University of Zurich[UZH] | ETH zürich

Institute of Informatics – Institute of Neuroinformatics

ROBOTICS & PERCEPTION GROUP

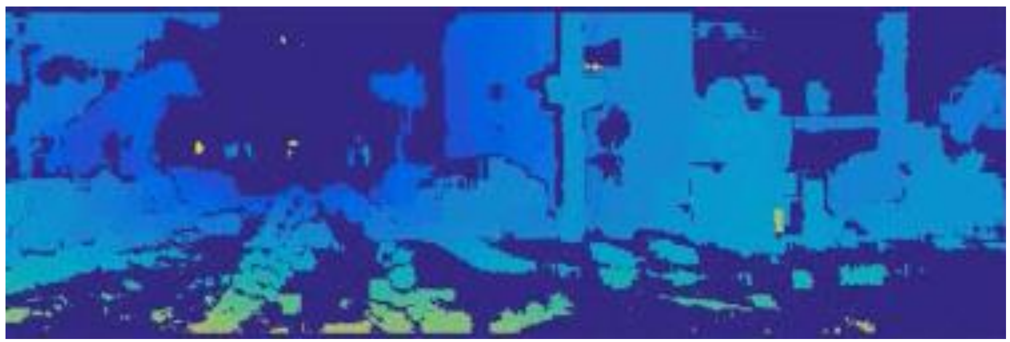
# Lecture 07
# Multiple View Geometry 1

Davide Scaramuzza

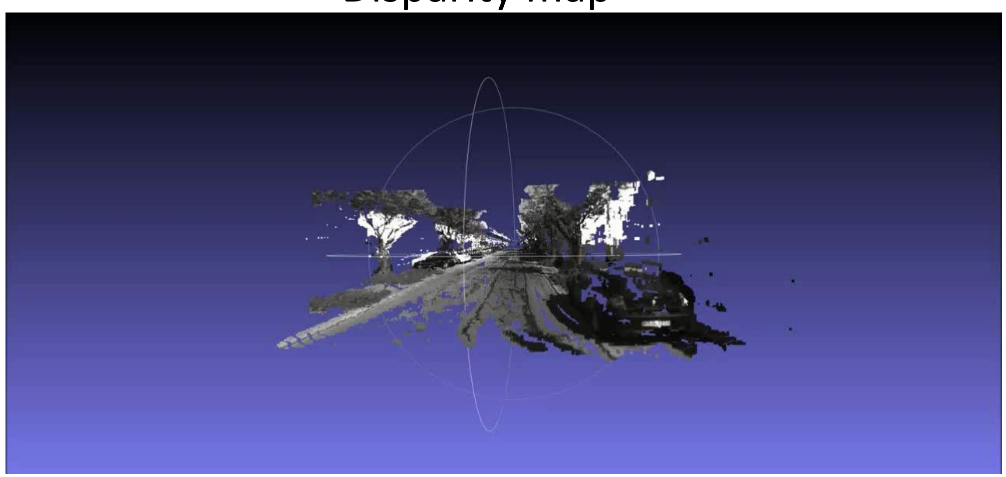
Guillermo Gallego

# Lab Exercise 4 - Today afternoon

- Room ETH HG E 1.1 from 13:15 to 15:00
- Work description:  Stereo vision: rectification, epipolar matching, disparity, triangulation



Disparity map



3D point cloud

# Course Topics

- Principles of image formation
- Image Filtering
- Feature detection and matching
- Multi-view geometry
- Visual place recognition
- Event-based Vision
- Dense reconstruction
- Visual inertial fusion

# Multiple View Geometry

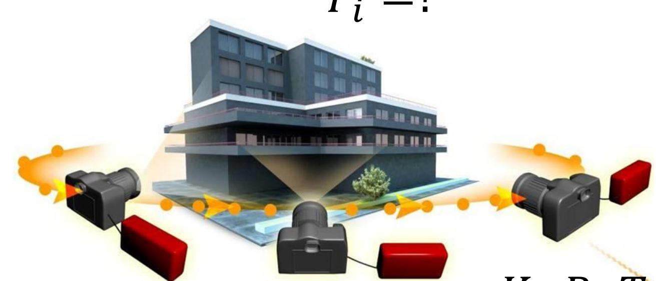
# Multiple View Geometry

**San Marco square, Venice**
14,079 images, 4,515,157 points

# Multiple View Geometry

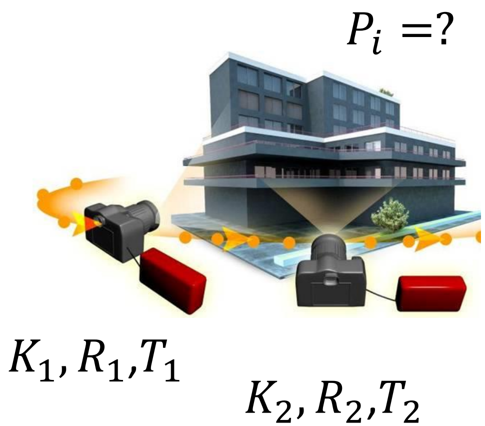
- **3D reconstruction from multiple views:**
  - **Assumptions:** K, T and R are known.
  - **Goal**: Recover the 3D structure from images


- **Structure From Motion:**
  - **Assumptions**: none (K, T,  and R are unknown).
  - **Goal**: Recover simultaneously 3D scene structure and camera poses (up to scale) from multiple images

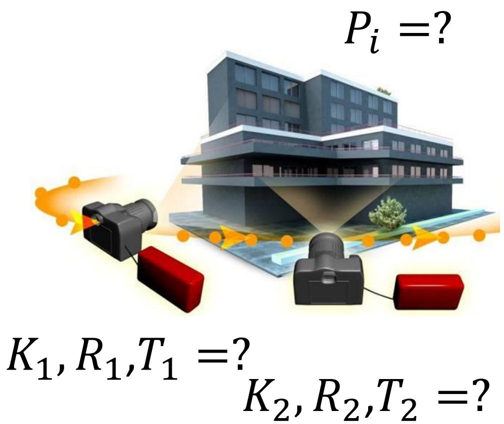


# 2-View Geometry

- **Depth from stereo (i.e., stereo vision)**
  - **Assumptions:** K, T and R are known.
  - **Goal**: Recover the 3D structure from images




- **2-view Structure From Motion:**
  - **Assumptions**: none (K, T,  and R are unknown).
  - **Goal**: Recover simultaneously 3D scene structure, camera poses (up to scale), and intrinsic parameters from two different views of the scene

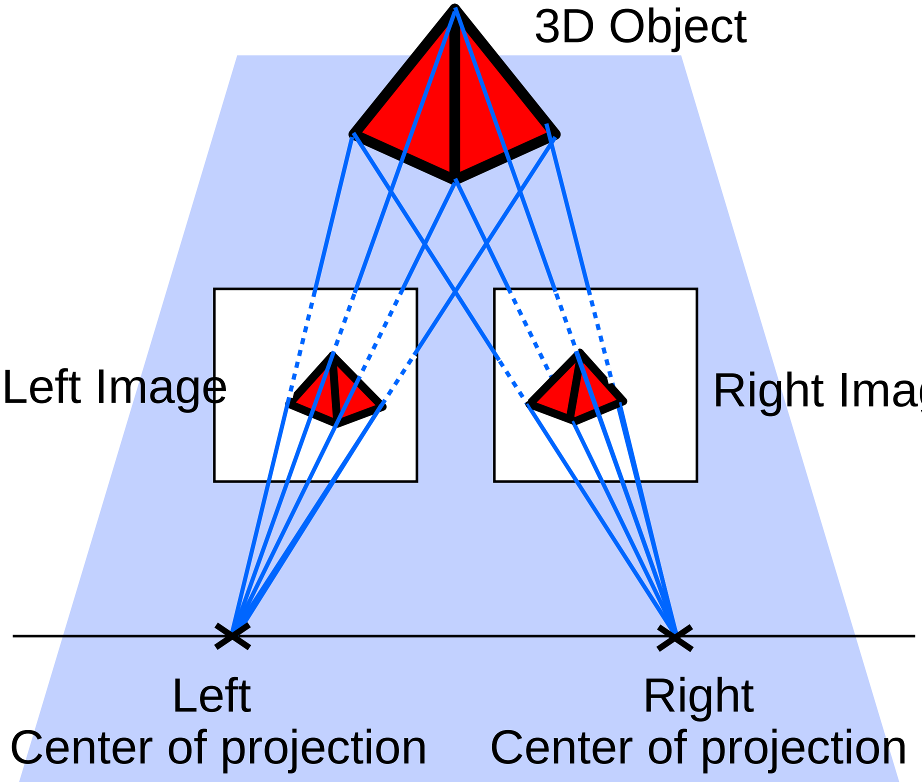


# Today's outline

- Stereo Vision
- Epipolar Geometry

# Depth from Stereo

- From a single camera, we can only compute the **ray** on which each image point lies
- With a stereo camera (binocular), we can solve for the intersection of the rays and recover the 3D structure



# The “human” binocular system

- **Stereopsys:** the brain allows us to see the left and right retinal images as a single 3D image
- The images project on our retina up-side-down but our brains lets us perceive them as «straight». Radial distortion is also removed. This process is called «**rectification**». What happens if you wear a pair of mirrors for a week?

Image from the left eye

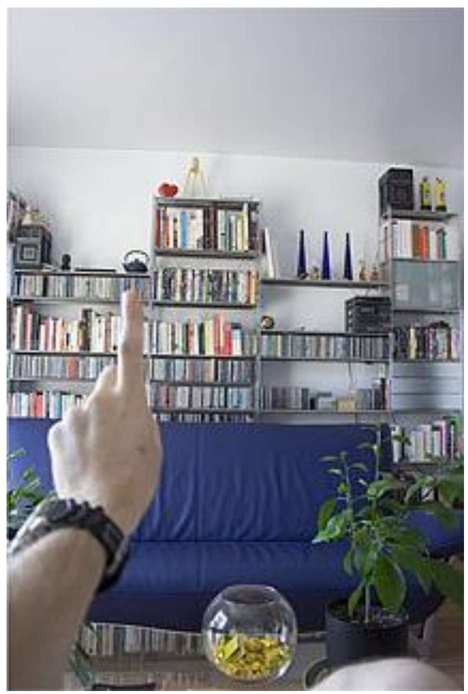
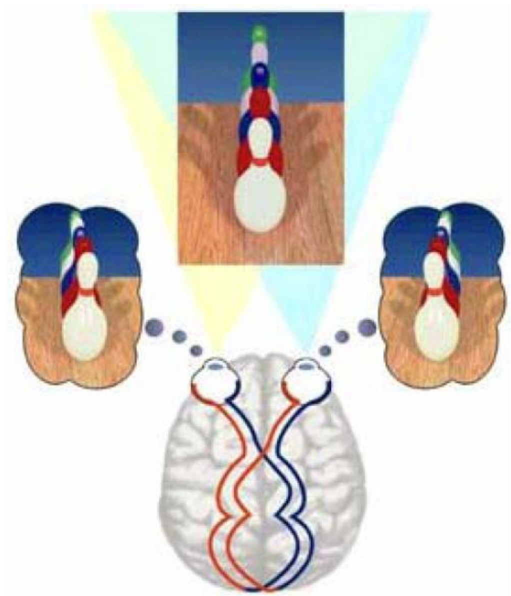
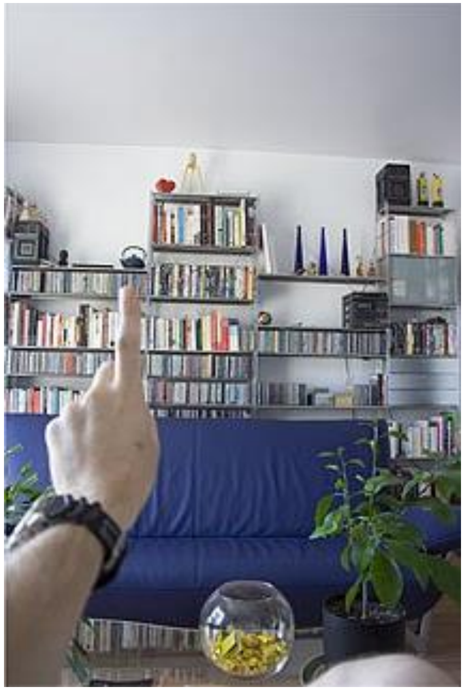


Image from the right eye



# The “human” binocular system

- **Stereopsys:** the brain allows us to see the left and right retinal images as a single 3D image
- The images project on our retina up-side-down but our brain lets us perceive them as «straight». Radial disotion is also removed. This process is called «**rectification**»



**Make a simple test:**

1. Fix an object
2. Open and close alternatively the left and right eyes.

- The horizontal displacement is called **disparity**
- The smaller the disparity, the farther the object

# The “human” binocular system

- **Stereopsys:** the brain allows us to see the left and right retinal images as a single 3D image
- The images project on our retina up-side-down but our brains lets us perceive them as «straight». Radial disotion is also removed. This process is called «**rectification**»


**Make a simple test:**

1. Fix an object
2. Open and close alternatively the left and right eyes.

- The horizontal displacement is called **disparity**
- The smaller the disparity, the farther the object

# Disparity

- The disparity between the left and right image allows us to perceive the depth

These animated GIF images display intermittently the left and right image

# Applications: Stereograms

Exploit disparity as depth cue using single image

Image from magiceye.com

# Applications: Stereograms

Exploit disparity as depth cue using single image

Image from magiceye.com

# Applications: Stereo photography and stereo viewers

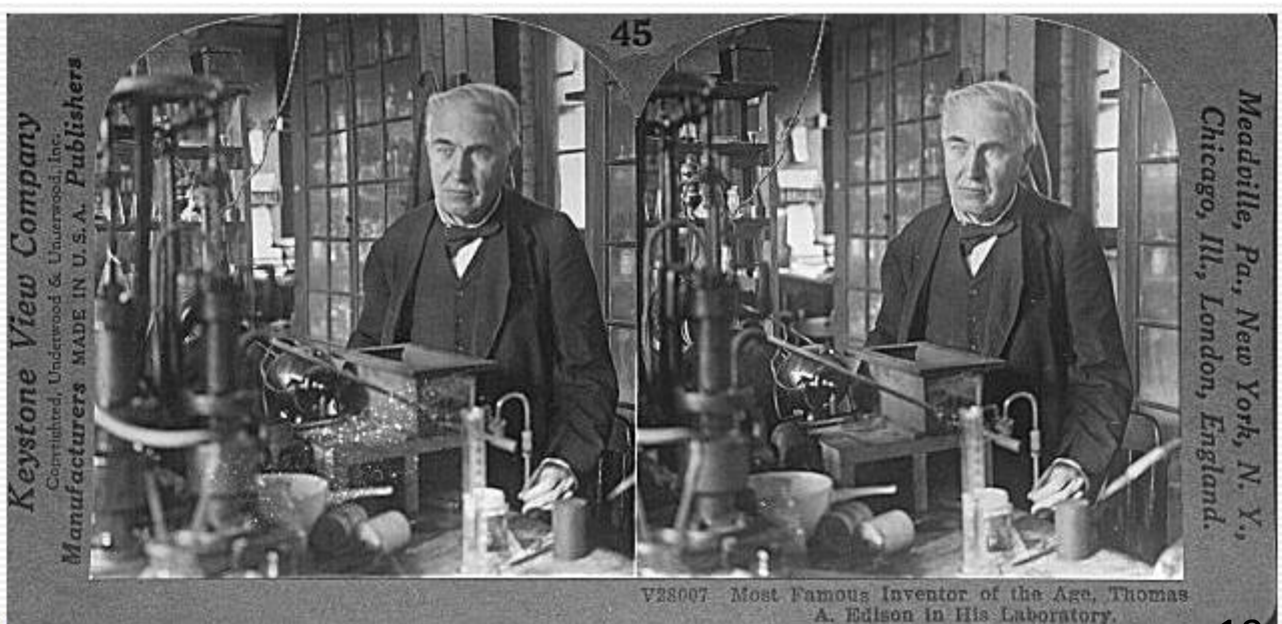
Take two pictures of the same subject from two different viewpoints and display them so that each eye sees only one of the images.

Invented by Sir Charles Wheatstone, 1838

# Applications: Anaglyphs

The first method to produce anaglyph images was developed in 1852 by Wilhelm Rollmann in Leipzig, Germany

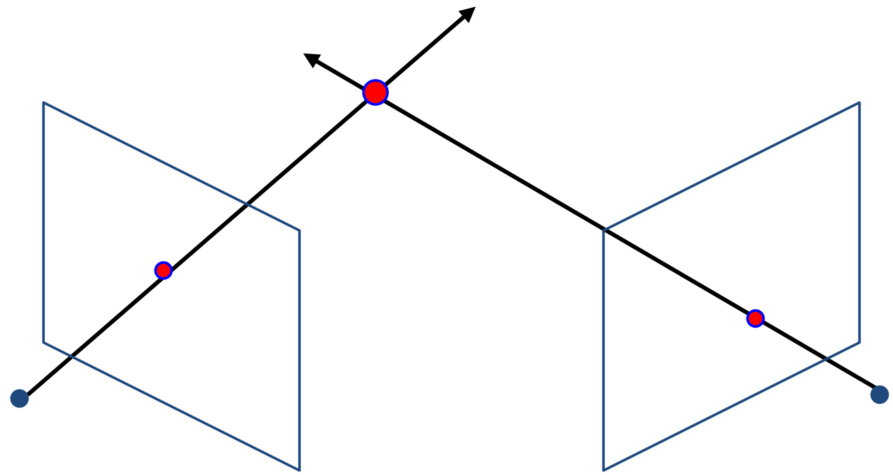
© Copyright 2001 Johnson-Shaw Stereoscopic Museum



# Stereo Vision

- Triangulation
  - Simplified case
  - General case
- Correspondence problem
- Stereo rectification

# Stereo Vision: basic idea



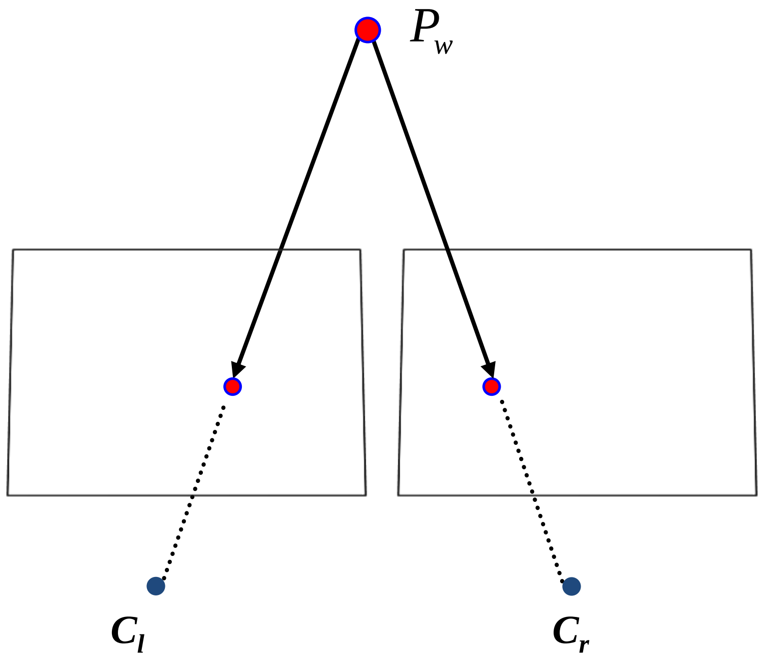
Basic Principle:  Triangulation

- Gives reconstruction as intersection of two rays
- Requires
  - camera pose (calibration)
  - point correspondence

# Stereo Vision: basic idea

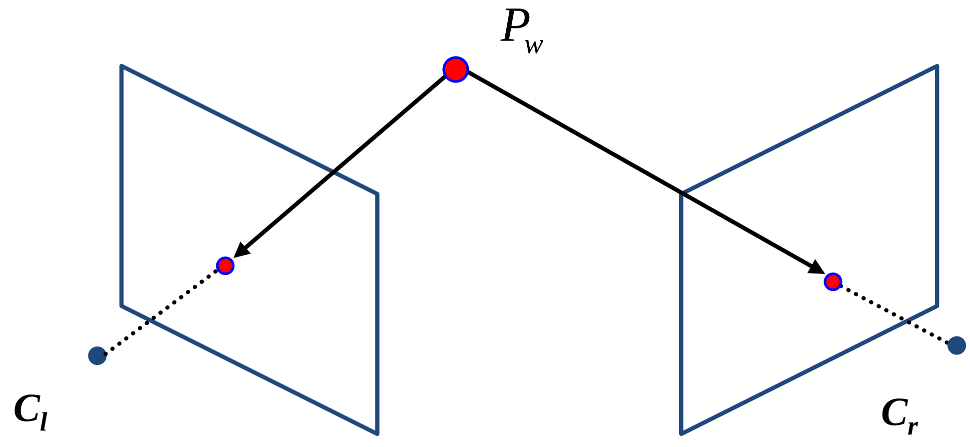
**Simplified case**

(identical cameras and aligned)



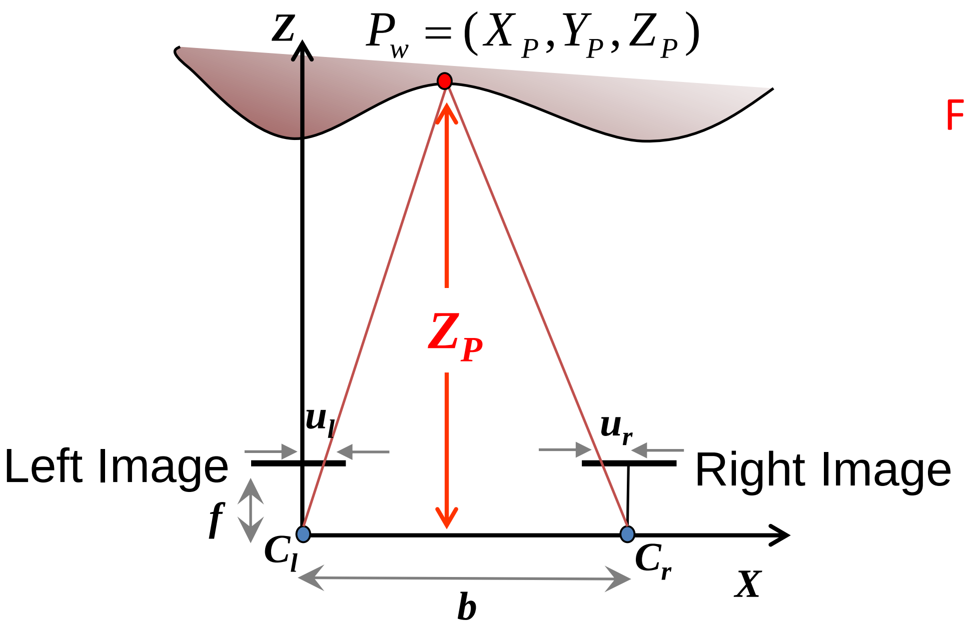

**General case**

(non identical cameras and not aligned)



# Stereo Vision - The simplified case

Both cameras are **identical** (i.e., same focal length) and are **aligned** with the x-axis

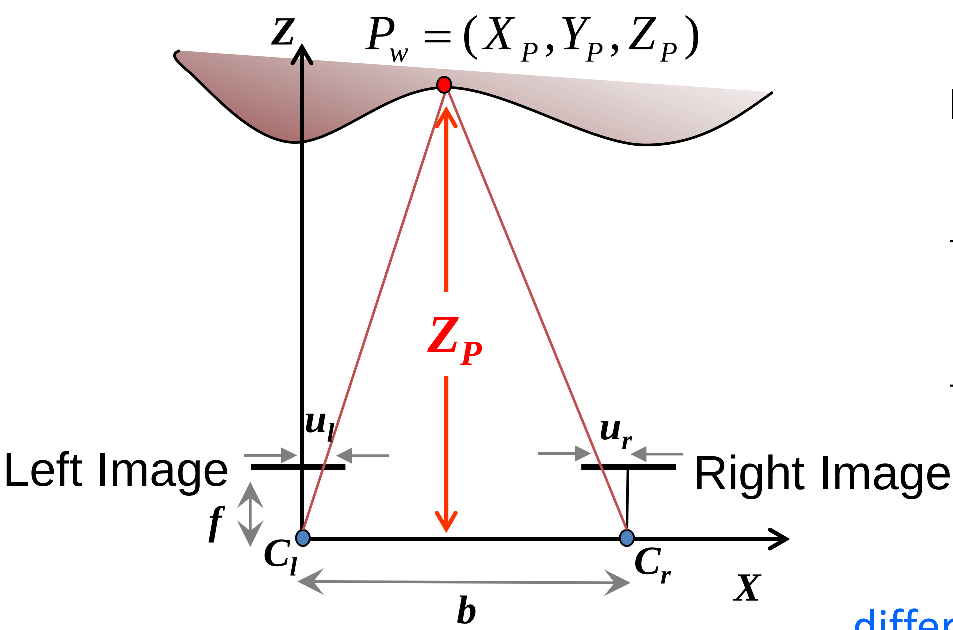



**Baseline**
distance between the optical centers of the two cameras

Find an expression for the depth $Z_P$ of point $P_w$

# Stereo Vision - The simplified case

Both cameras are **identical** and are **aligned** with the x-axis




From Similar Triangles:

$$\frac{f}{Z_P} = \frac{u_l}{X_P}$$

$$\frac{f}{Z_P} = \frac{-u_r}{b - X_P}$$

$$Z_P = \frac{bf}{\boxed{u_l - u_r}}$$

**Disparity**

difference in image location of the projection of a 3D point on two image planes

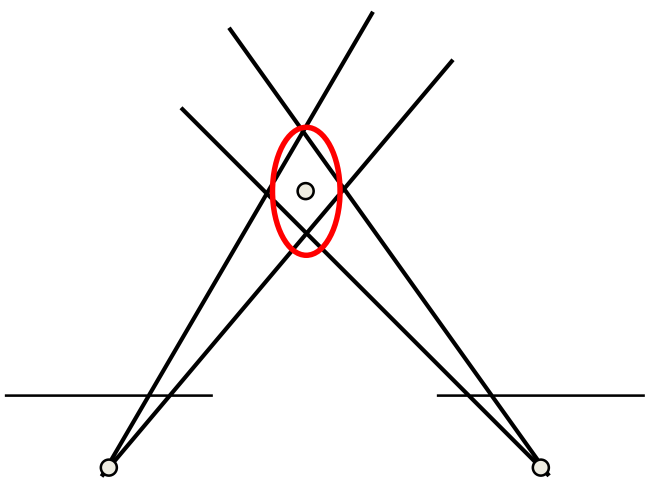
**Baseline**

distance between the optical centers of the two cameras

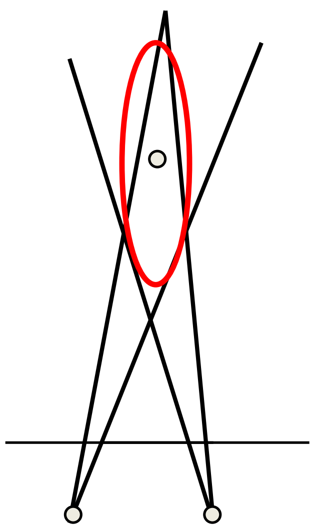
1. What's the max disparity of a stereo camera?
2. What's the disparity of a point at infinity?
3. How does the uncertainty of depth depend on the disparity?
4. And on the depth estimate?
5. How do I increase the accuracy of a stereo system?

# Choosing the Baseline

- What's the optimal baseline?
  - **Too small:**
    - Large depth error
    - Can you quantify the error as a function of the disparity?
  - **Too large:**
    - Minimum measurable distance increases
    - Difficult search problem for close objects



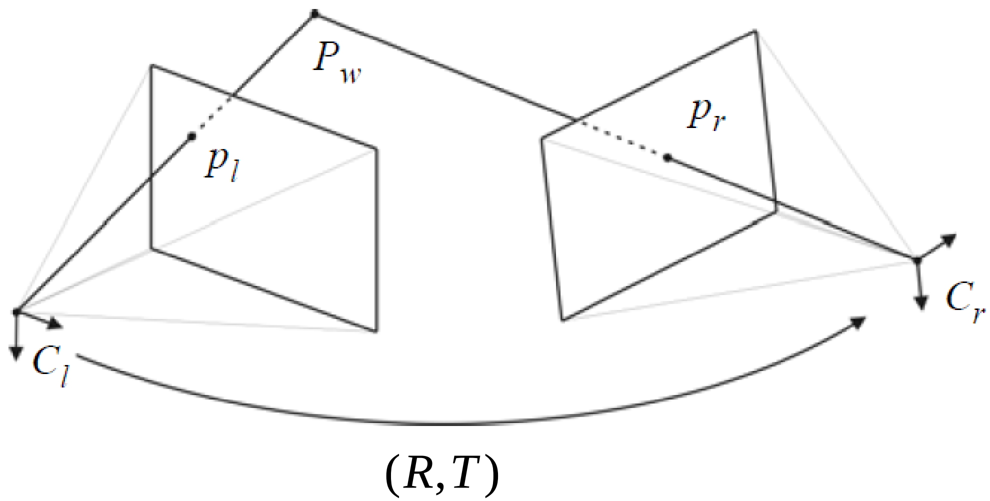
Large Baseline



Small Baseline

# Stereo Vision – the general case

- Two identical cameras do not exist in nature!
- Aligning both cameras on a horizontal axis is  very hard -> Impossible, why?



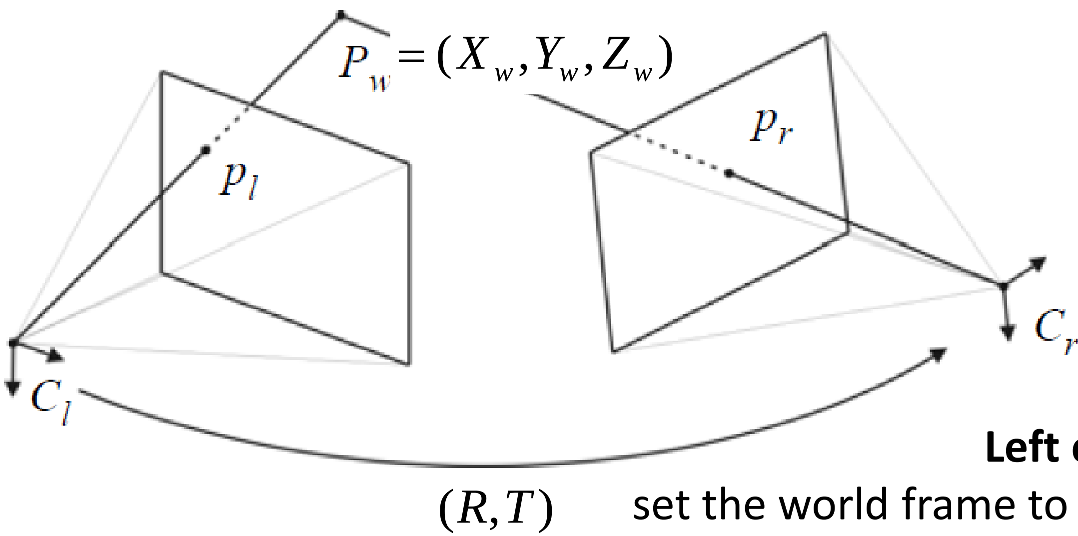

- In order to be able to use a stereo camera, we need the
  - **Extrinsic parameters** (relative rotation and translation)
  - **Instrinsic parameters** (focal length, optical center, radial distortion of each camera)

  ⇨ Use a calibration method (Tsai or Homographies, see Lectures 2, 3)

  ⇨ How do we compute the relative pose?

# Stereo Vision – the general case

- To estimate the 3D position of $P_w$ we construct the system of equations of the left and right cameras, and solve it. Do lines always intersect in the 3D space?




**Left camera:**
set the world frame to coincide with the left camera frame

$$\tilde{p}_l = \lambda_l \begin{bmatrix} u_l \\ v_l \\ 1 \end{bmatrix} = K_l \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix}$$

**Right camera:**

$$\tilde{p}_r = \lambda_r \begin{bmatrix} u_r \\ v_r \\ 1 \end{bmatrix} = K_r R \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} + T$$

- "Triangulation": the problem of determining the 3D position of a point given a set of corresponding image locations and known camera poses.

# Triangulation: least-squares approximation

Left camera

$$\lambda_1 \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} = K\left[I|0\right] \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \Rightarrow \lambda_1 p_1 = M_1 \cdot P$$

Right camera

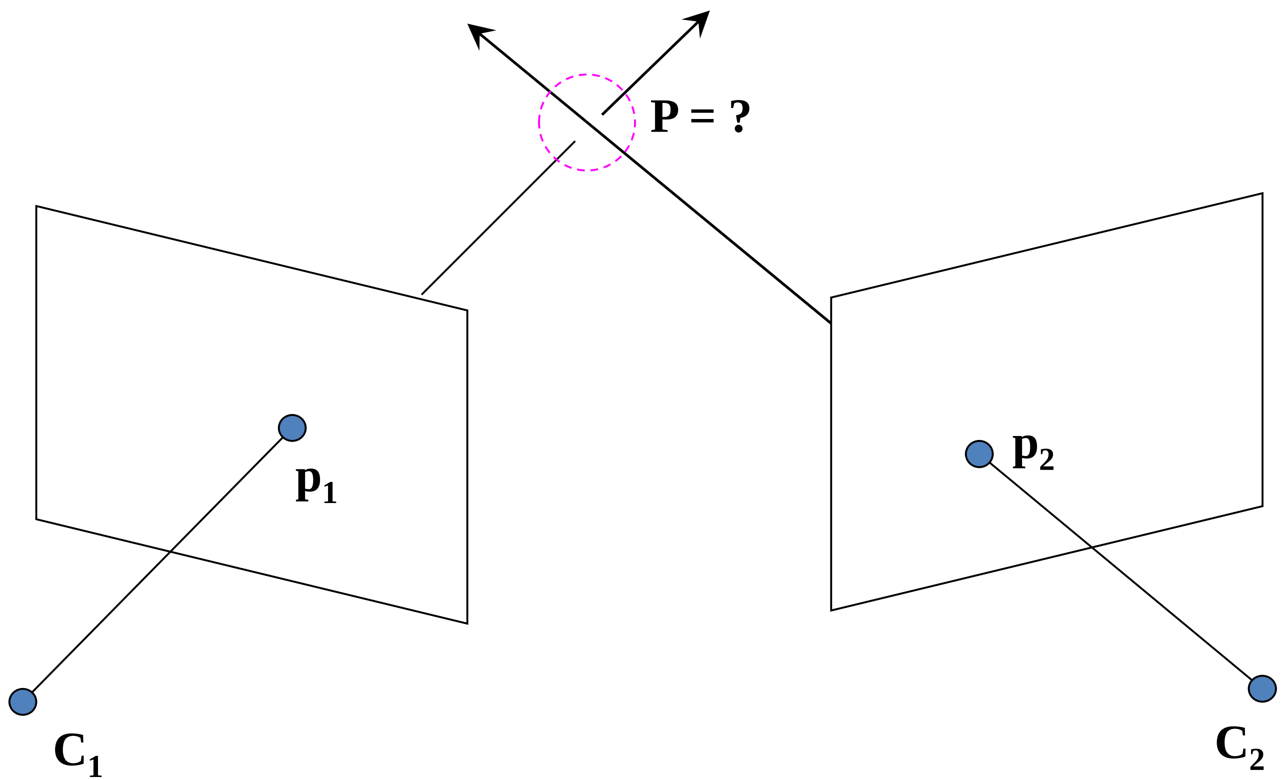
$$\lambda_2 \begin{bmatrix} u_2 \\ v_2 \\ 1 \end{bmatrix} = K\left[R|T\right] \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \Rightarrow \lambda_2 p_2 = M_2 \cdot P$$

by solving for P, we arrive to a system of the type $\mathbf{A}\begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = \mathbf{b}$ , which cannot be inverted (A is 3x2 matrix). However, can be solved using the pseudoinverse approximation:

$$A^T A \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = A^T \mathrm{b}, \text{ and so } \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = (A^T A)^{-1} A^T \mathrm{b}$$

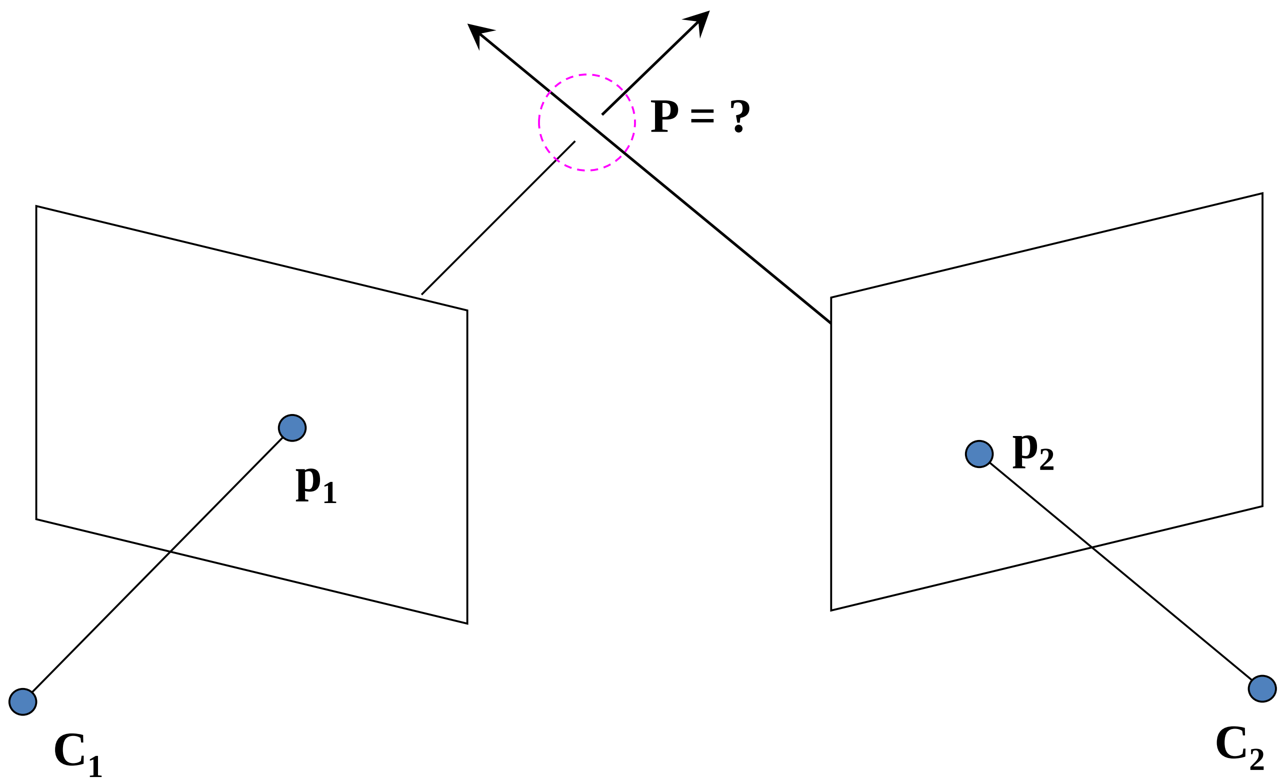
# Triangulation: geometric interpretation

- Given the projections $\mathbf{p_1}$ and $\mathbf{p_2}$ of a 3D point $\boldsymbol{P}$ in two or more images (with known camera matrices R and T), find the coordinates of the 3D point



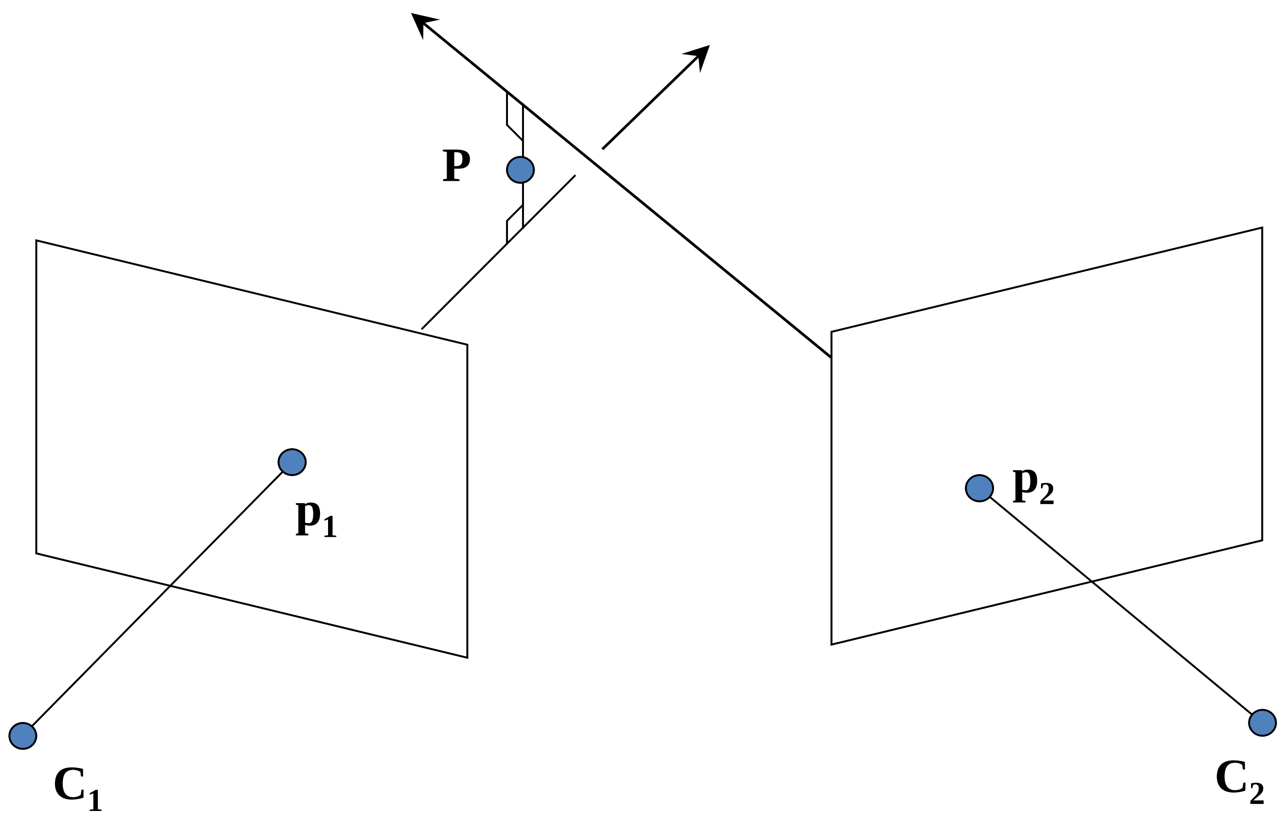
# Triangulation: geometric interpretation

- We want to intersect the two visual rays corresponding to $\mathbf{p_1}$ and $\mathbf{p_2}$, but because of noise and numerical errors, they don't meet exactly



# Triangulation: geometric interpretation

- Find shortest segment connecting the two viewing rays and let **P** be the midpoint of that segment



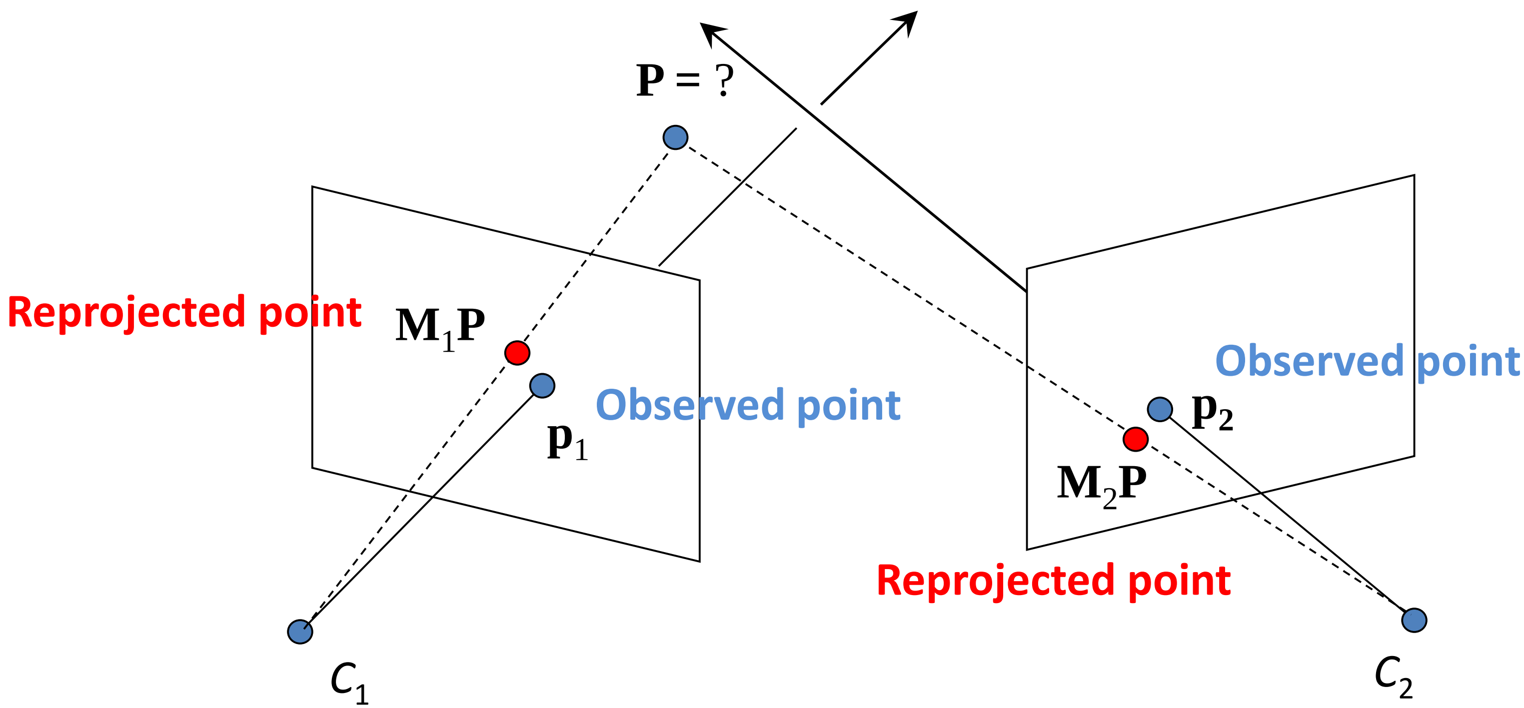
# Triangulation: Nonlinear approach

- Find $P$ that minimizes the **Sum of Squared Reprojection Error**:

$$SSRE = d^2(p_1, \pi_1(P)) + d^2(p_2, \pi_2(P))$$

where $d(p_1, \pi_1(P)) = \left\| p_1 - \pi_1(P) \right\|$ is called ***Reprojection Error***.

- In practice, initialize ***P*** using linear approach and then minimize SSRE using Gauss-Newton or Levenberg-Marquardt.



# Stereo Vision

- Triangulation
  - Simplified case
  - General case
- Correspondence problem
- Stereo rectification

# Correspondence Problem

Given the point $p$ in left image, where is its corresponding point $p$’ in the right image?


Left image


Right image

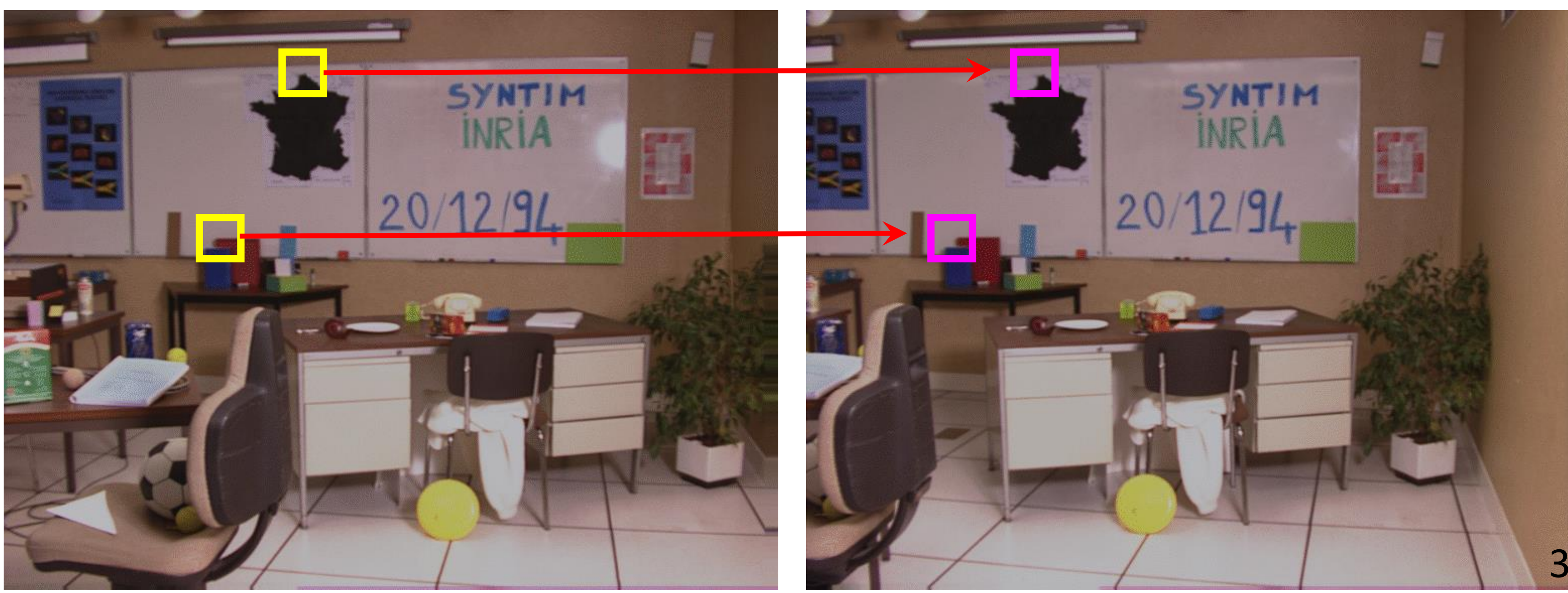
# Correspondence Problem

Given the point $p$ in left image, where is its corresponding point $p$' in the right image?


Left image | Right image

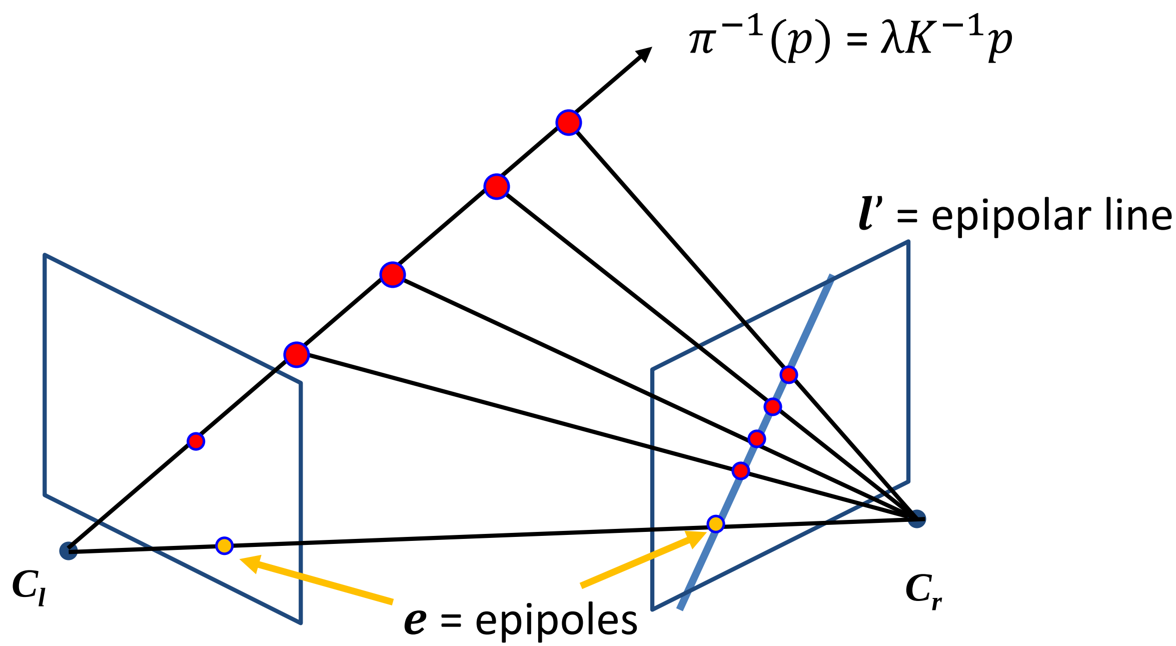
# Correspondence Problem

- **Correspondence search**: identify image patches in the left & right images, corresponding to the same scene structure.
- **Similarity measures**:
  - **(Z)ZNCC**
  - **(Z)SSD**
  - **(Z)SAD**
  - **Census Transform**



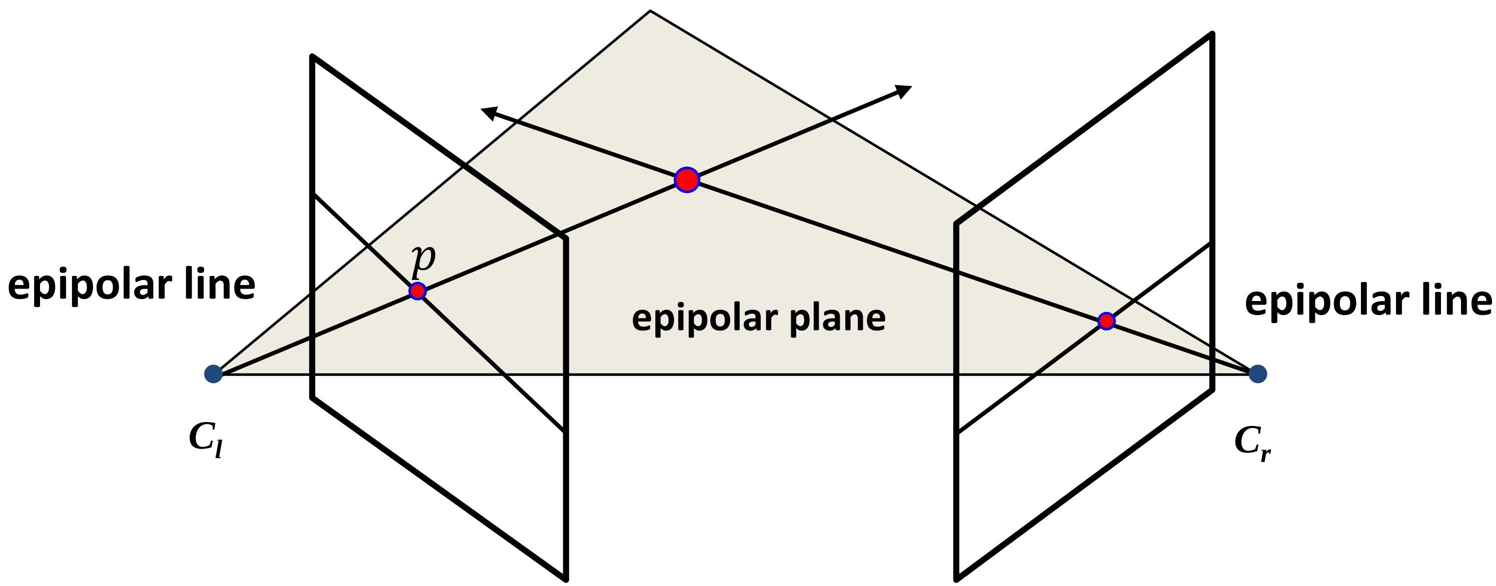
# Correspondence Problem

- **Exhaustive** image search can be computationally very expensive!
- Can we make the correspondence search in 1D?
- Potential matches for $\boldsymbol{p}$ have to lie on the corresponding epipolar line $\boldsymbol{l'}$
  - The **epipolar line** is the projection of the infinite ray $\pi^{-1}(p)$ corresponding to $\boldsymbol{p}$ in the other camera image
  - The **epipole** is the projection of the optical center on the other camera image
  - A stereo camera has two epipoles



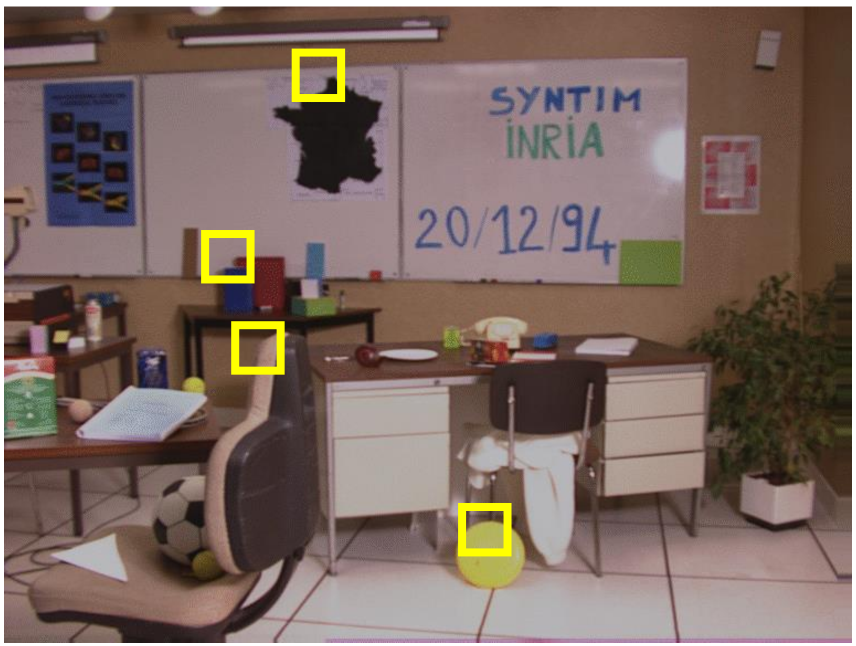
# The Epipolar Constraint

- The epipolar plane is uniquely defined by the two optical centers $C_l, C_r$ and one image point $p$
- The **epipolar constraint** constrains the location, in the second view, of the corresponding point to a given point in the first view.
- Why is this useful?
  - Reduces correspondence problem to 1D search along *conjugate epipolar lines*

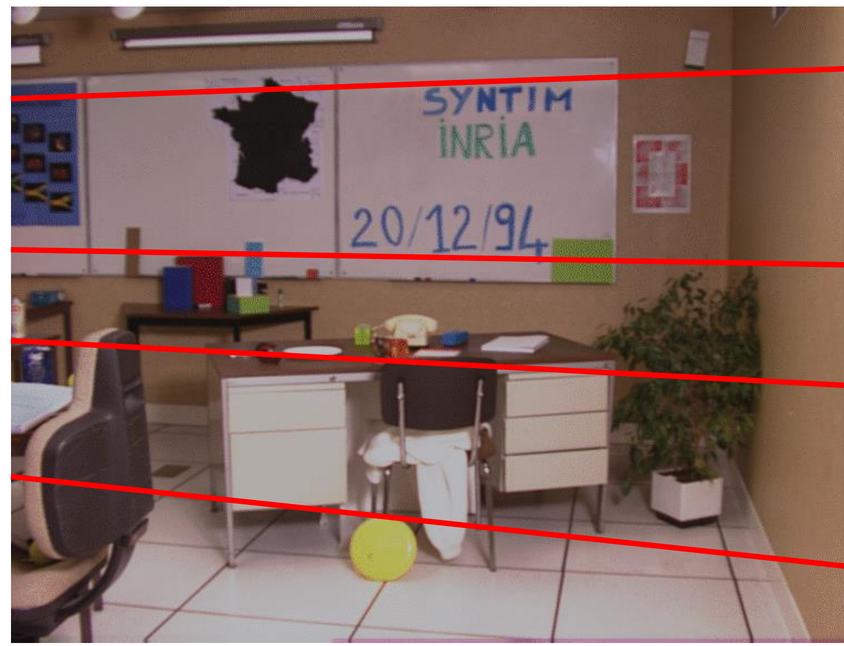


# Correspondence Problem: Epipolar Constraint

Thanks to the epipolar constraint, corresponding points can be searched for, along epipolar lines: ⇨ computational cost reduced to 1 dimension!



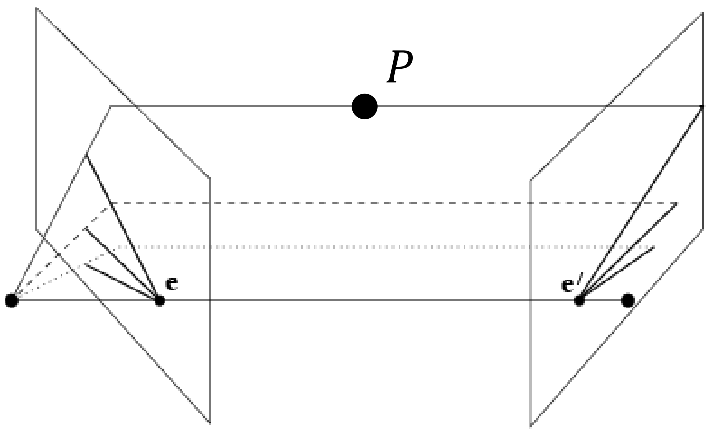

Left image

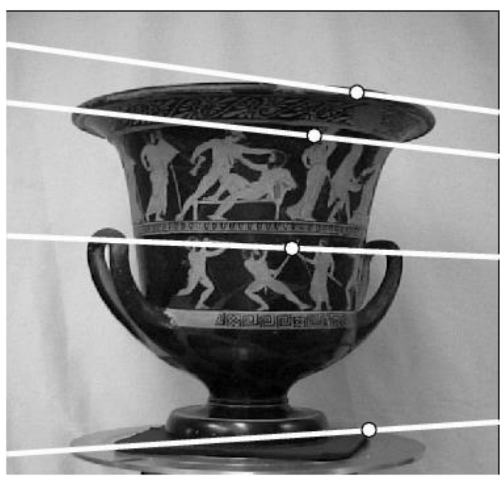



Right image

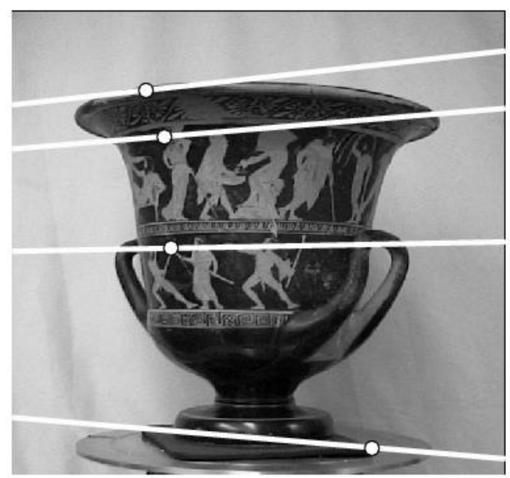
# Example: converging cameras

- **Remember**: all the epipolar lines intersect at the epipole
- As the position of the 3D point varies, the epipolar lines "rotate" about the baseline



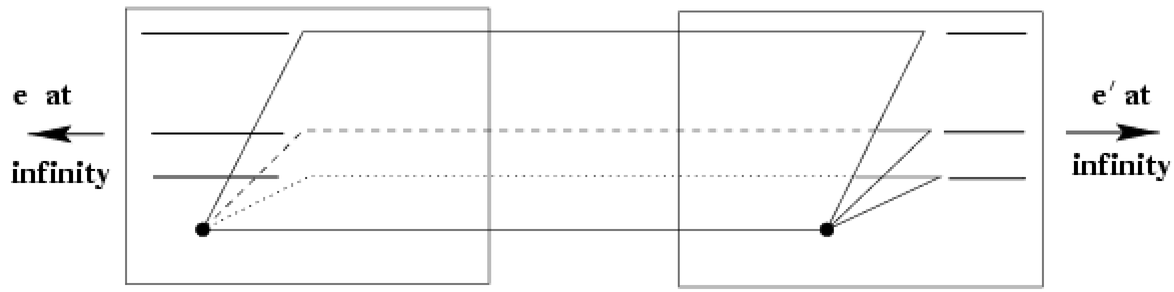



Left image



Right image

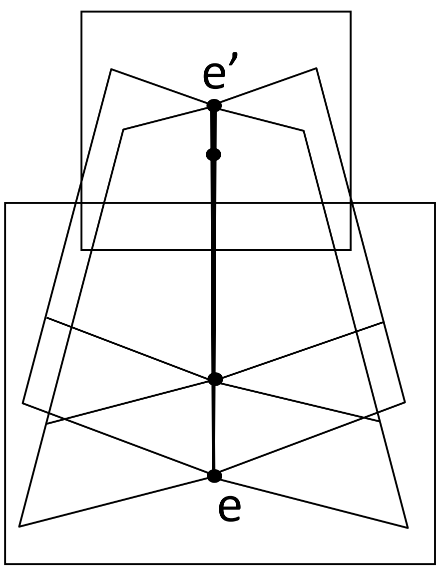
# Example: identical and horizontally-aligned cameras

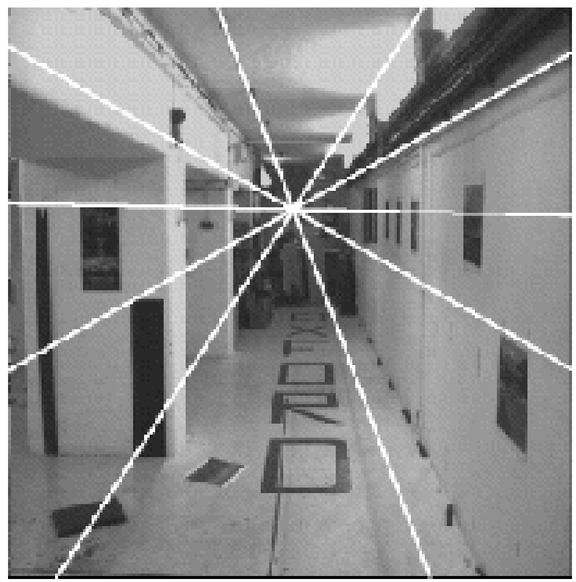



Left image Right image

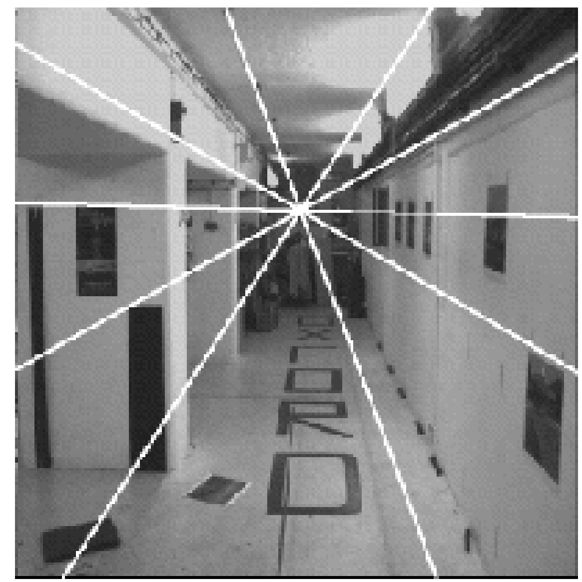
# Example: forward motion (parallel to the optical axis)

- Epipole has the **same coordinates** in both images
- Points move along lines radiating from e: “Focus of expansion”



Left image



Right image

# Stereo Vision

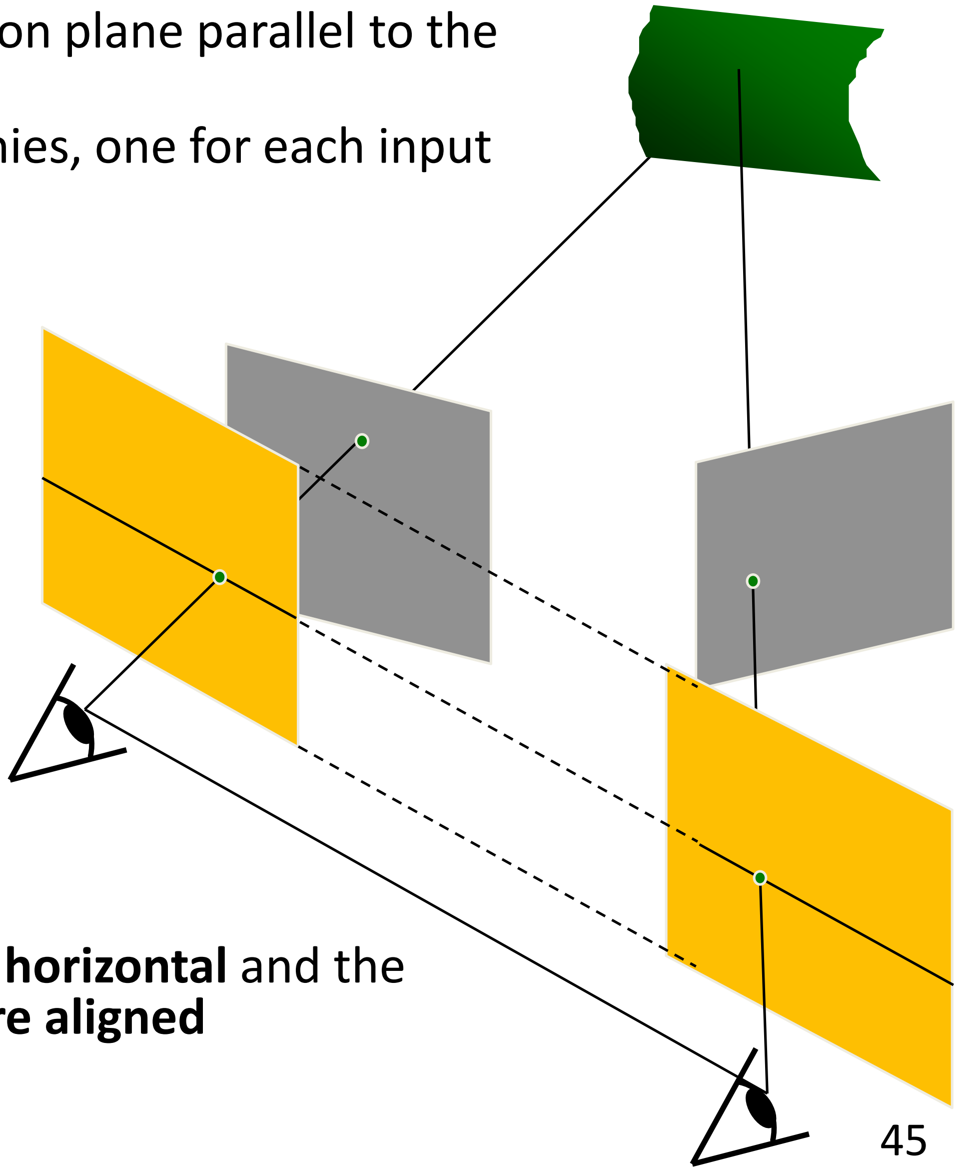
- Simplified case
- General case
- Correspondence problem
- Stereo rectification
- Triangulation

# Stereo Rectification

- Even in commercial stereo cameras the left and right images are never perfectly aligned.
- In practice,  it is convenient if image scanlines are the epipolar lines.
- Stereo rectification warps the left and right images into new “rectified” images, whose epipolar lines are aligned to the baseline.

# Stereo Rectification

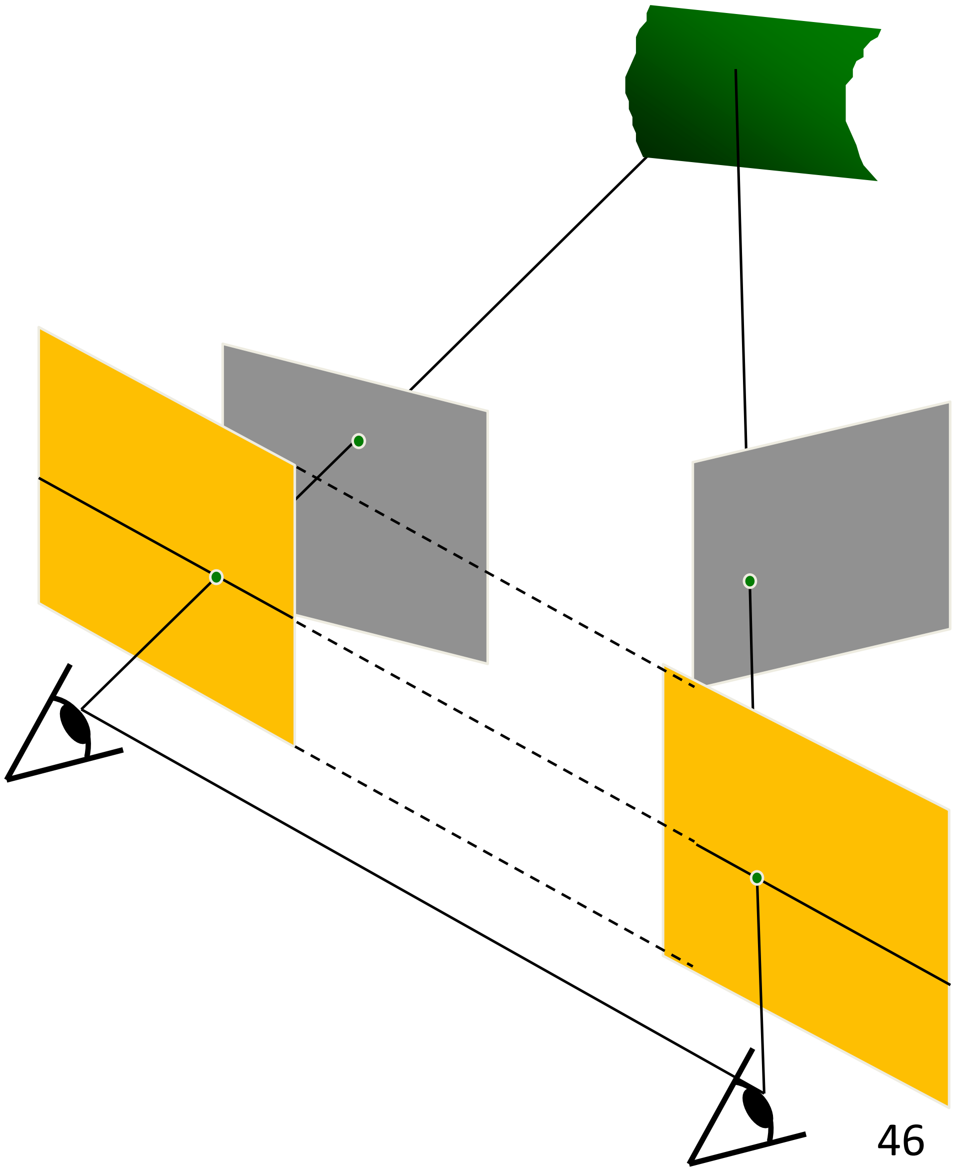
- Reprojects image planes onto a common plane parallel to the baseline
- It works by computing two homographies, one for each input image reprojection



- As a result, the new **epipolar lines** are **horizontal** and the **scanlines** of the left and right image **are aligned**

# Stereo Rectification

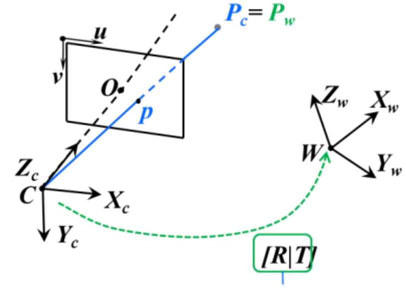
- The idea behind rectification is to define two new Perspective Projection Matrices obtained by rotating the old ones around their optical centers until focal planes becomes coplanar, thereby containing the baseline.
- This ensures that epipoles are at infinity, hence epipolar lines are parallel.
- To have horizontal epipolar lines, the baseline must be parallel to the new X axis of both cameras.
- In addition, to have a proper rectification, corresponding points must have the same vertical coordinate. This is obtained by requiring that the new cameras have the same intrinsic parameters.
- Note that, being the focal length the same, the new image planes are coplanar too
- PPMs are the same as the old cameras, whereas the new orientation (the same for both cameras) differs from the old ones by suitable rotations; intrinsic parameters are the same; for both cameras.



# Stereo Rectification (1/5)

We have seen in Lecture 02 that the Perspective Equation for a point $P_w$ in the world frame is

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K[R|T] \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

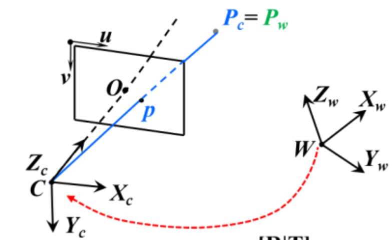


Extrinsic Parameters

where $[R|T]$ is the transformation from the Camera to the World's frame.
This can be re-written in a more convenient way by considering $[\mathrm{R}|\mathrm{T}]$ (or $[\mathrm{R}|\mathrm{C}]$) as the transformation from the World to the Camera frame:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = KR^{-1} \left[ \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} - T \right]$$

$$\boxed{\Rightarrow \quad \lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = KR^{-1} \left[ \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} - C \right]}$$



Extrinsic Parameters

# Stereo Rectification (2/5)

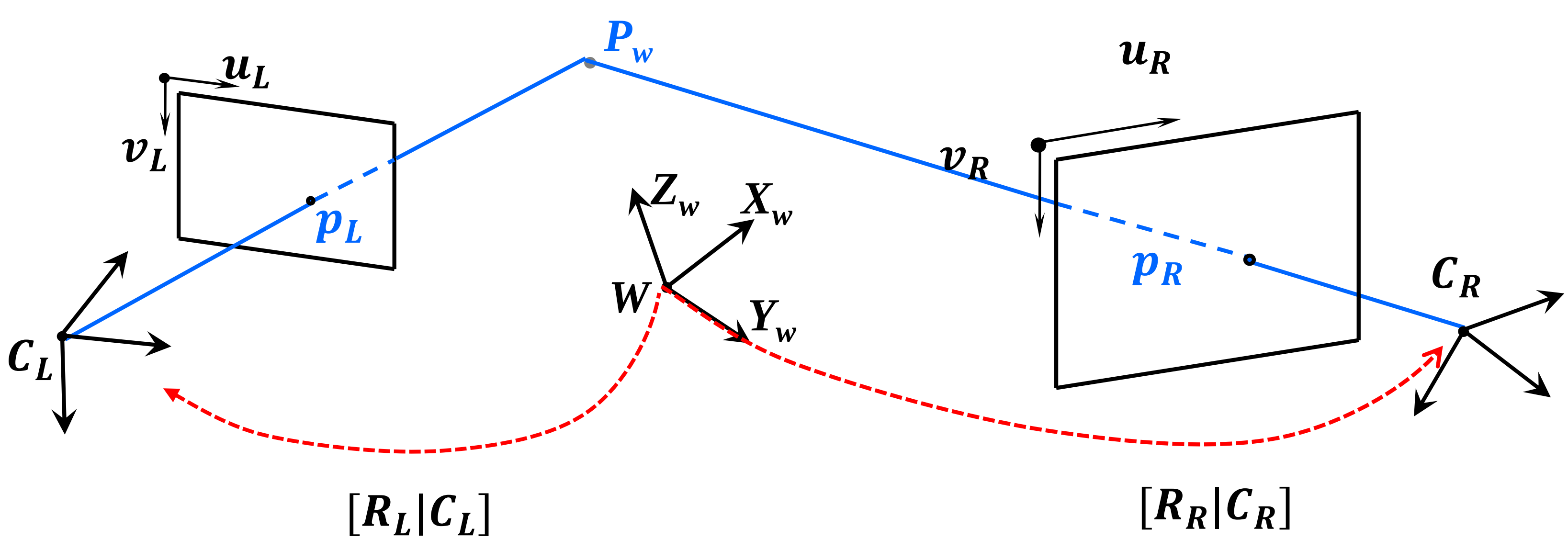
We can therefore write the Perspective Equation for the Left and Right cameras. For generality, we assume that Left and Right cameras have different intrinsic parameters (see also illustration below).

Left camera

$$\lambda_L \begin{bmatrix} u_L \\ v_L \\ 1 \end{bmatrix} = K_L R_L^{-1} \left[ \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} - C_L \right]$$

Right camera

$$\lambda_R \begin{bmatrix} u_R \\ v_R \\ 1 \end{bmatrix} = K_R R_R^{-1} \left[ \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} - C_R \right]$$



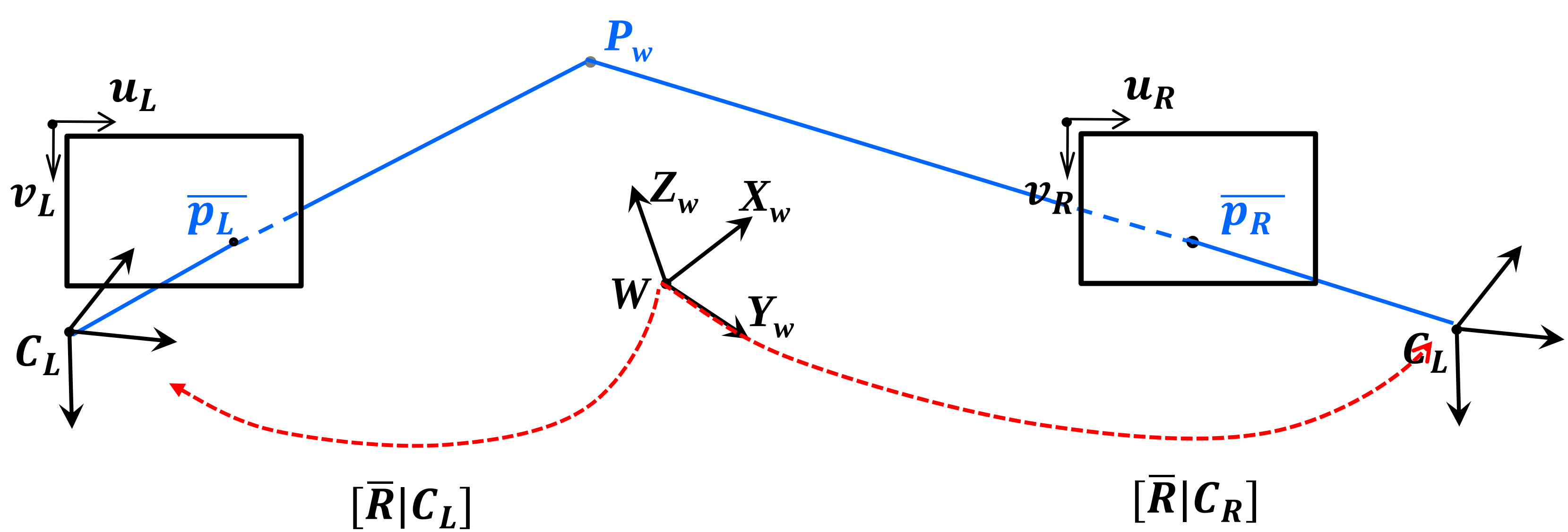
# Stereo Rectification (3/5)

The goal of stereo rectification is to warp the left and right camera images such that their focal planes are coplanar and their intrinsic parameters are identical.

| Old Left camera | New Left camera | Old Right camera | New Right camera |
| --- | --- | --- | --- |

$$\lambda_L \begin{bmatrix} u_L \\ v_L \\ 1 \end{bmatrix} = K_L R_L^{-1} \left( \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} - C_L \right) \rightarrow \bar{\lambda}_L \begin{bmatrix} \bar{u}_L \\ \bar{v}_L \\ 1 \end{bmatrix} = \bar{K} \bar{R}^{-1} \left( \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} - C_L \right)$$

$$\lambda_R \begin{bmatrix} u_R \\ v_R \\ 1 \end{bmatrix} = K_R R_R^{-1} \left( \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} - C_R \right) \rightarrow \bar{\lambda}_R \begin{bmatrix} \bar{u}_R \\ \bar{v}_R \\ 1 \end{bmatrix} = \bar{K} \bar{R}^{-1} \left( \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} - C_R \right)$$

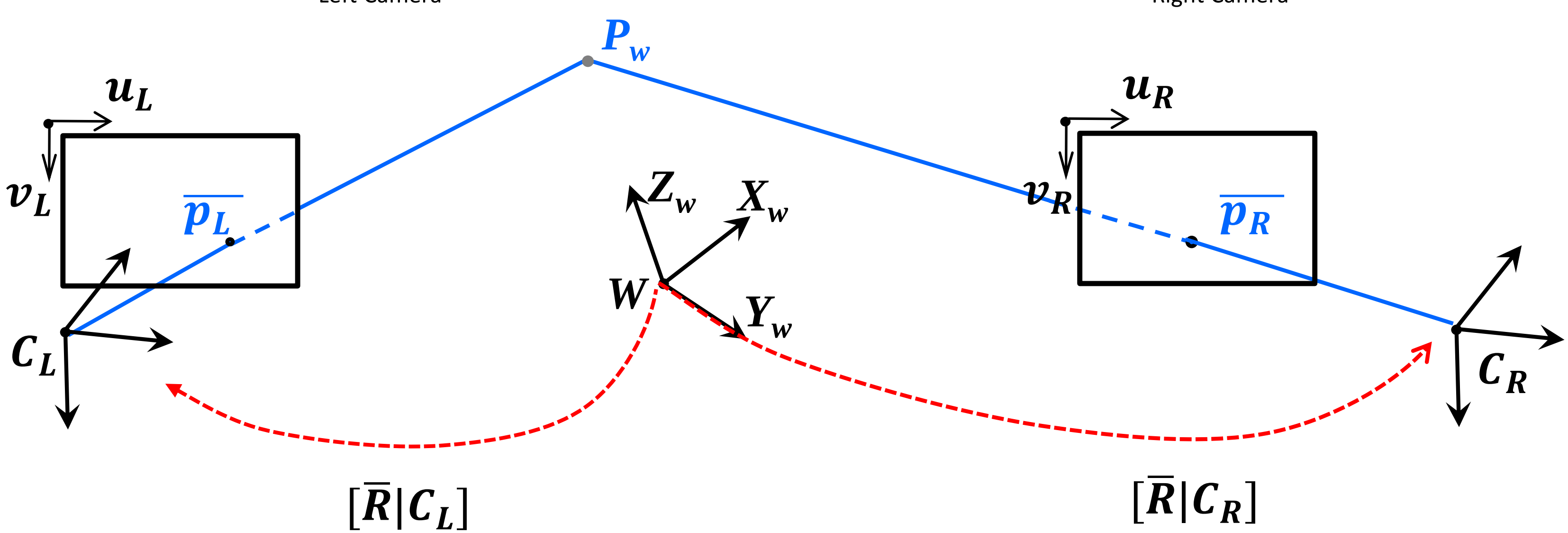


# Stereo Rectification (4/5)

By solving for $\begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix}$ for each camera, we can compute the Homography (or Warping) that needs to be applied to rectify each camera image

$$\bar{\lambda}_L \begin{bmatrix} \bar{u}_L \\ \bar{v}_L \\ 1 \end{bmatrix} = \lambda_L \underbrace{\bar{K}\bar{R}^{-1} \mathrm{R}_L \mathrm{K}_L^{-1}}_{\substack{\text{Homography} \\ \text{Left Camera}}} \begin{bmatrix} u_L \\ v_L \\ 1 \end{bmatrix}$$

$$\bar{\lambda}_R \begin{bmatrix} \bar{u}_R \\ \bar{v}_R \\ 1 \end{bmatrix} = \lambda_R \underbrace{\bar{K}\bar{R}^{-1} \mathrm{R}_R \mathrm{K}_R^{-1}}_{\substack{\text{Homography} \\ \text{Right Camera}}} \begin{bmatrix} u_R \\ v_R \\ 1 \end{bmatrix}$$



# Stereo Rectification (5/5)

How do we chose the new $\bar{K}$ and $\bar{R}$ ? A good choice is to impose that:

$$\bar{K} = (K_L + K_R)/2$$

$$\bar{R} = [\overline{r_1}, \overline{r_2}, \overline{r_3}]$$

with $\overline{r_1}, \overline{r_2}, \overline{r_3}$ being the column vectors of $\bar{R}$, where

$$\overline{r_1} = \frac{C_2 - C_1}{\|C_2 - C_1\|}$$

$\overline{r_2} = r_3 \times \overline{r_1}$ , where $r_3$ is the 3rd column of the rotation matrix of the left camera, i.e., $R_L$

$$\overline{r_3} = \overline{r_1} \times \overline{r_2}$$

More details can be found in the paper "A Compact Algorithm for Rectification of Stereo Pairs"

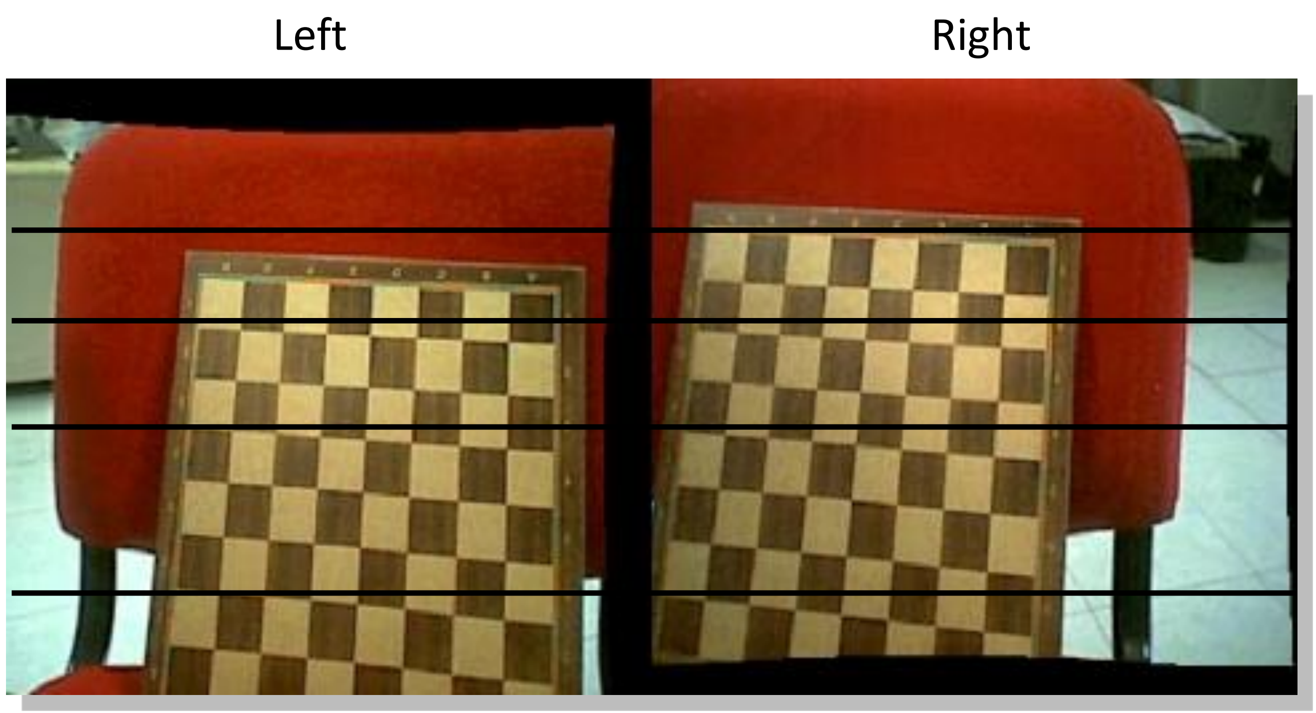
# Stereo Rectification: example

Left Right

scanlines

# Stereo Rectification: example

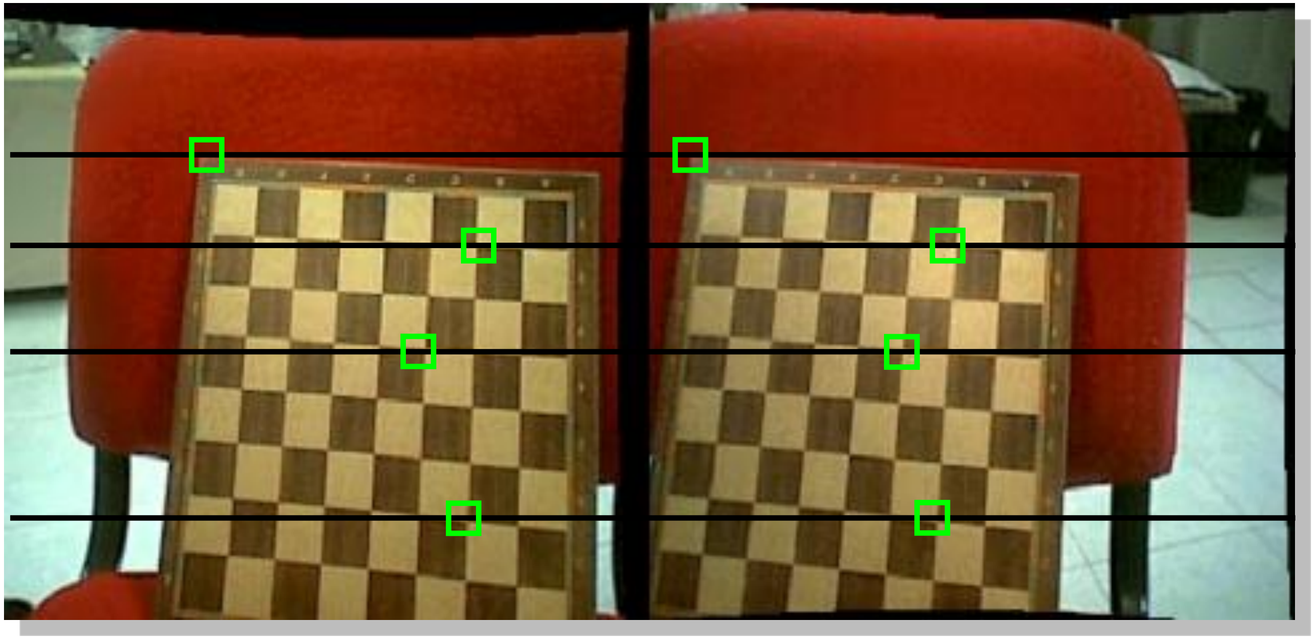
- First, remove radial distortion (use bilinear interpolation (see lect. 06))



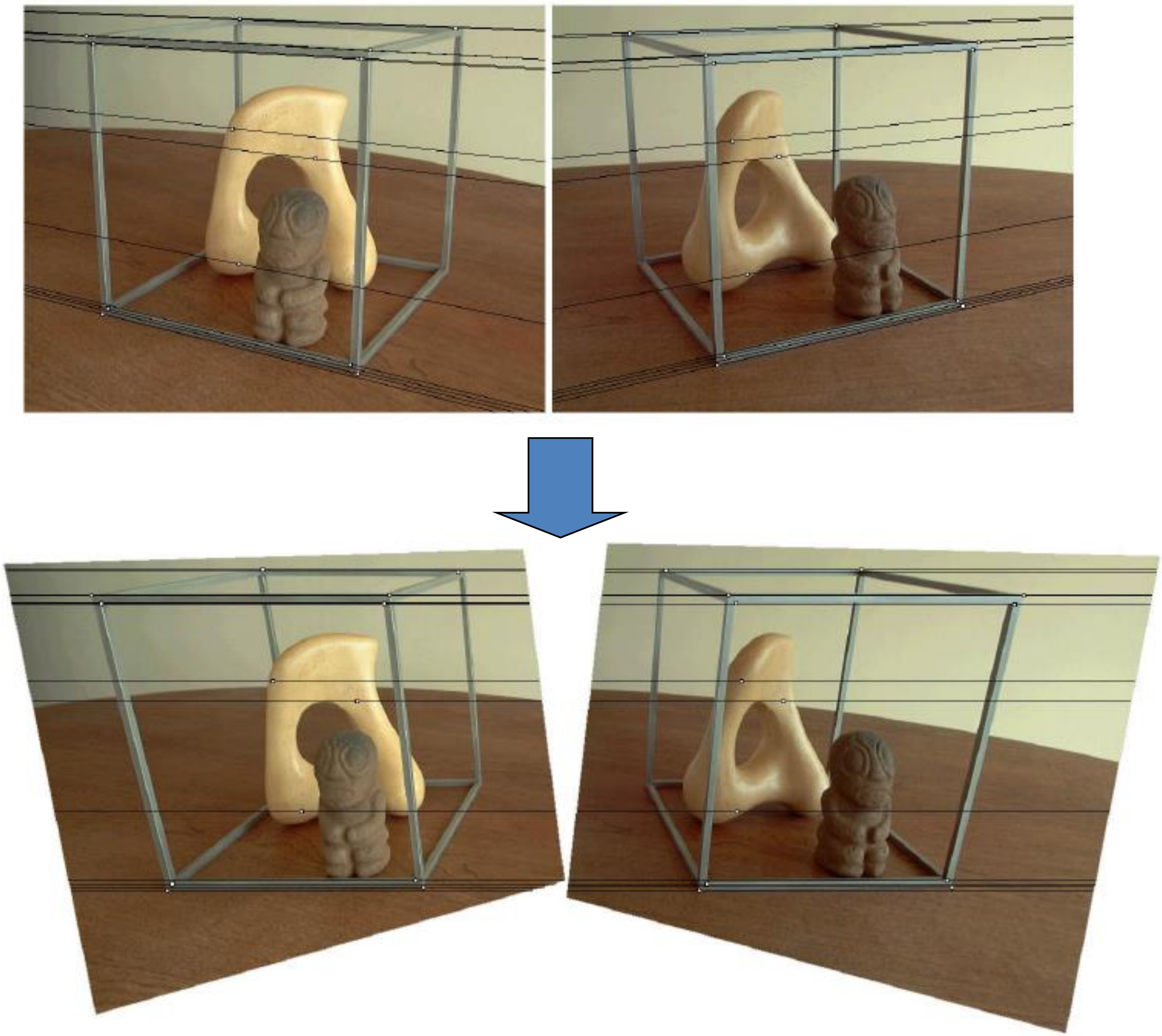
# Stereo Rectification: example

- First, remove radial distortion (use bilinear interpolation (see lect. 06))
- Then, compute homographies and rectify (use bilinear interpolation)

Left | Right



# Stereo Rectification: example

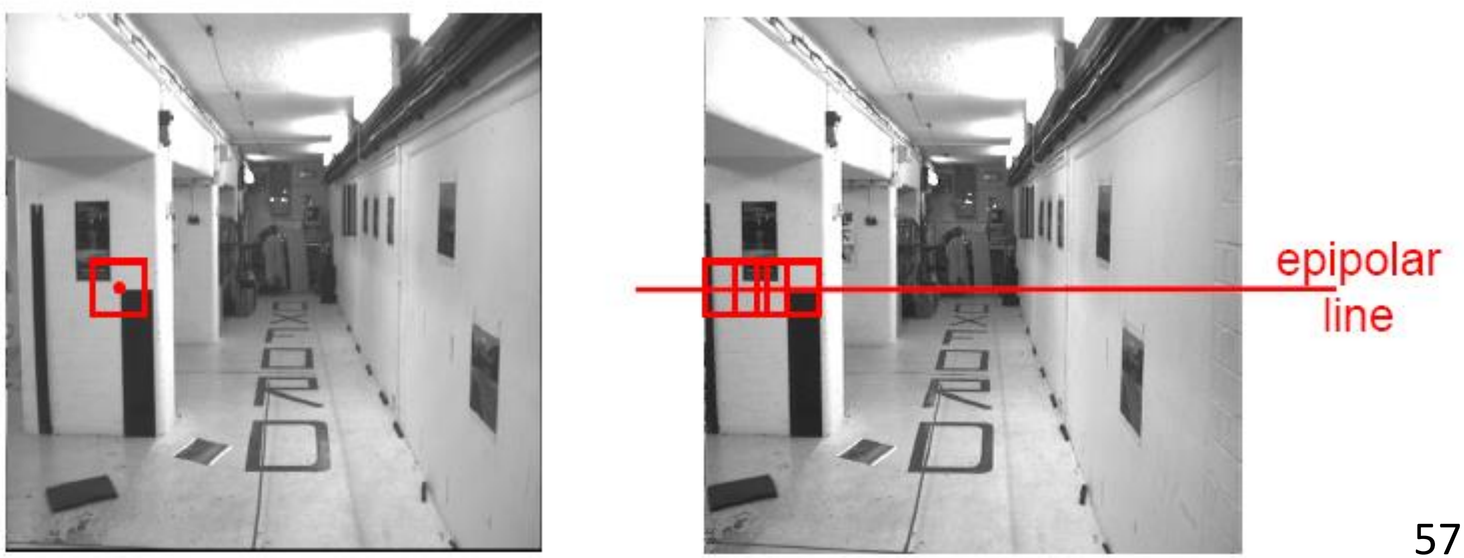


# Stereo Vision

- Simplified case
- General case
- Correspondence problem (continued)
- Stereo rectification
- Triangulation

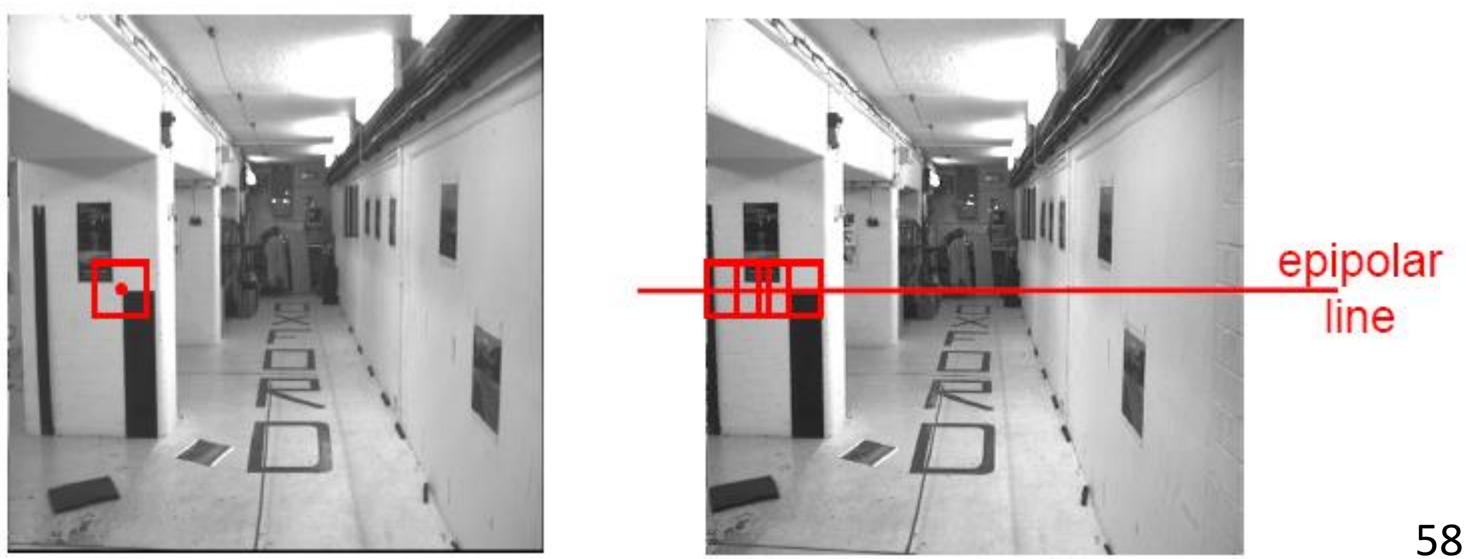
# Correspondence problem

- Now that the left and right images are rectified, the correspondence search can be done along the same scanlines

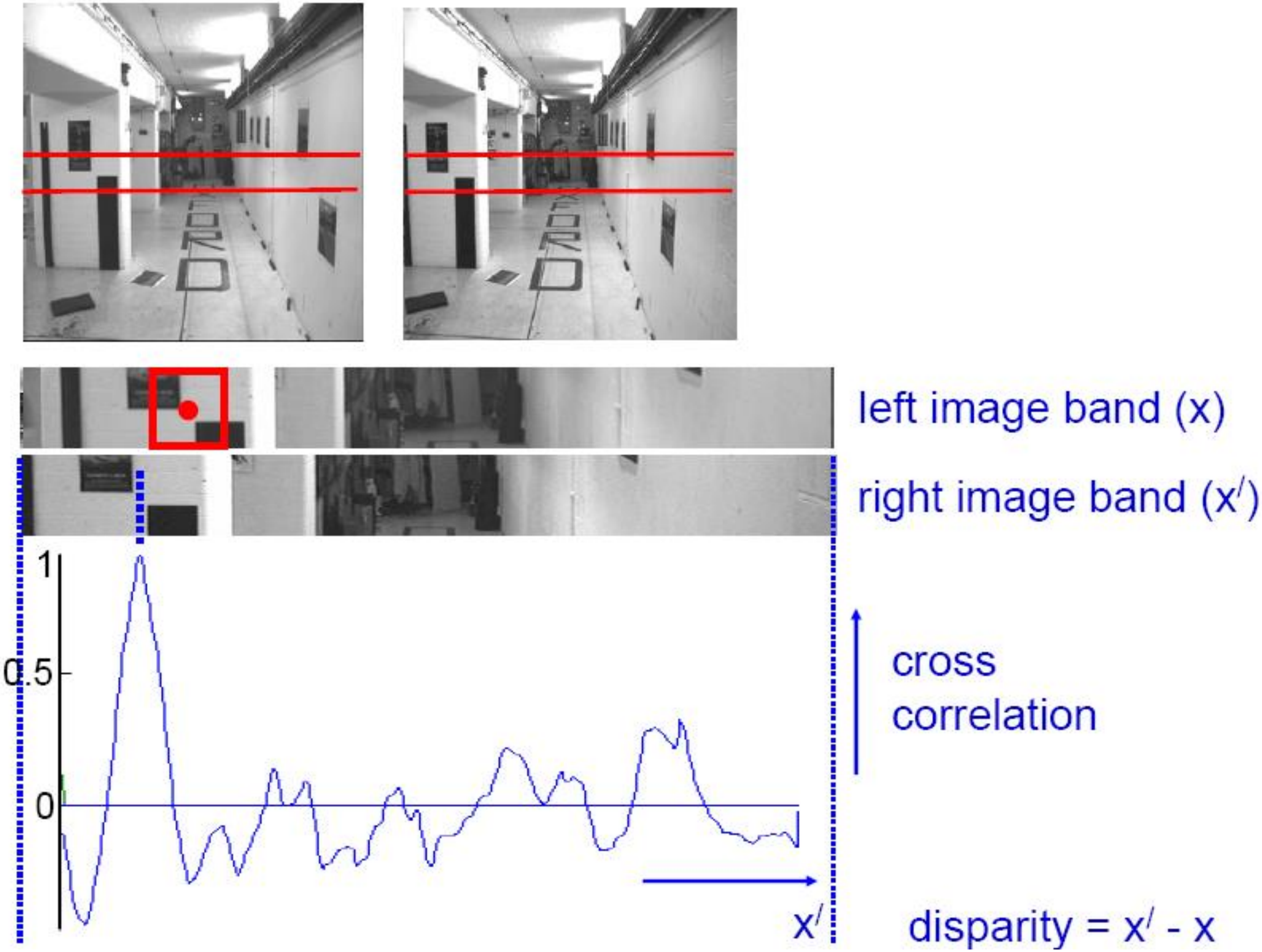


# Correspondence problem

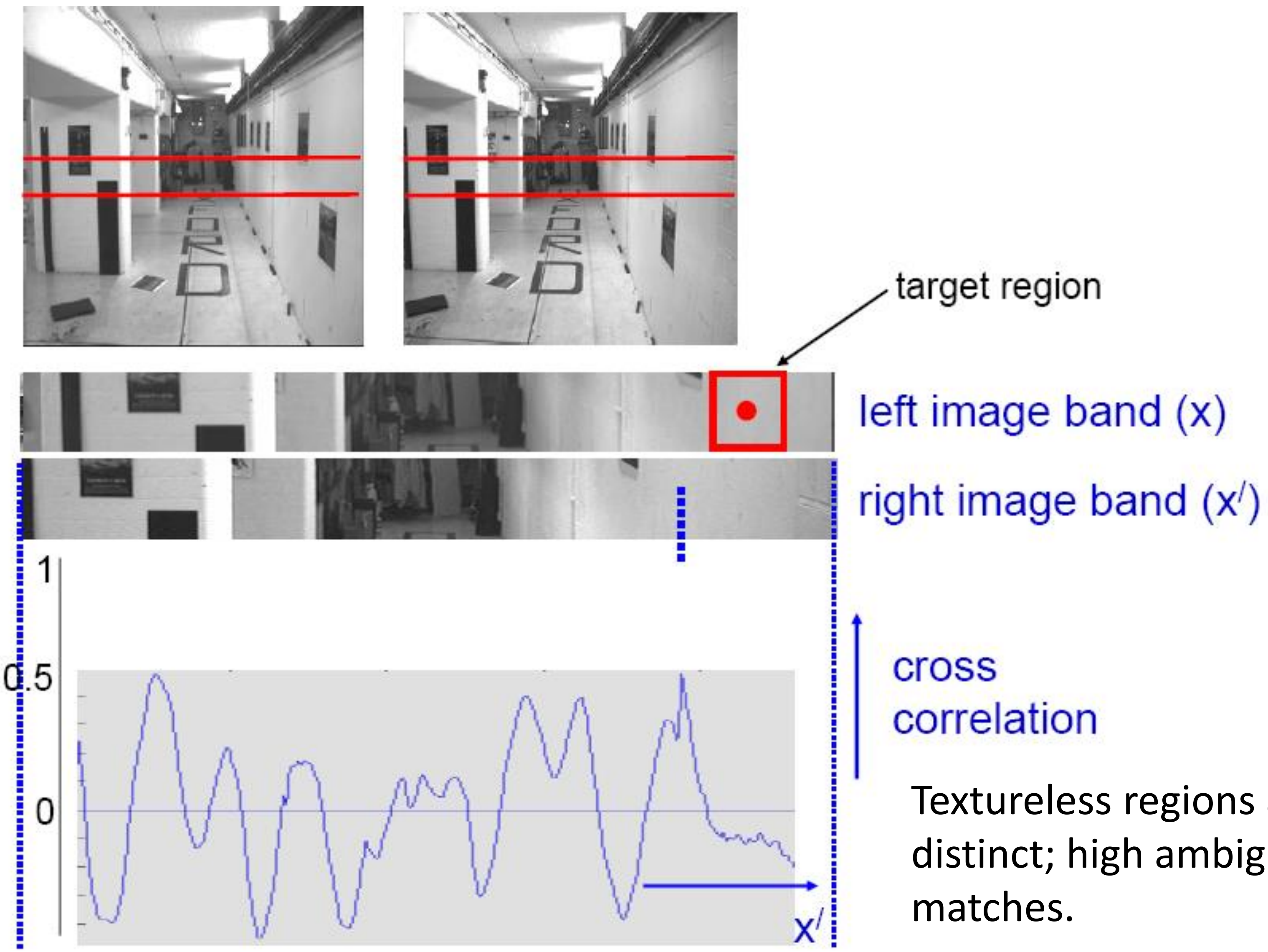
- To average noise effects, use a window around the point of interest
- Neighborhoods of corresponding points are  similar in intensity patterns
- **Similarity measures**:
  - **(Z)NCC**
  - **(Z)SSD**
  - **(Z)SAD**
  - **Census Transform** (Census descriptor plus Hamming distance)



# Correlation-based window matching

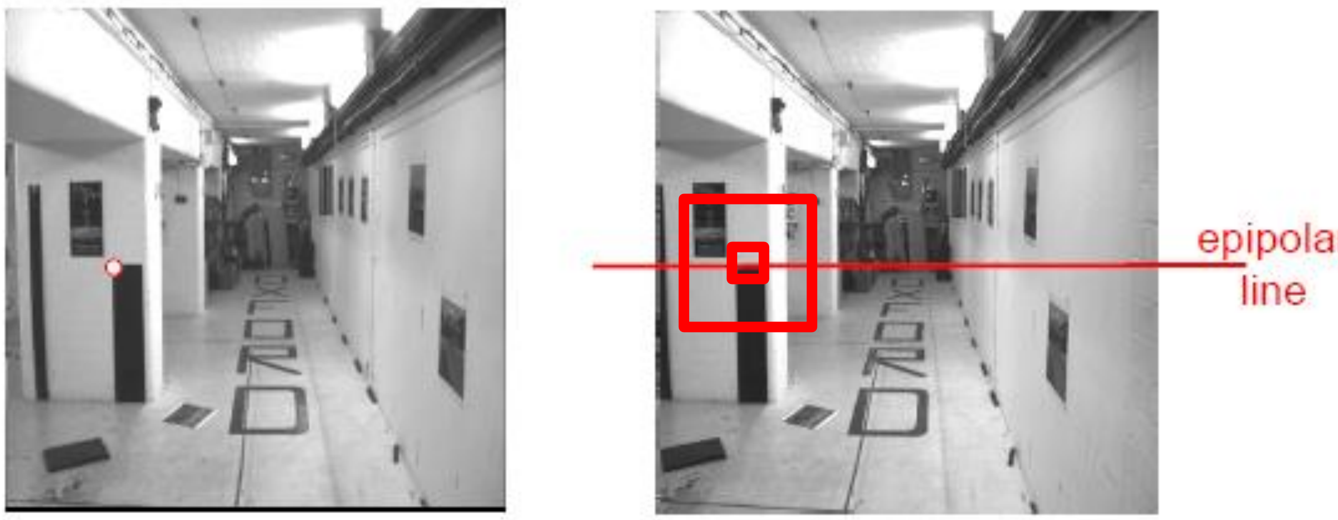


# Correspondence Problems:
# Textureless regions (**the aperture problem**)



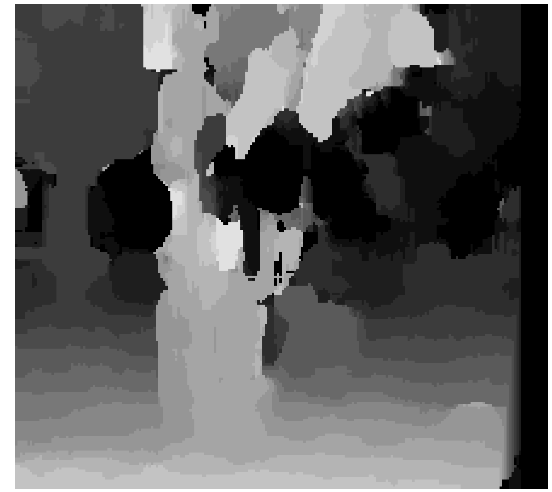

Textureless regions are non-distinct; high ambiguity for matches.

# Solution: increase window size



# Effects of window size

W = 3



W = 20

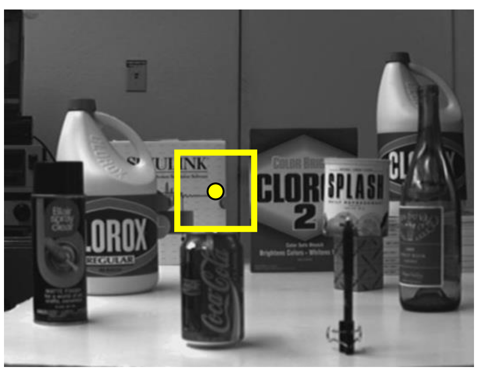
- Smaller window
  - \+ More detail
  - – More noise

- Larger window
  - \+ Smoother disparity maps
  - – Less detail

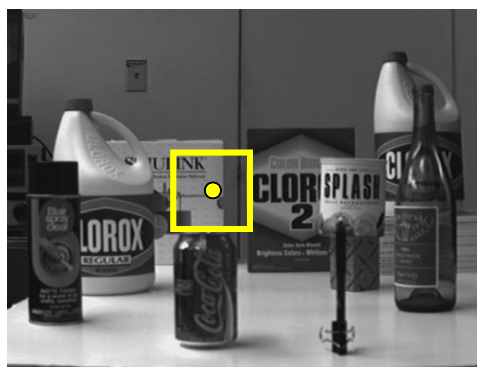
# Disparity map

Input to dense 3D reconstruction

1. For each pixel in the left image, find its corresponding point in the right image
2. Compute the disparity for each pair of correspondences
3. Visualised in gray-scale or color coded image




Left image

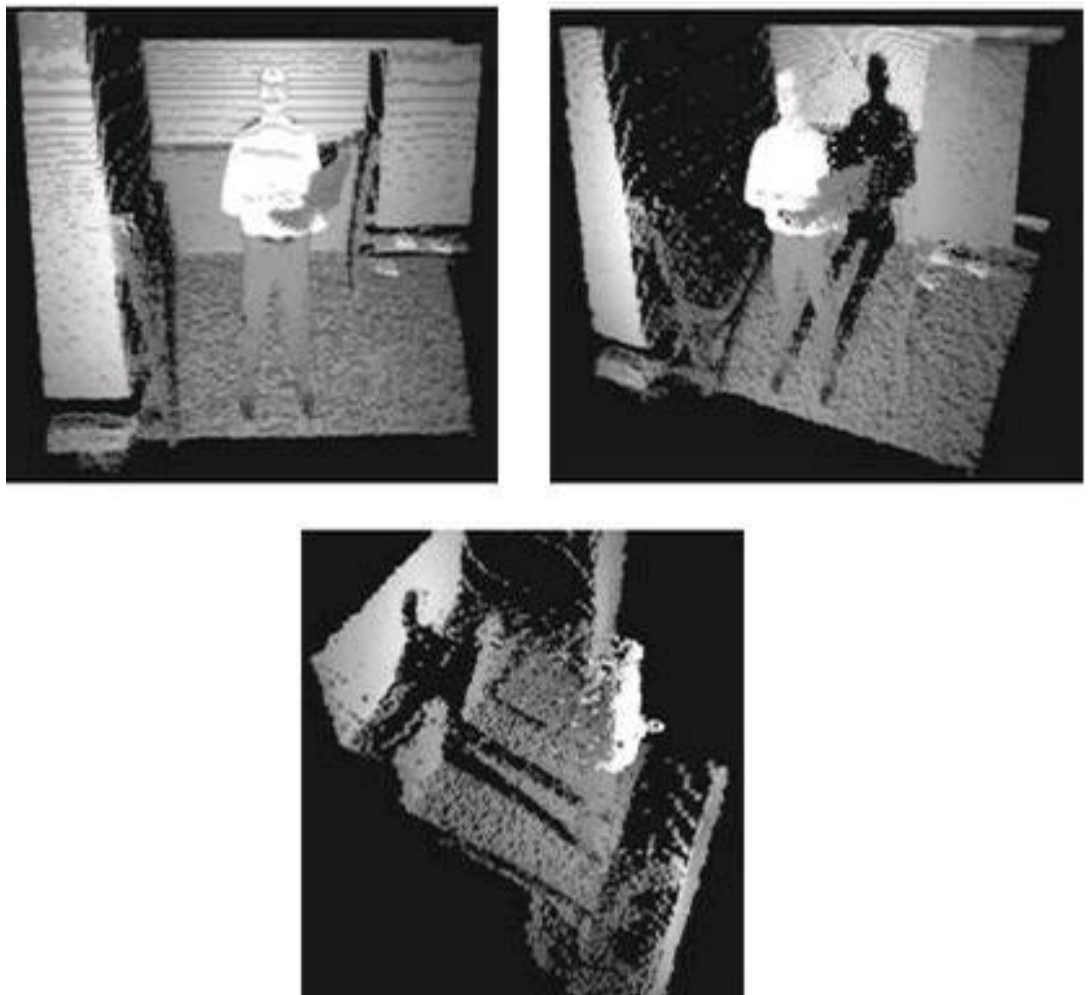



Right image


Close objects experience bigger disparity
⇨ appear brighter in disparity map

# From Disparity Maps to Point Cloud

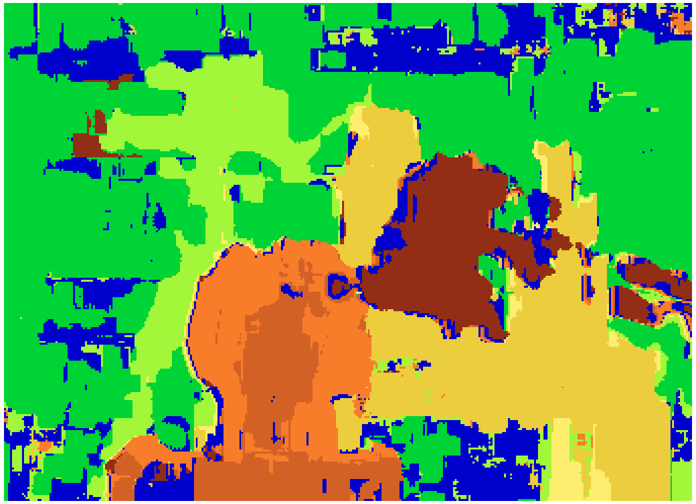
The depth Z can be computed from the disparity by recalling that $Z_P = \frac{bf}{u_l - u_r}$



# Accuracy

Data

Window-based matching



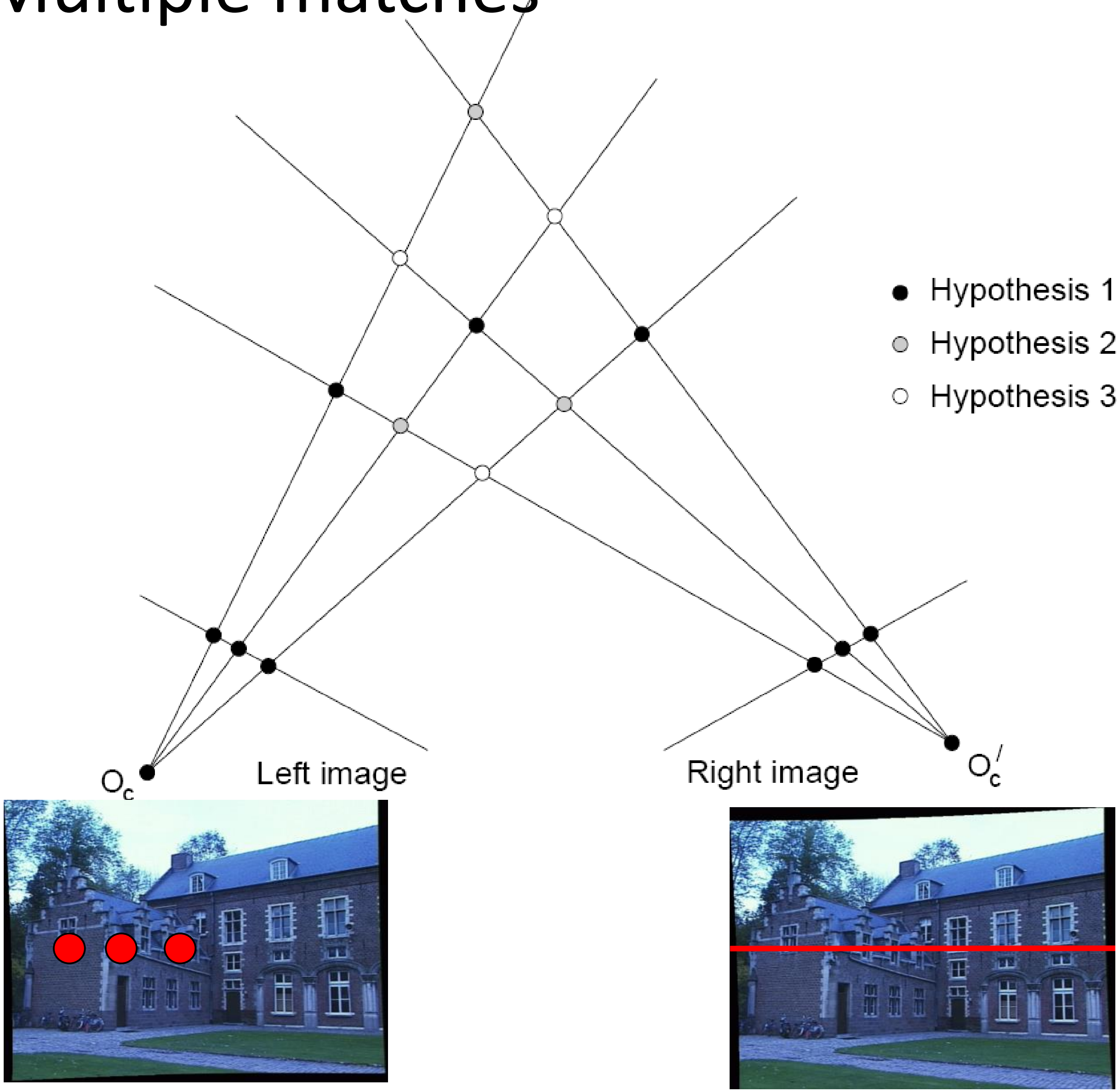
Ground truth

# Challenges

Occlusions, repetition

Non-Lambertian surfaces (e.g., specularities), textureless surfaces

# Correspondence Problems: Multiple matches




Multiple match hypotheses satisfy epipolar constraint, but which one is correct?

# How can we improve window-based matching?

- Beyond the epipolar constraint, there are “soft” constraints to help identify corresponding points
  - Uniqueness
    - Only one match in right image for every point in left image
  - Ordering
    - Points on **same surface** will be in same order in both views
  - Disparity gradient
    - Disparity changes smoothly between points on the same surface

# Better methods exist...

Graph cuts

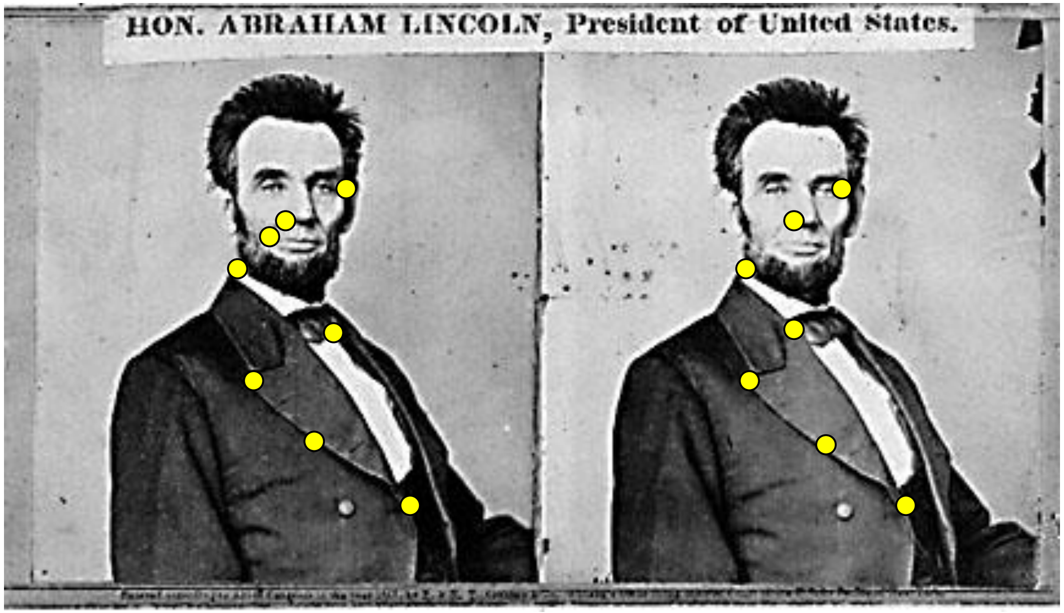
Ground truth

Y. Boykov, O. Veksler, and R. Zabih, [Fast Approximate Energy Minimization via Graph Cuts](), PAMI 2001

For the latest and greatest: [http://www.middlebury.edu/stereo/]()

# Sparse Stereo Correspondence

- Restrict search to sparse set of detected features
- Feature matching
- Use epipolar geometry to narrow the search further



# Things to Remember

- Disparity
- Triangulation: simplified and general case, linear and non linear approach
- Choosing the baseline
- Correspondence problem: epipoles, epipolar lines, epipolar plane
- Stereo rectification

- Readings:
  - Szeliski book: Chapter 11
  - Peter Corke book: Chapter 14.3
  - Autonomous Mobile Robot book: Chapter 4.2.5