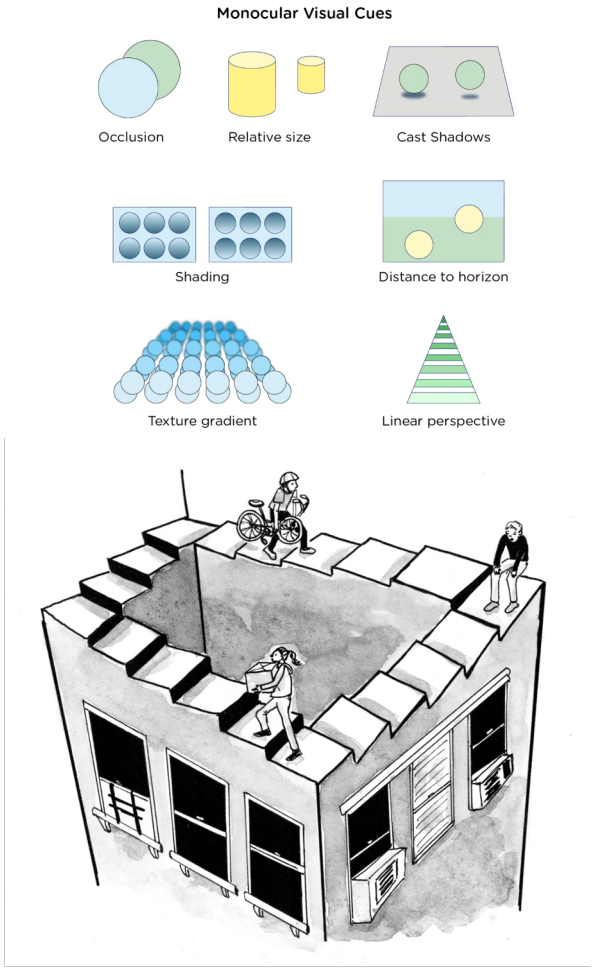
Estimate the equation of the two projection rays ℓ, ℓ'

Compute the intersection (or the closest point wrt both lines) it will be the 3D reconstruction P



It is not all about geometry

- Stereopsis in humans (given the narrow baseline) works primarily in short range (up to 10 meters)
- Beyond this range, the brain uses other cues
- Humans can still infer depth from single images



Motion Analysis

introduction

What do we gain if we analyse videos instead than images?

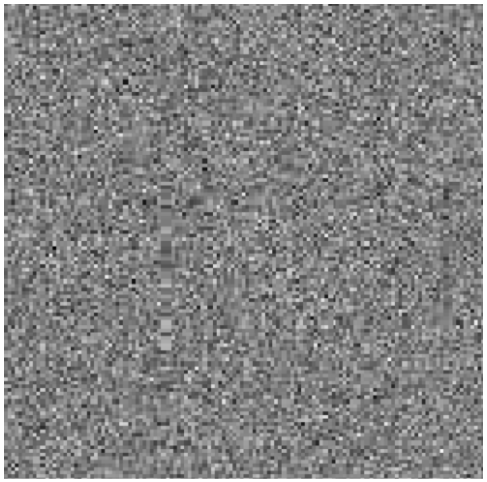
We are observing a scene with **one** camera acquiring a set of images “close in time”

image sequence: series of N images, or *frames*, acquired at discrete time instants

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

fixed time interval (small)

the importance of motion cues



the presence of a structure is suggested to the visual system only by motion information: the changes we perceive from one frame to the next

Introduction

Brightness Constancy assumption

An important **assumption** to make when analysing image sequences is the following:

illumination does not vary in the observation interval

In this case the image changes from t_1 to t_2 are caused by the *relative motion* between scene and camera

the problems of motion

We may identify different types of questions related with motion analysis

- **correspondence:** which elements of a frame correspond to which elements of the next frame?
  (very closely related with image matching...)
  → motion estimation - optic flow computation
- **reconstruction:** what was the 3D position and 3D velocity of the observed elements

Since adjacent frames are similar, correspondence is rather easy, but reconstruction is hard!!

the problems of motion (cont'd)

Other problems related to motion analysis

- **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**
  (in the next slides)

- **tracking:** estimate the (2D or even 3D) trajectories of points or objects from image sequences
  Main computational tools: dynamic filters (eg Kalman)

Motion analysis

What we will do

Motion segmentation: change detection

Correspondence: optical flow estimation

Motion segmentation

Change detection

I_t

M_t

In the case the camera is still motion segmentation is can be implemented as a difference

Assuming we have a **reference image** I_{ref}

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

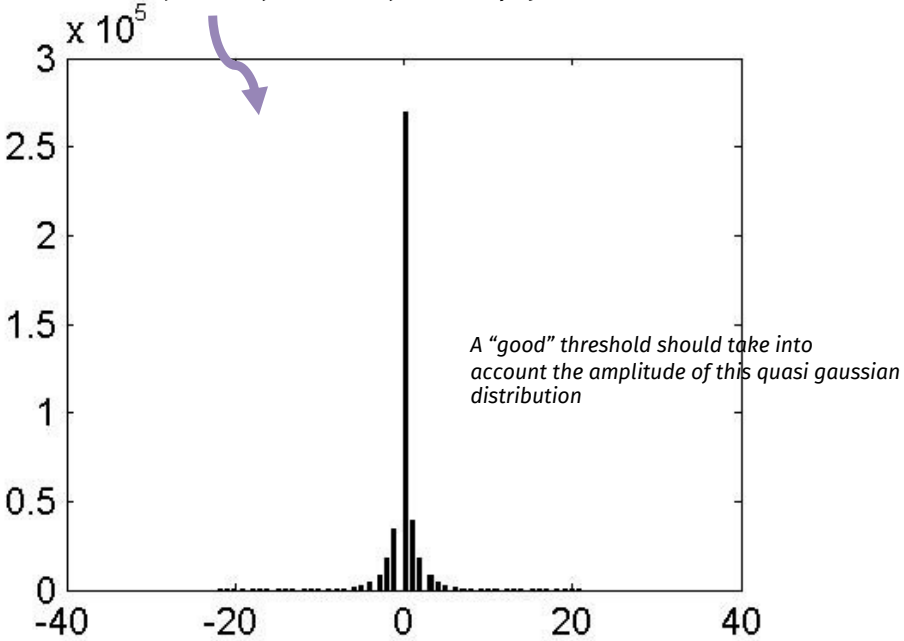
the threshold τ depends on the acquisition device and the acquisition conditions. It must be chosen considering a trade-off between false positives and false negatives.

The threshold

I_t

I_{t+1}

Histogram of the difference between 2 apparently identical images (two adjacent frames of an empty scene)



A "good" threshold should take into account the amplitude of this quasi gaussian distribution

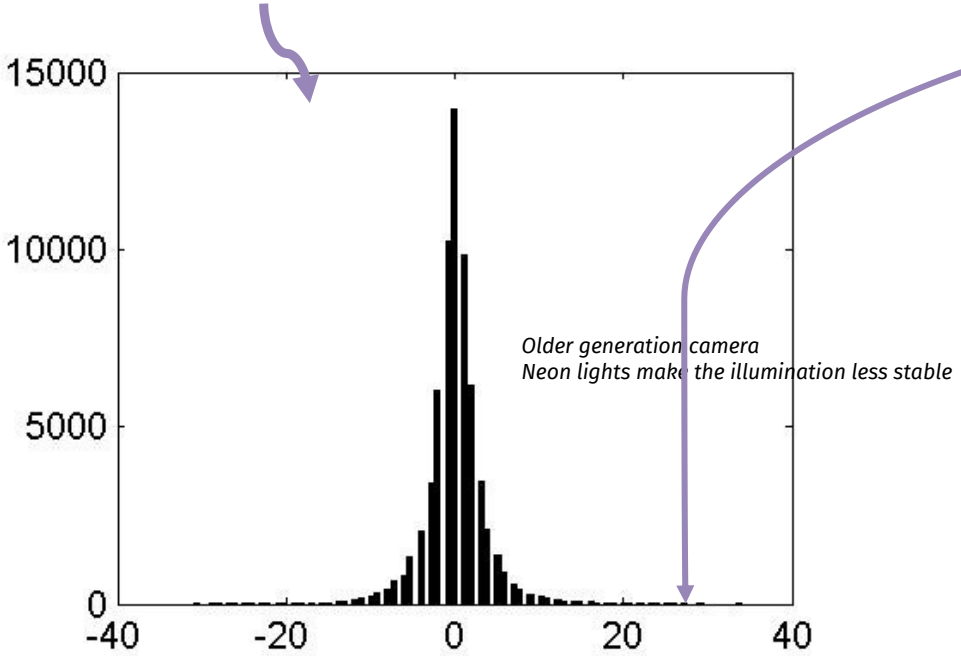
threshold=10

The threshold

I_t

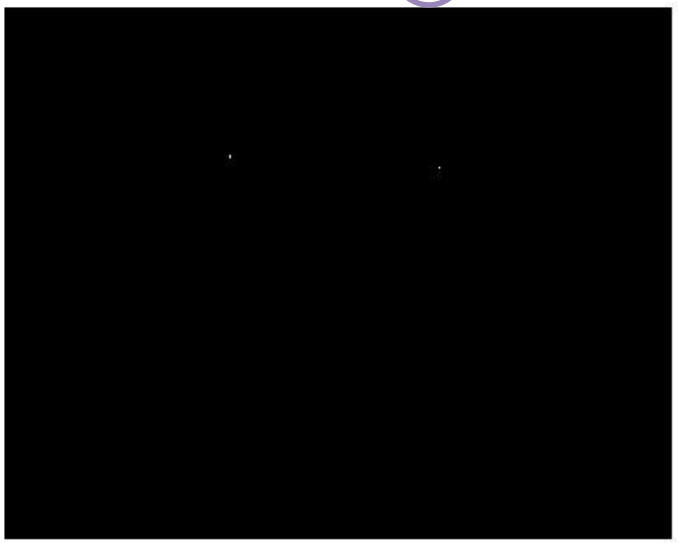
I_{t+1}

Histogram of the difference between 2 apparently identical images
(two adjacent frames of an empty scene)



Older generation camera
Neon lights make the illumination less stable

threshold=30



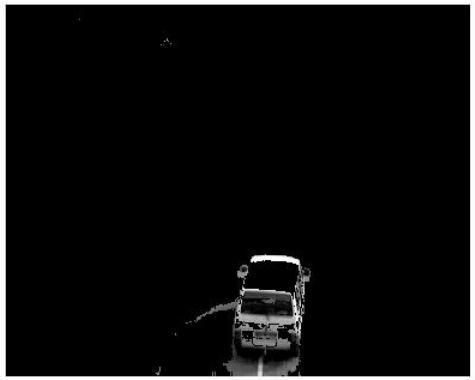
The reference image

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

frame 1

frame 2

|I1-I2|>th



In general it is not a good choice

Artifacts
Difficult to detect slow motion
Noisy localization

The reference image

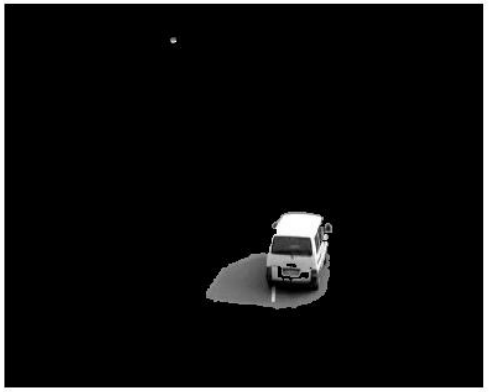
The reference image or, more in general the *background model*, is a “picture” of the empty scene, containing all the parts which are not moving

background

frame

|I-B|>th

after a cleaning procedure



Change detection algorithm

Version 1

INPUT – image sequence $\{I_0,\dots,I_T\}$

Initialize I_{ref} with the first N images

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

For each new frame $I_t \in \{I_{N+1}\dots I_T\}$

detect changes:

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

OUTPUT – binary maps sequence $\{M_{N+1},\dots,M_T\}$

PROS
- Very simple

CONS
- It assumes the background will not change in all the observation span (almost impossible in practice!)

Look at the door!

Change detection algorithm

Version 2 – running average

NOTICE : this method deals with *slowly changing* background (e.g, illumination changing in the scene)

INPUT – image sequence $\{I_0,\ldots,I_T\}$

Initialize the background B_0 as in version 1

For each new frame $I_t \in \{I_{N+1}\ldots I_T\}$

1. detect changes

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

2. Update the background

$$B_t(x,y) = \begin{cases} B_{t-1}(x,y) & \text{if } |I_t(x,y) - B_{t-1}(x,y)| > \tau \\ (1-\alpha)B_{t-1}(x,y) + \alpha I_t(x,y) & \text{otherwise} \end{cases}$$

OUTPUT – binary maps sequence $\{M_{N+1},\ldots,M_T\}$

notice: the background model is updated **at each frame**

Change detection

The reference image – version 2 – running average

$$B_t(x,y) = \begin{cases} B_{t-1}(x,y) & \text{if } |I_t(x,y) - B_{t-1}(x,y)| > \tau \\ (1-\alpha)B_{t-1}(x,y) + \alpha I_t(x,y) & \text{otherwise} \end{cases}$$

PROS

- it is quite robust to moving objects in the scene
- it incorporates smooth 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

General case

If the camera is moving, we need to consider the presence of a dominant motion (the ego motion)

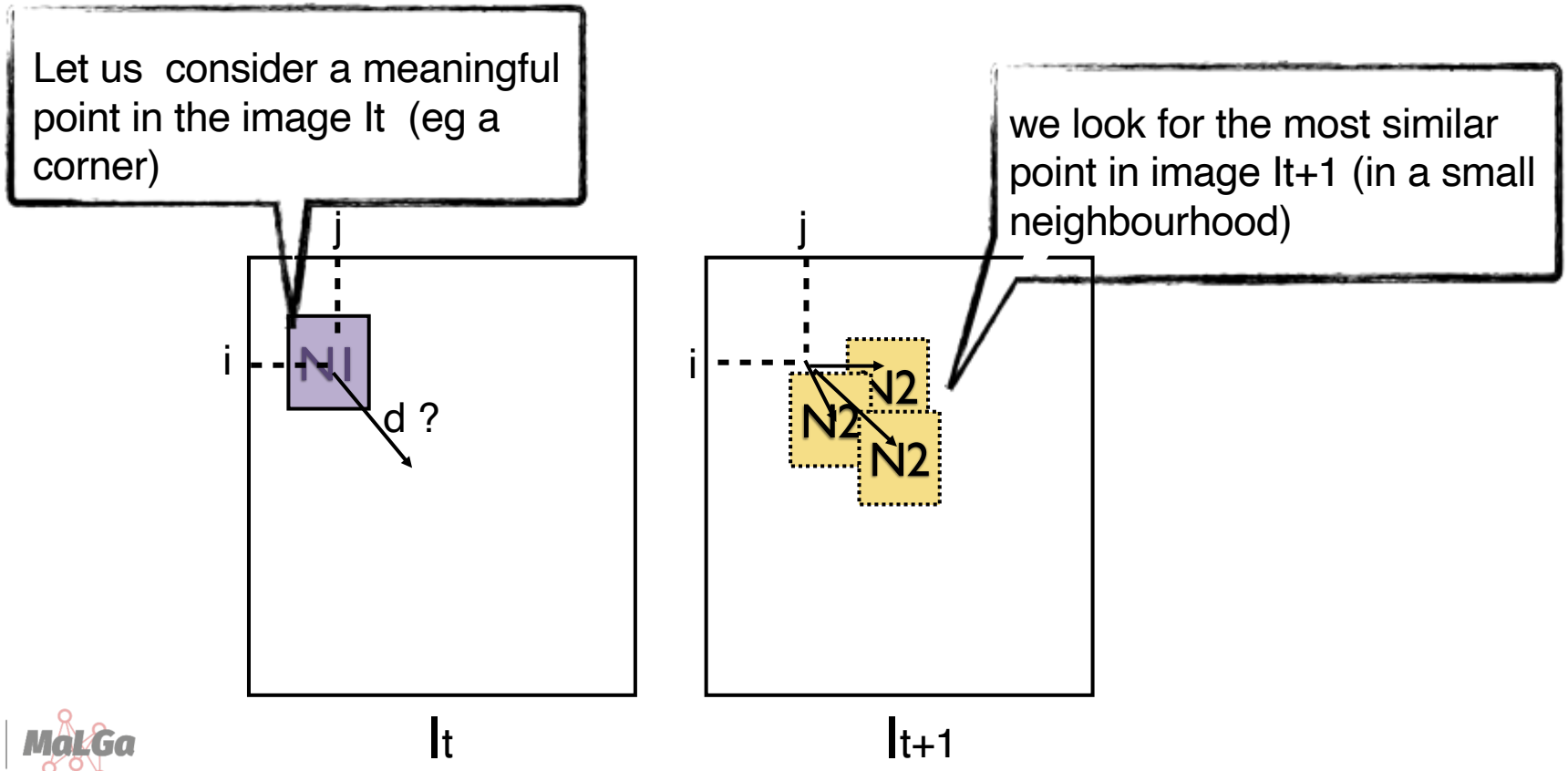
A possible approach is to first obtain a rough estimate of the ego motion and then compensate it to generate a synthetic datum *as if* the camera were still

Much harder and computationally intensive (as an alternative we often look for semantic info: people detection, car detection, ...)

Motion analysis as a matching problem

correspondence: which elements of a frame correspond to which elements of the next frame?

We take advantage of the high similarity between adjacent frames



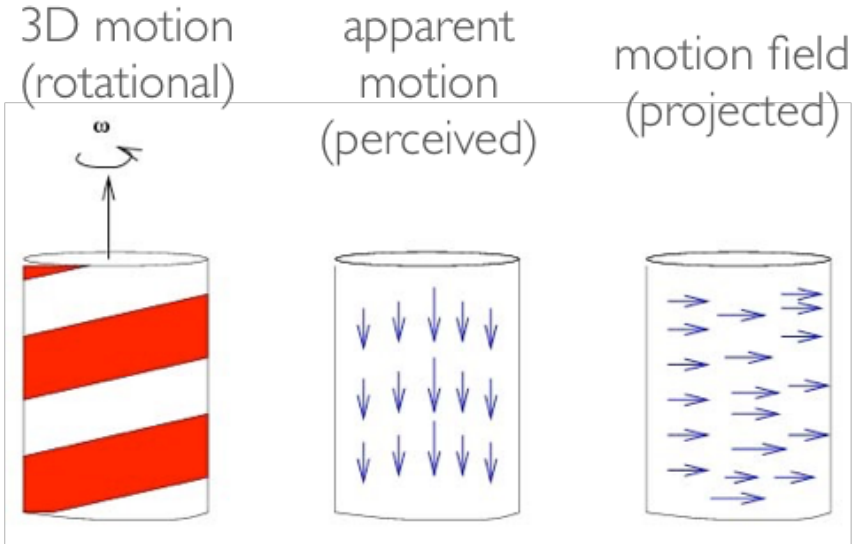
Motion analysis as a correspondence problem

In the case of motion, correspondence may also be seen as a problem of *estimating the apparent motion of the brightness pattern* (the so called **optical flow**)

Why ”apparent”? *Because it’s not the real motion, but the one we perceive*

Example1. A ball is bouncing in a dark room: what do we perceive? Nothing!

Example2: the barber pole



Optical Flow

Gradient-based derivation

image brightness constancy equation: from one frame to the next the image appearance does not change

$$I(x, y, t) = I(x + u, y + v, t + 1)$$

If the motion is small, we can use the following Taylor approximation

$$I(x + u, y + v, t + 1) = I(x, y, t) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial x}v + \frac{\partial I}{\partial t} \quad + H.O$$

We thus obtain

$$\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} = 0$$

Or more compactly

$$I_x u + I_y v + I_t = 0$$

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

Optical Flow

Gradient-based derivation

The optical flow is a vector field subject to the constraint

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

This constraint gives us 1 equation per each image point

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

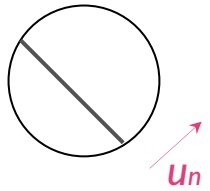
Optical Flow

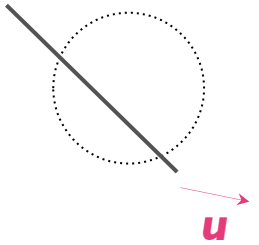
The aperture problem

the image brightness constancy equation allows us to determine the optical flow component parallel to the spatial image gradient (normal flow)

Analytically

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





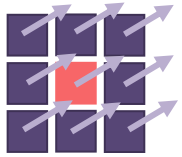
Optical Flow Algorithms

Lucas Kanade Algorithm

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

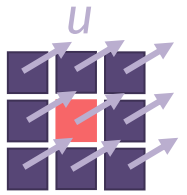


Optical Flow Algorithms

Lucas Kanade Algorithm

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.

Optical Flow Algorithms

Lucas Kanade Algorithm

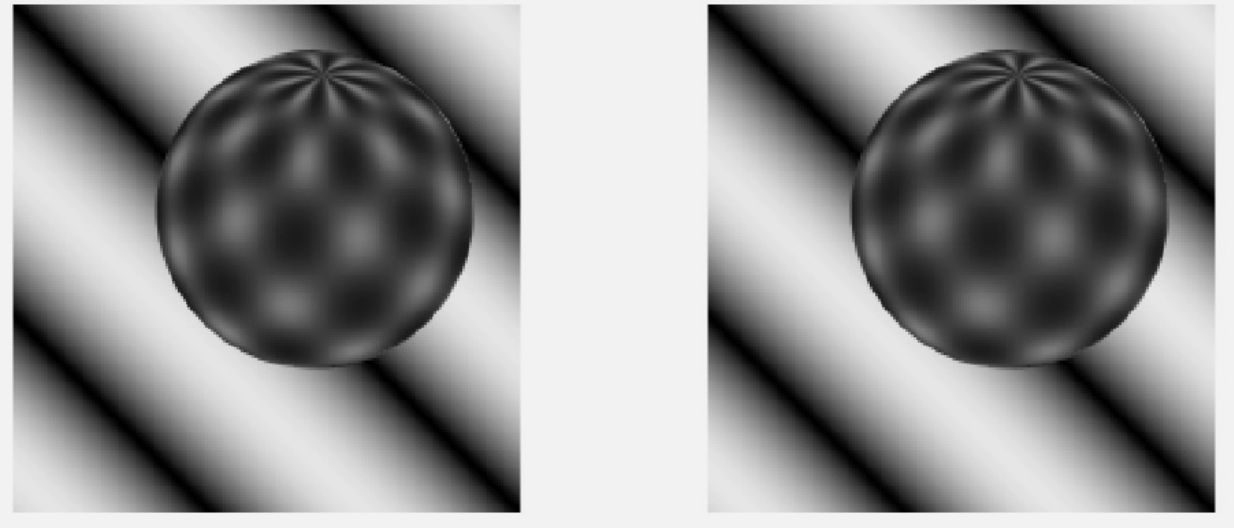
The linear system may be solved with the pseudo-inverse

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

notice that the inversion of the matrix will be ill-posed if the matrix is not full rank

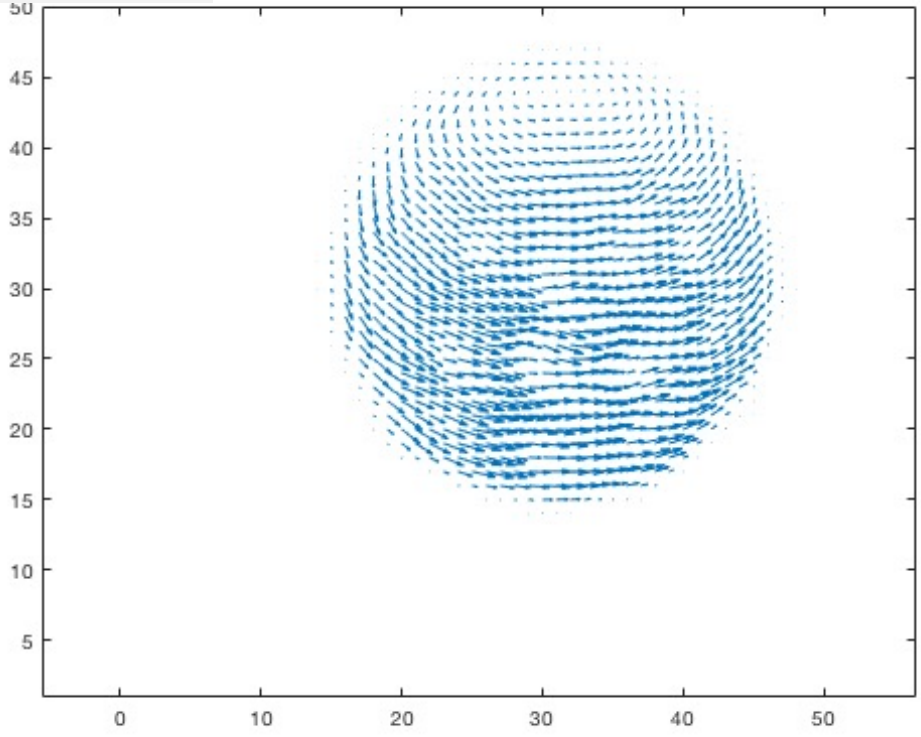
The matrix is full rank in the proximity of corners (points who do not suffer from the aperture problem)
This is why often times Lucas Kanade is implemented as a *sparse* algorithm (after a corner detection stage) – see for instance OpenCV

Lucas Kanade example

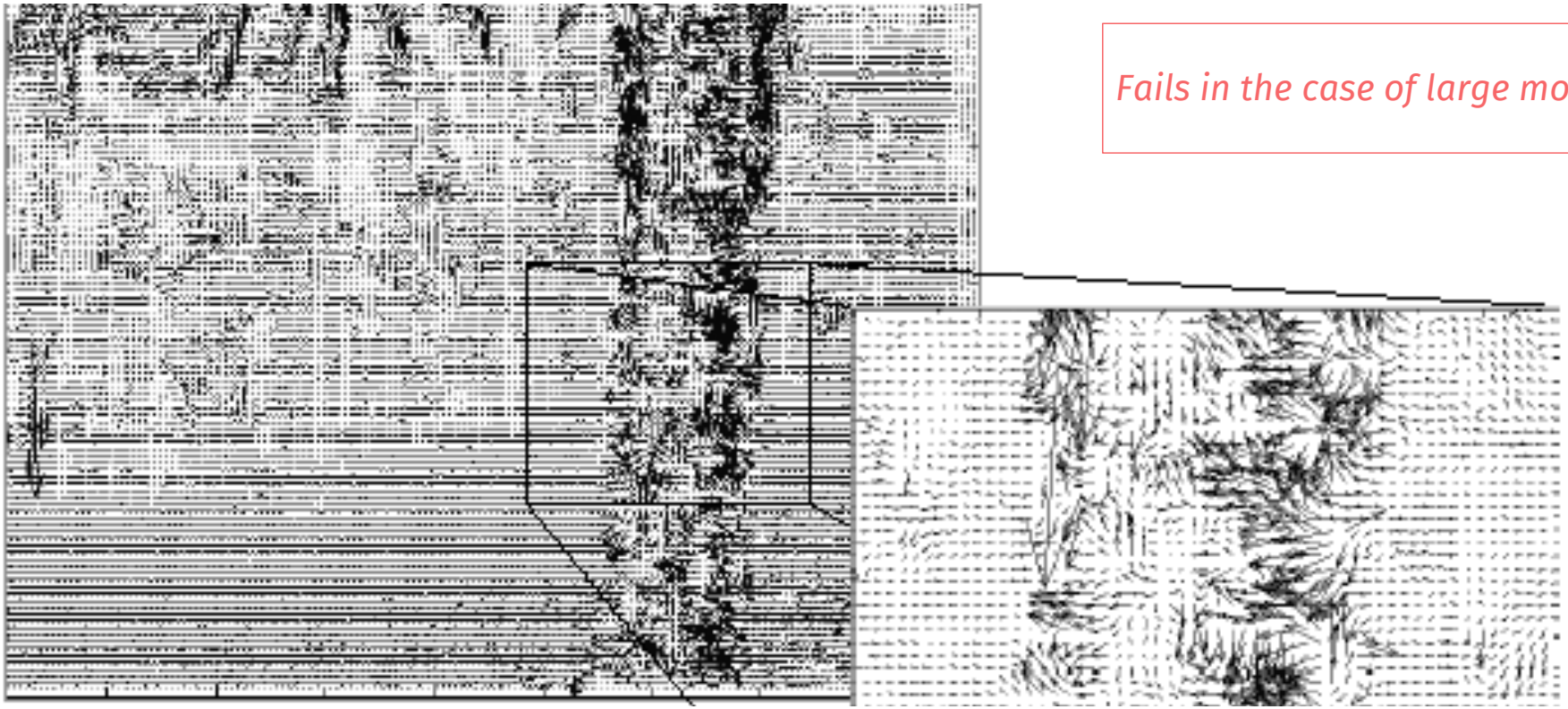


Lucas Kanade on a synthetic highly textured sequence of a *slowly* rotating sphere

Small displacements!

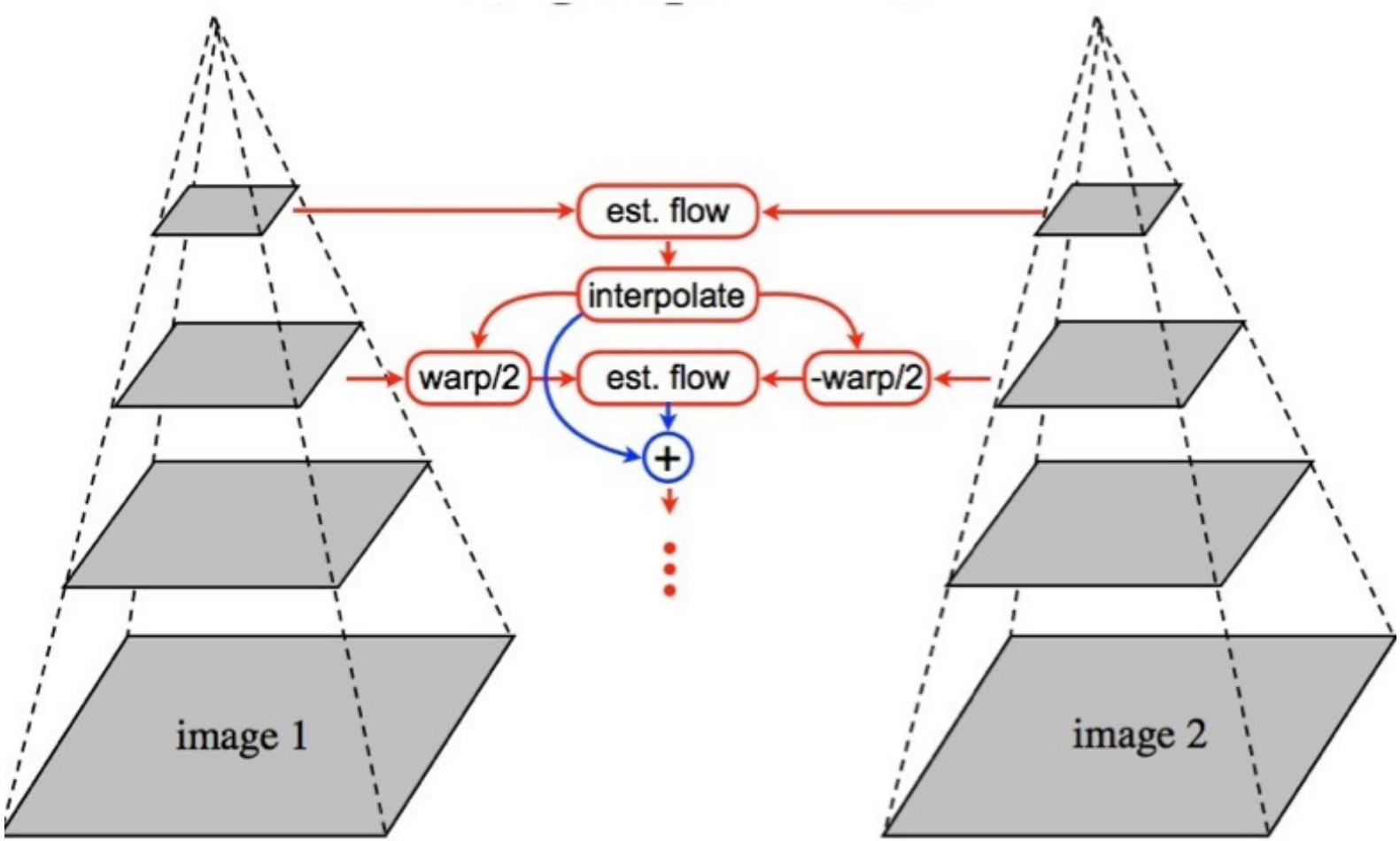


Lucas Kanade example



Fails in the case of large motion

coarse to fine estimation: just a sketch



The image frames are first represented as Gaussian Pyramids (see last week's class)

UniGe