ROBOTICS & PERCEPTION GROUP

Universität Zürich[UZH]

Lecture 07
Epipolar Geometry and Stereo

Prof. Dr. Davide Scaramuzza

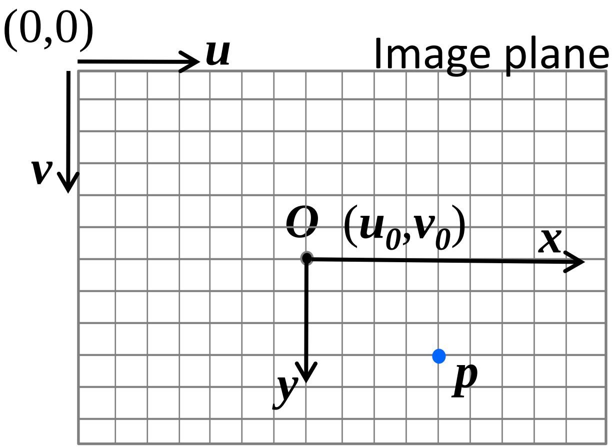
sdavide@ifi.uzh.ch

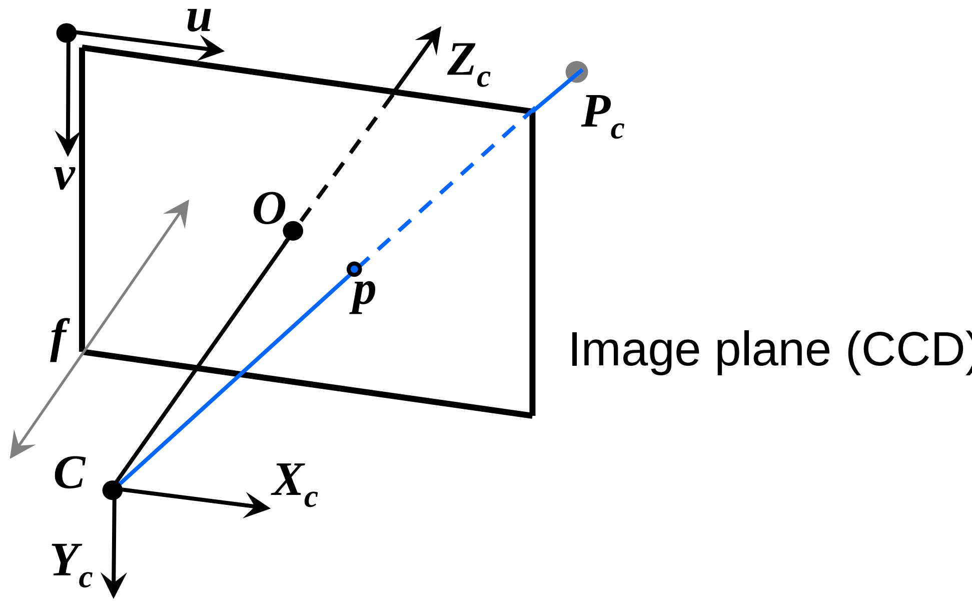
Course Topics

- Principles of image formation
- Image filtering
- Feature detection
- Multi-view geometry
- 3D Reconstruction
- Visual recognition

Course Topics (done)

- Principles of image formation
  - Perspective projection
  - Camera calibration





Course Topics (done)

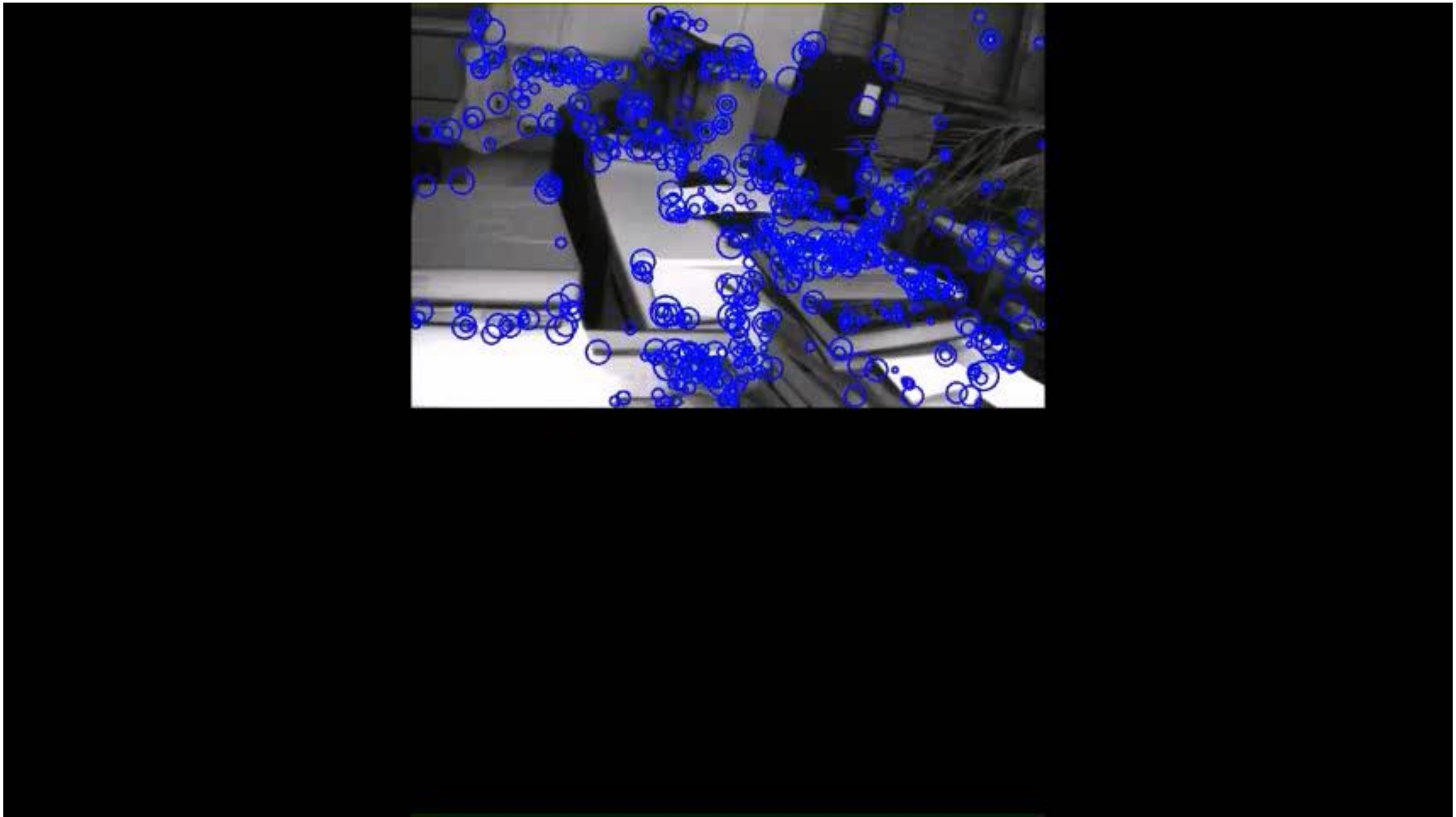
- Image filtering

Low-pass filtered image

High-pass filtered image

Course Topics (done)

- Feature detection



Course Topics (to do)

- Multi-view geometry and 3D reconstruction

Course Topics (to do)

- Multi-view geometry and 3D reconstruction

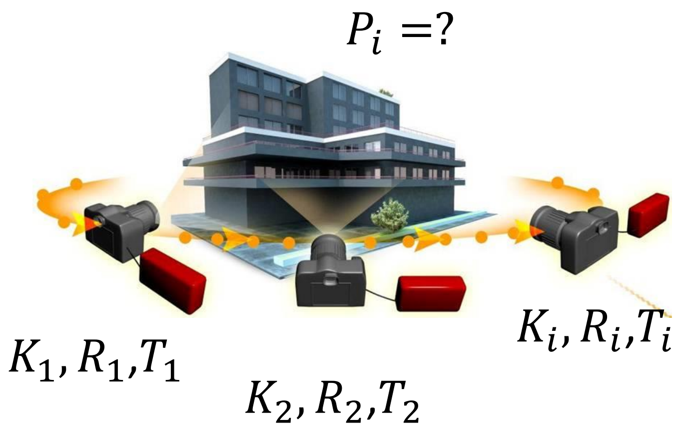
San Marco square, Venice
14,079 images, 4,515,157 points

Today’s outline

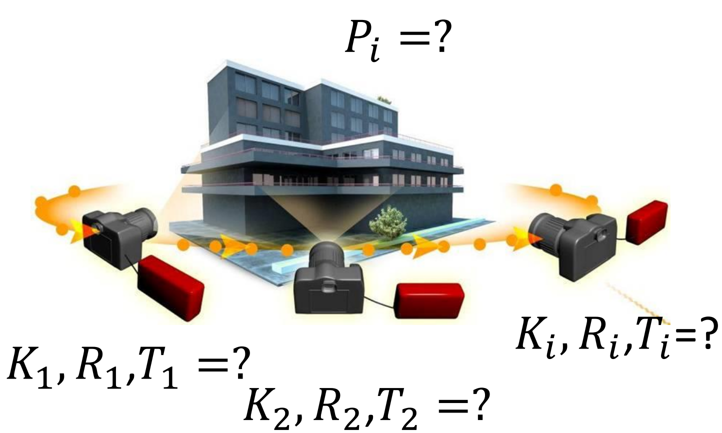
- Stereo Vision
- Epipolar Geometry

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

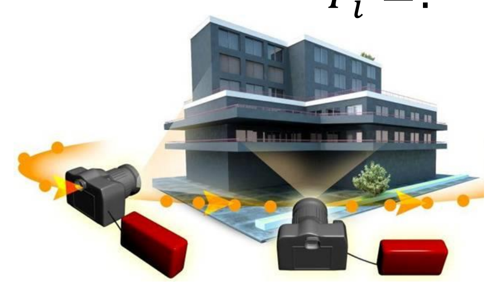
P_i =?

K_1, R_1, T_1

K_2, R_2, T_2

- **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

P_i =?

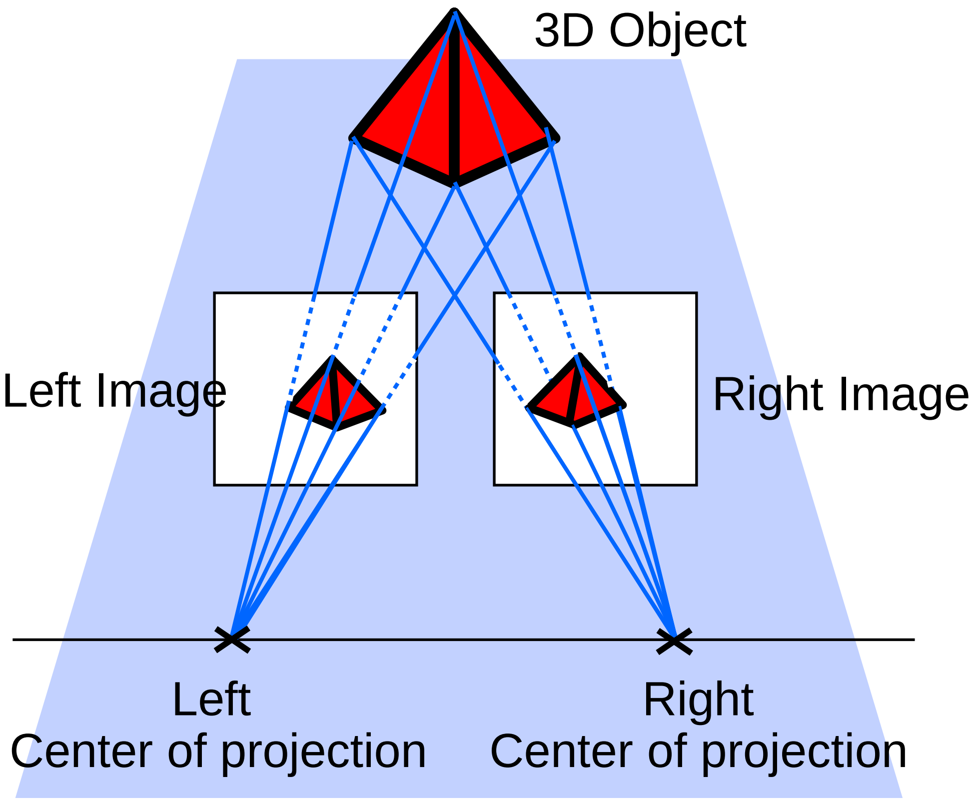


K_1, R_1, T_1 =?

K_2, R_2, T_2 =?

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**»

Image from the left eye

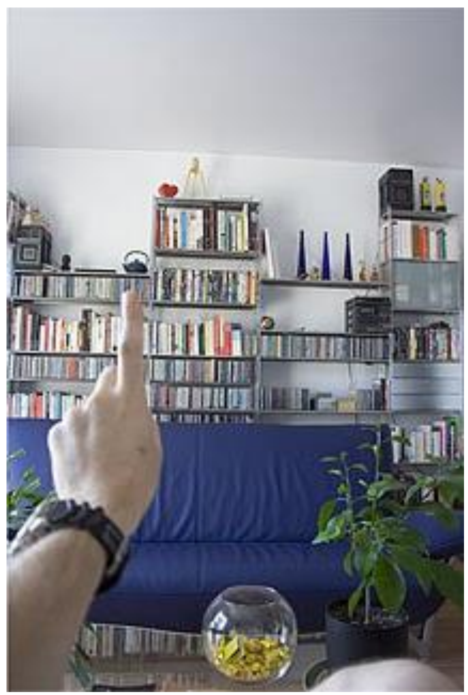
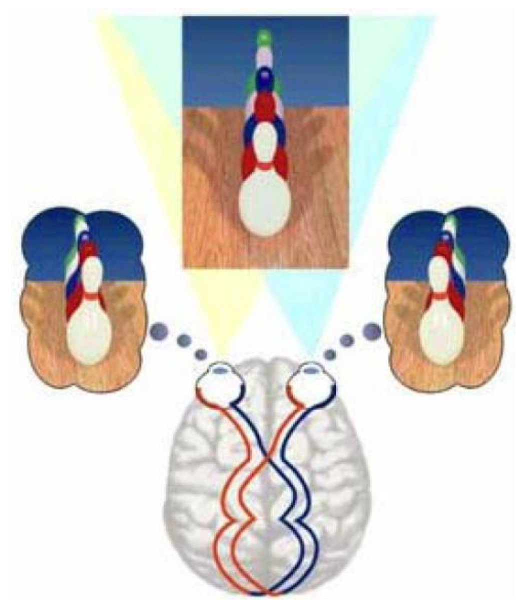
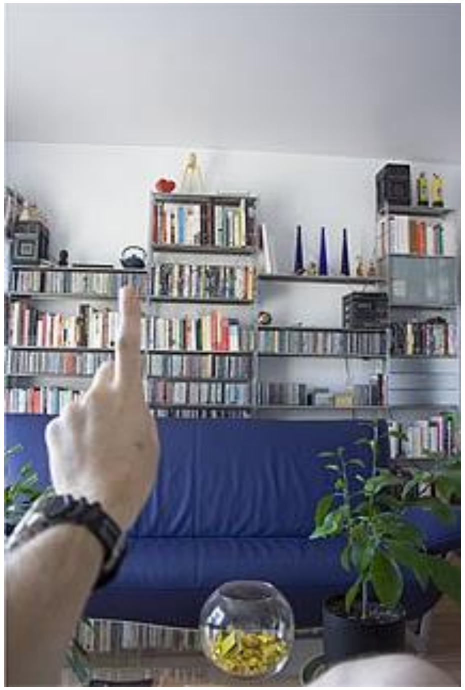


Image from the right eye



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 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

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

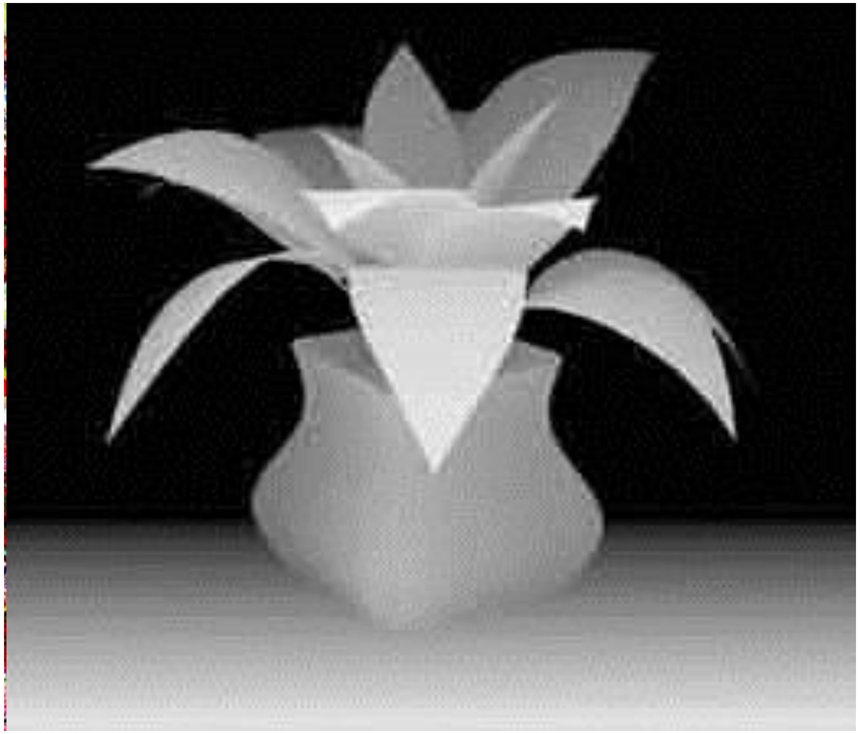
These animated GIF images display intermittently the left and right image

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

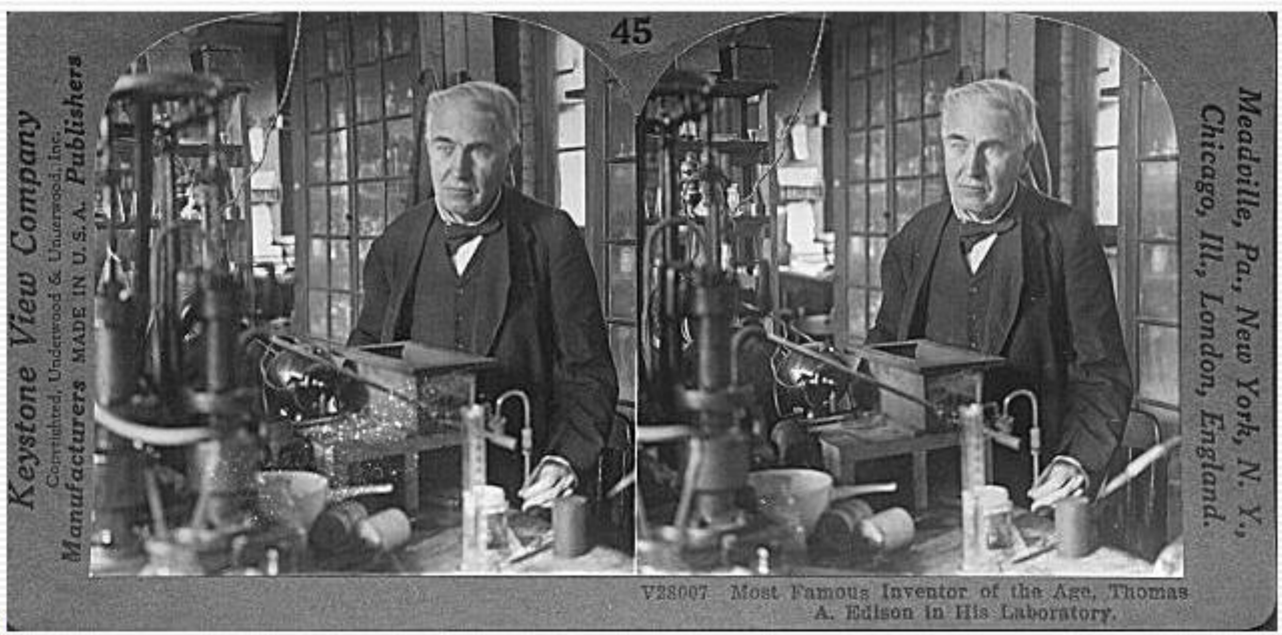
Applications: Stereo photography and stereo viewers

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

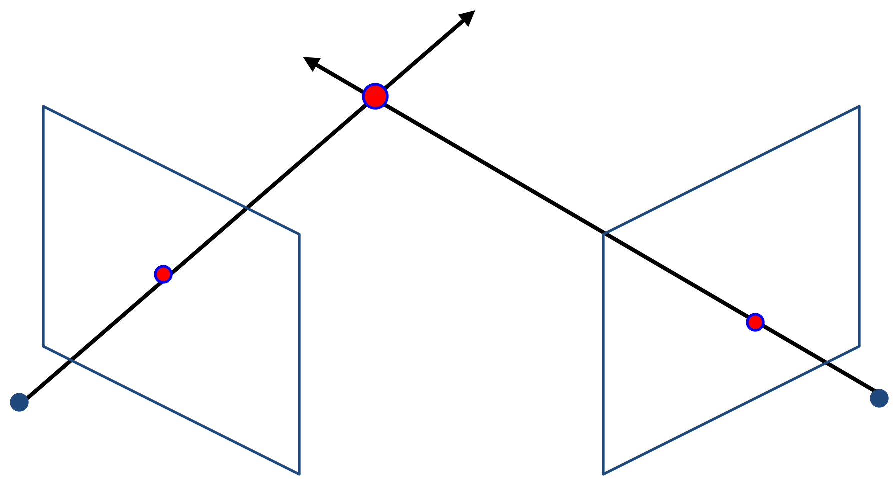
The first method to produce anaglyph images was developed in 1852 by Wilhelm Rollmann in Leipzig, Germany



Stereo Vision

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

Stereo Vision: basic idea



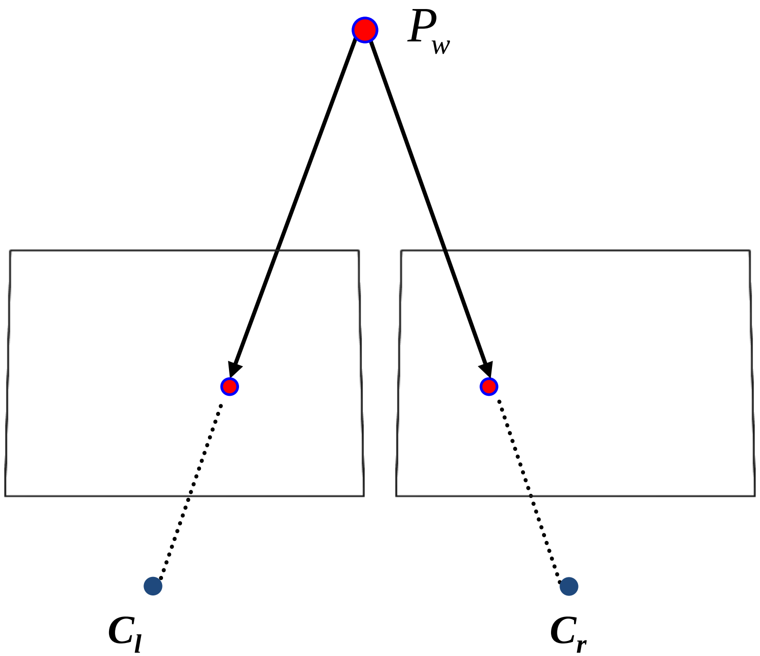
Basic Principle: Triangulation

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

Stereo Vision: basic idea

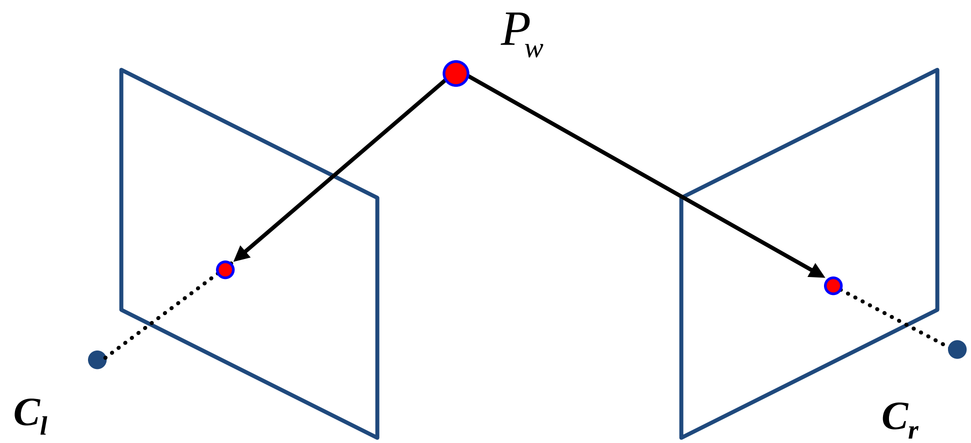
Simplified case

(identical cameras and aligned)



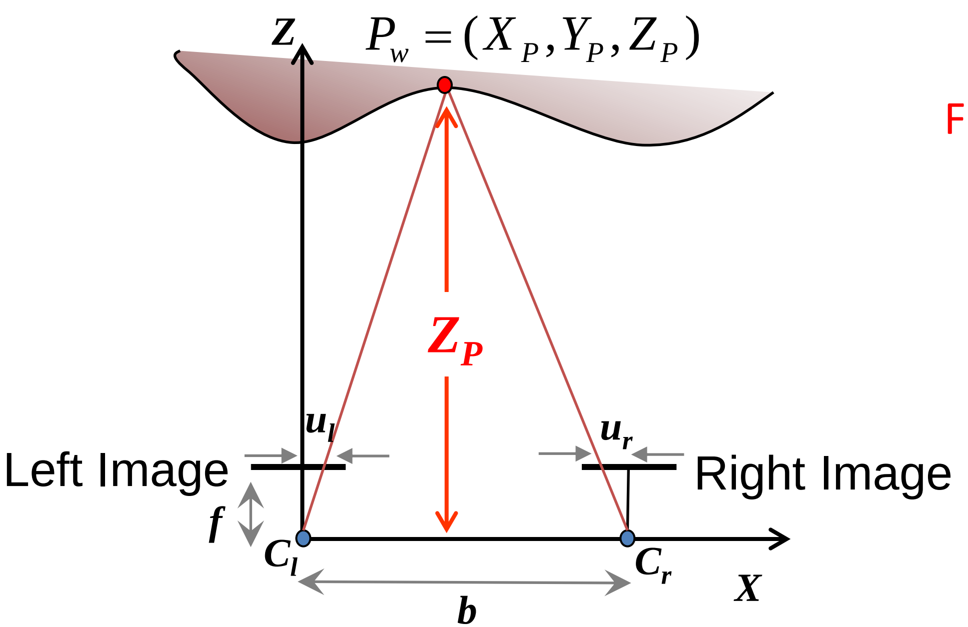
General case

(non identical cameras and not aligned)



Stereo Vision - The simplified case

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

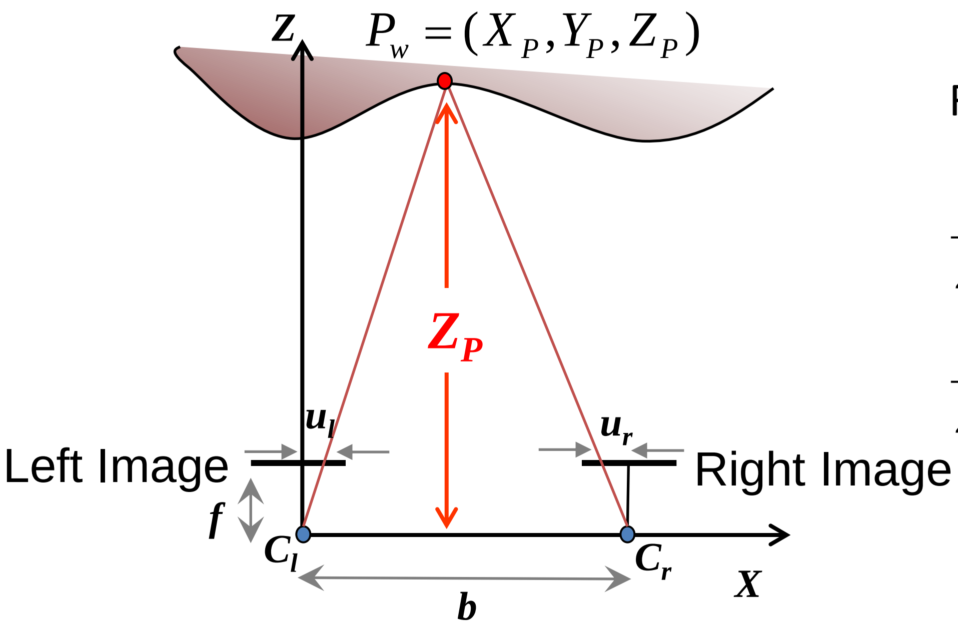


Baseline
distance between the optical centers of the two cameras

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



Baseline

distance between the optical centers of the two cameras

From Similar Triangles:

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

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

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

Disparity

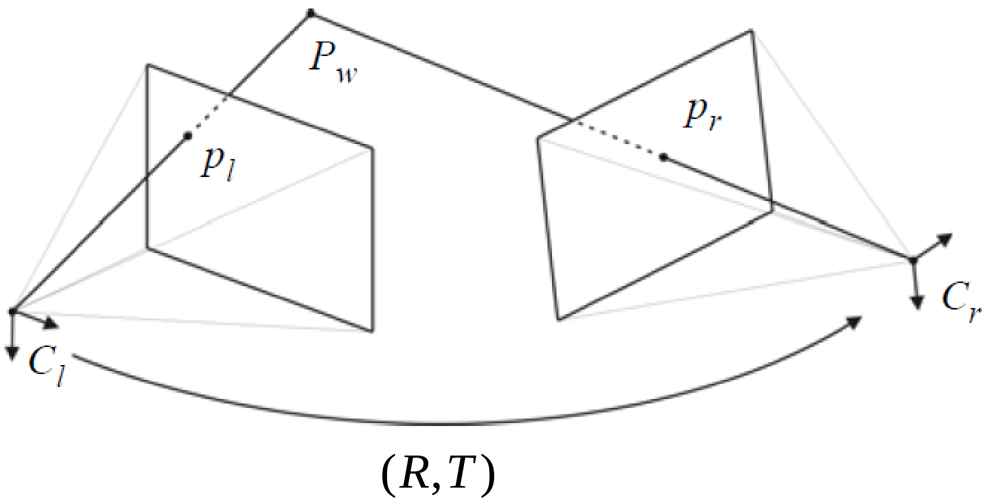
difference in image location of the projection of a 3D point in two image planes

Stereo Vision

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

Stereo Vision – the general case

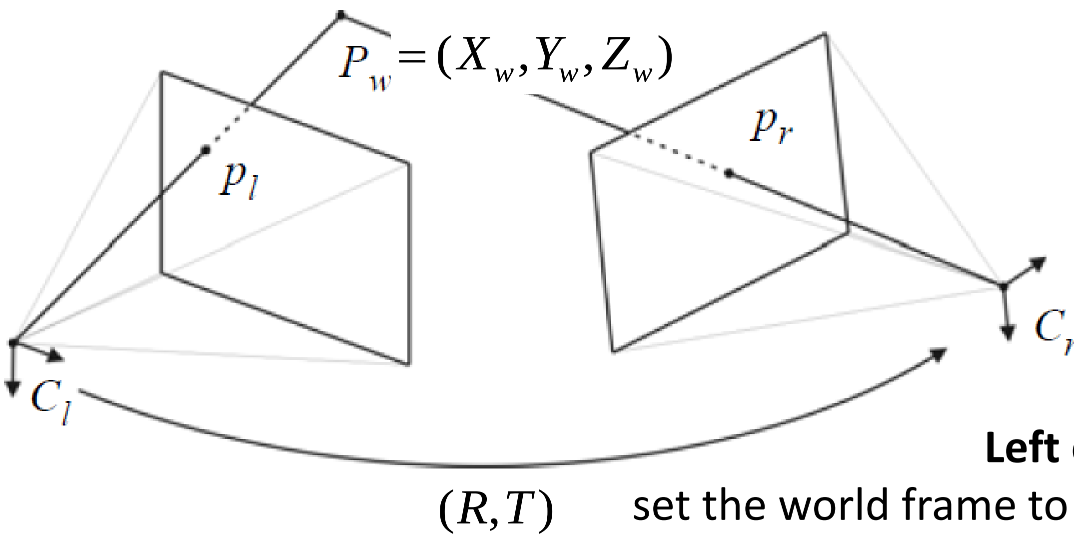
- Two identical cameras do not exist in nature!
- Aligning both cameras on a horizontal axis is very hard -> Impossible, why?



- 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.



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.

Stereo Vision

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

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

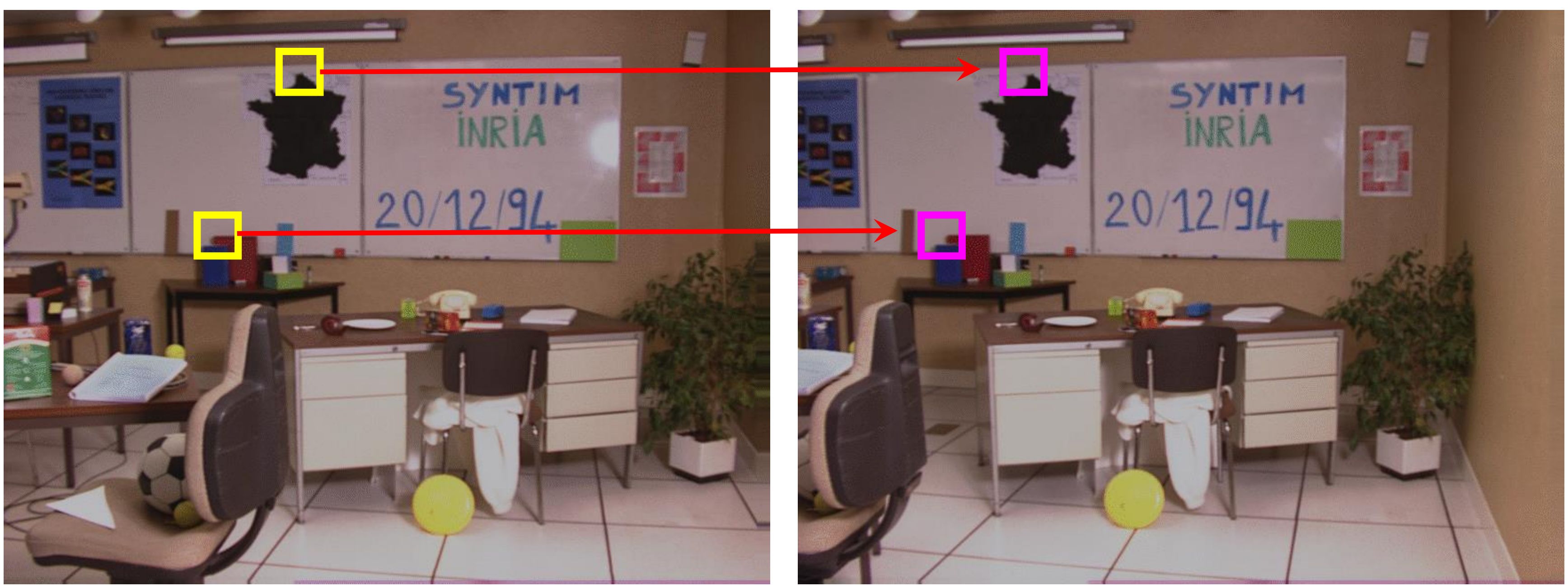
Given the point p in left image, where is its corresponding point p' in the right image?

Left image

Right image

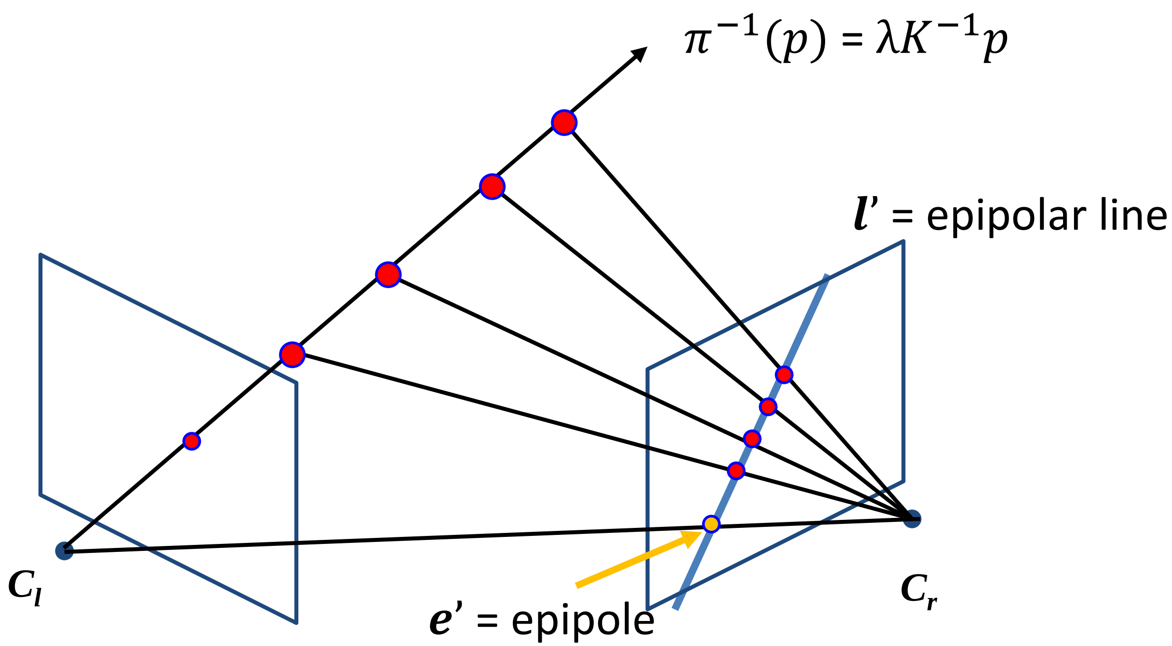
Correspondence Problem

- **Correspondence search**: identify image patches in the left & right images, corresponding to the same scene structure.
- **Similarity measures**:
  - Zero-Normalized Cross-Correlation (**ZNCC**)
  - Sum of Squared Differences (**SSD**)
  - Sum of Absolute Differences (**SAD**)
  - **Census Transform** (what is this?)



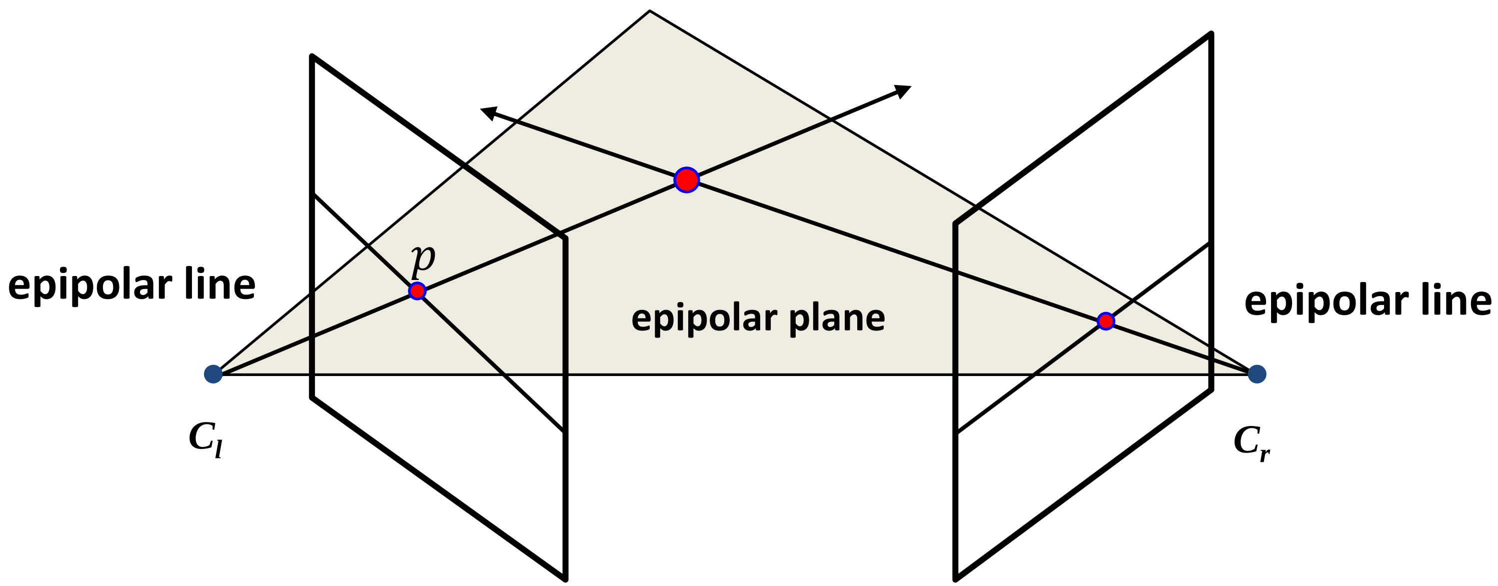
Correspondence Problem

- **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** $\boldsymbol{e}'$ is the projection of the optical center in in the other camera image



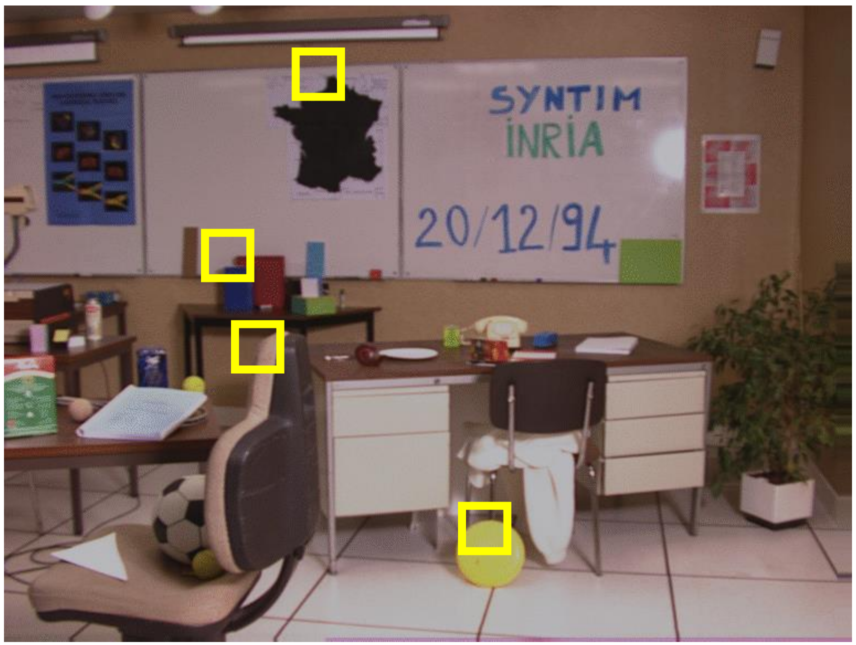
The Epipolar Constraint

- 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*

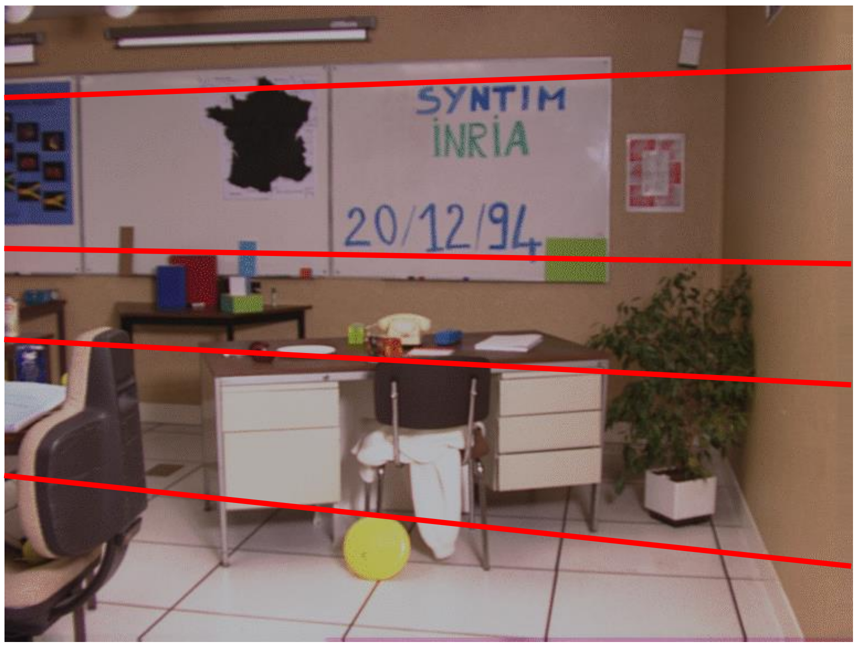


Correspondence Problem: Epipolar Constraint

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



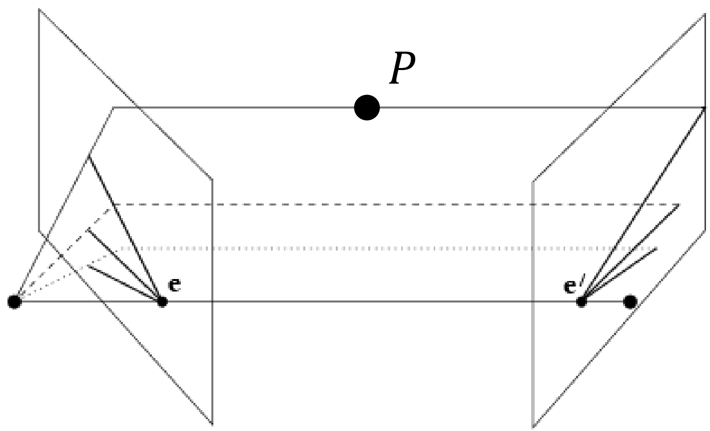
Left image



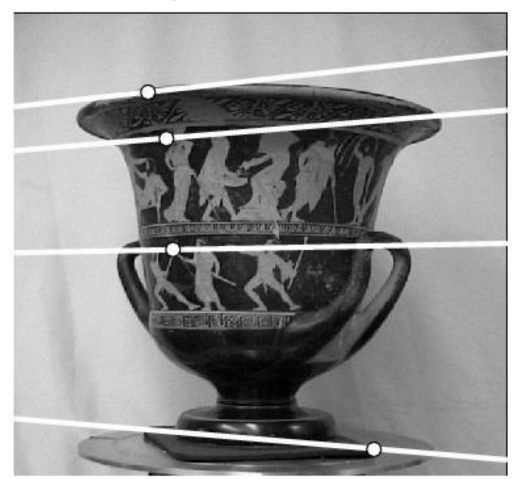
Right image

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

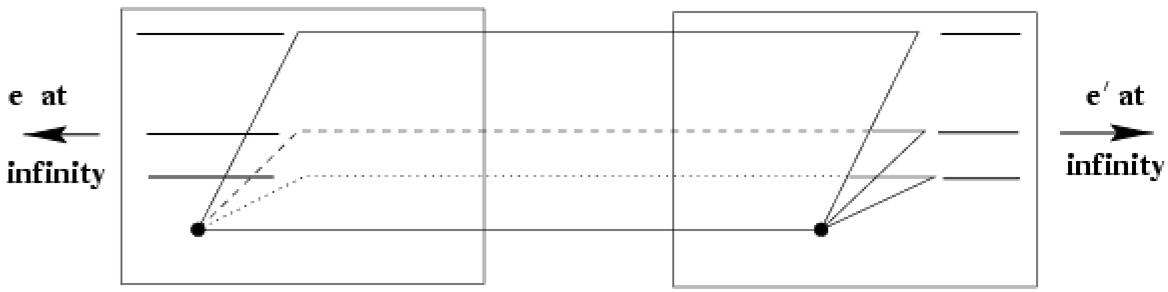


Left image



Right image

Example: identical and horizontally-aligned cameras

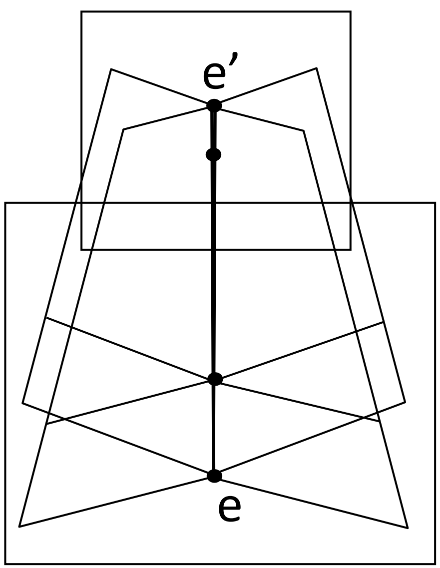


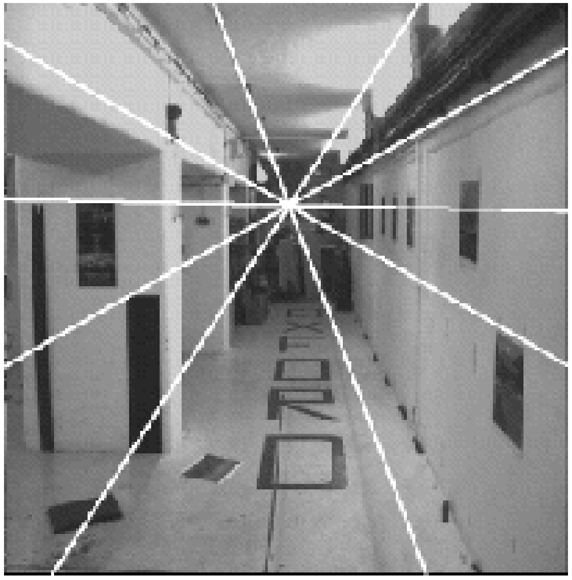
Left image

Right image

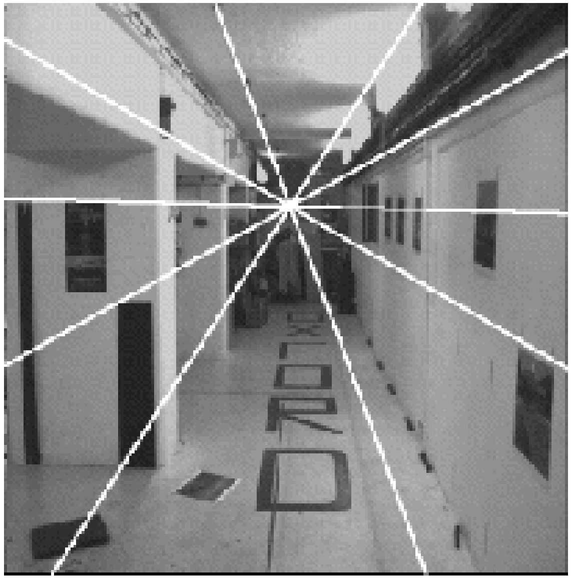
Example: forward motion (parallel to the optical axis)

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





Left image



Right image

Stereo Vision

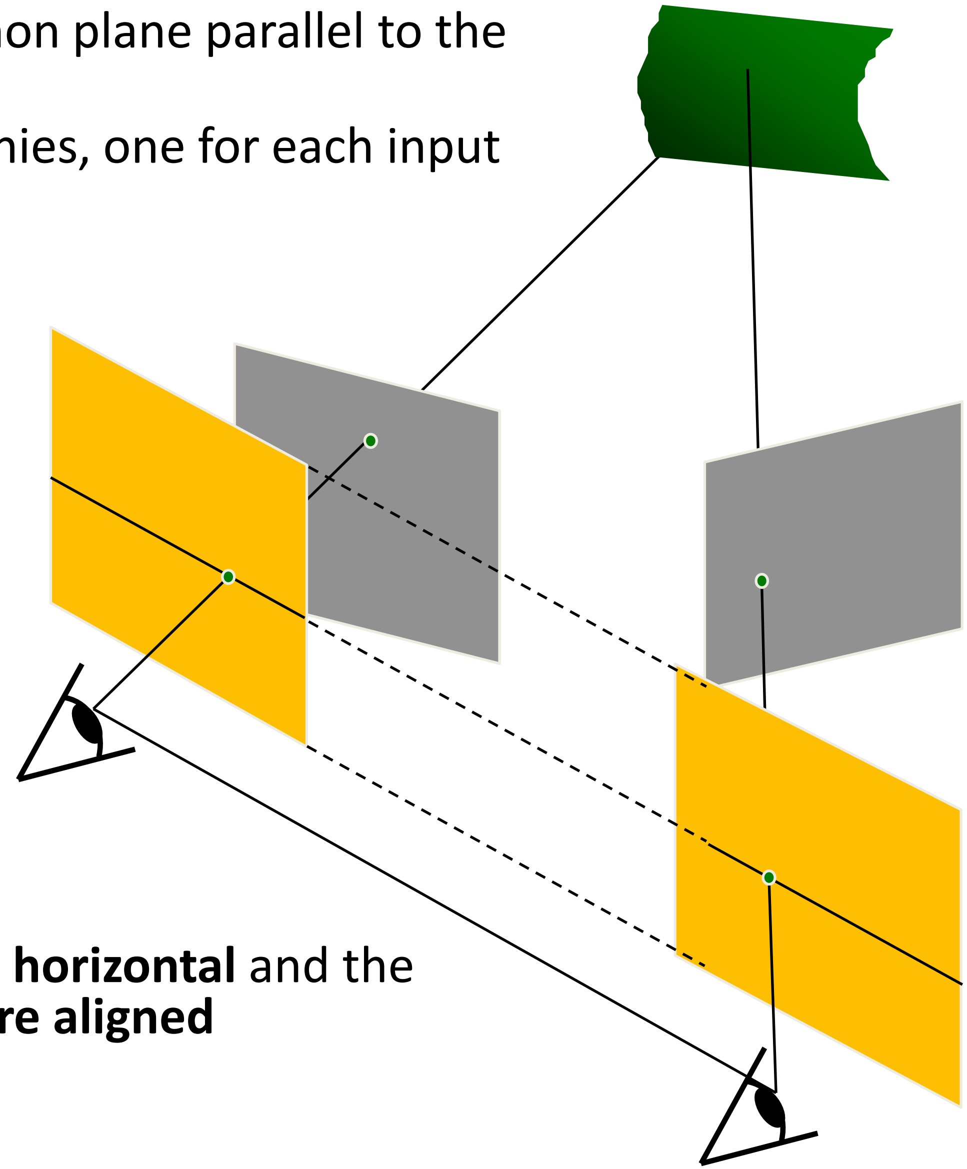
- Simplified case
- General case
- Correspondence problem
- Stereo rectification
- Triangulation

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: example

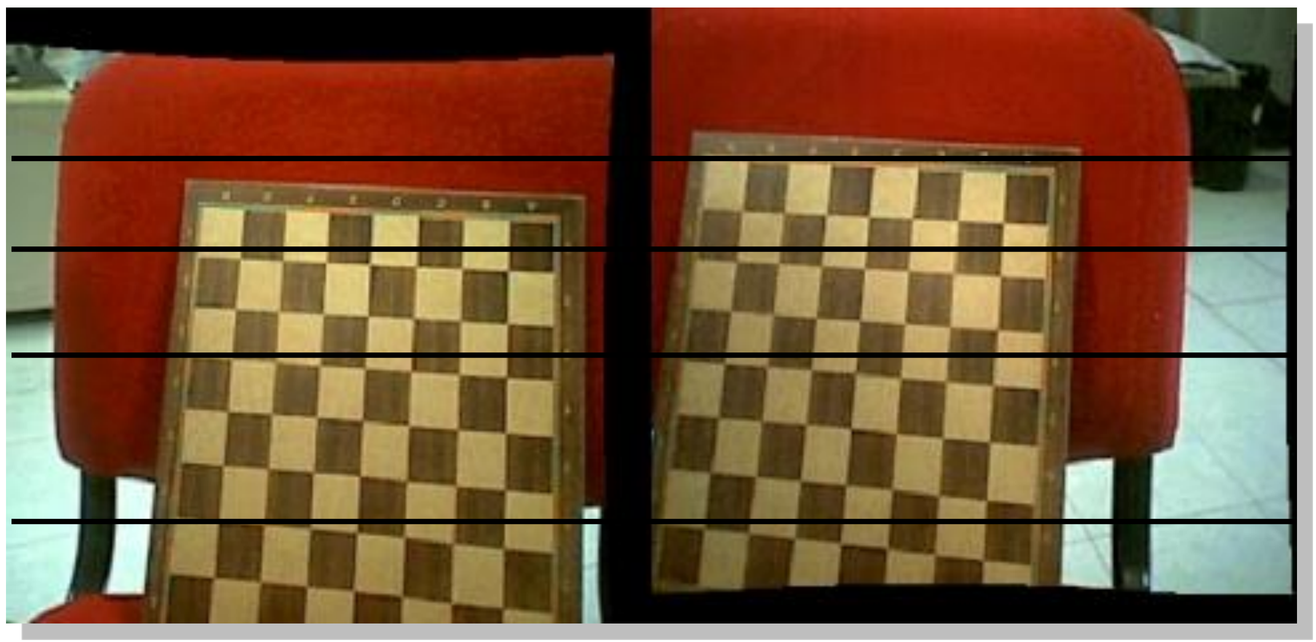
Left Right

scanlines

Stereo Rectification: example

- First, remove radial distortion

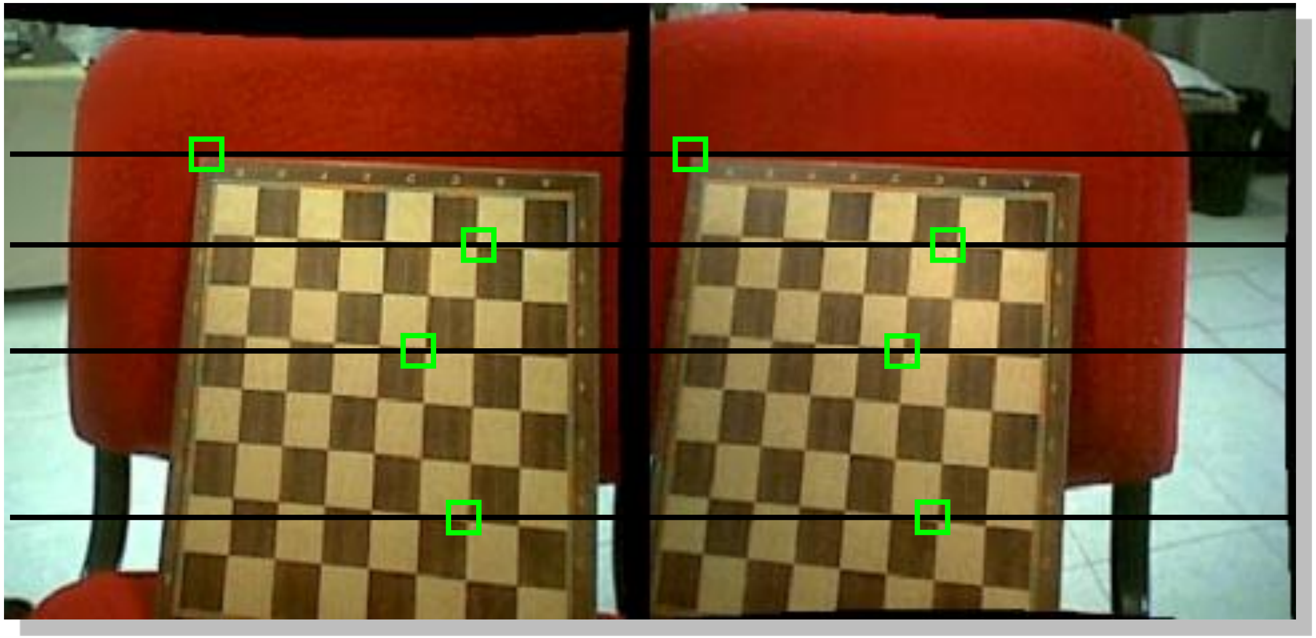
Left	Right



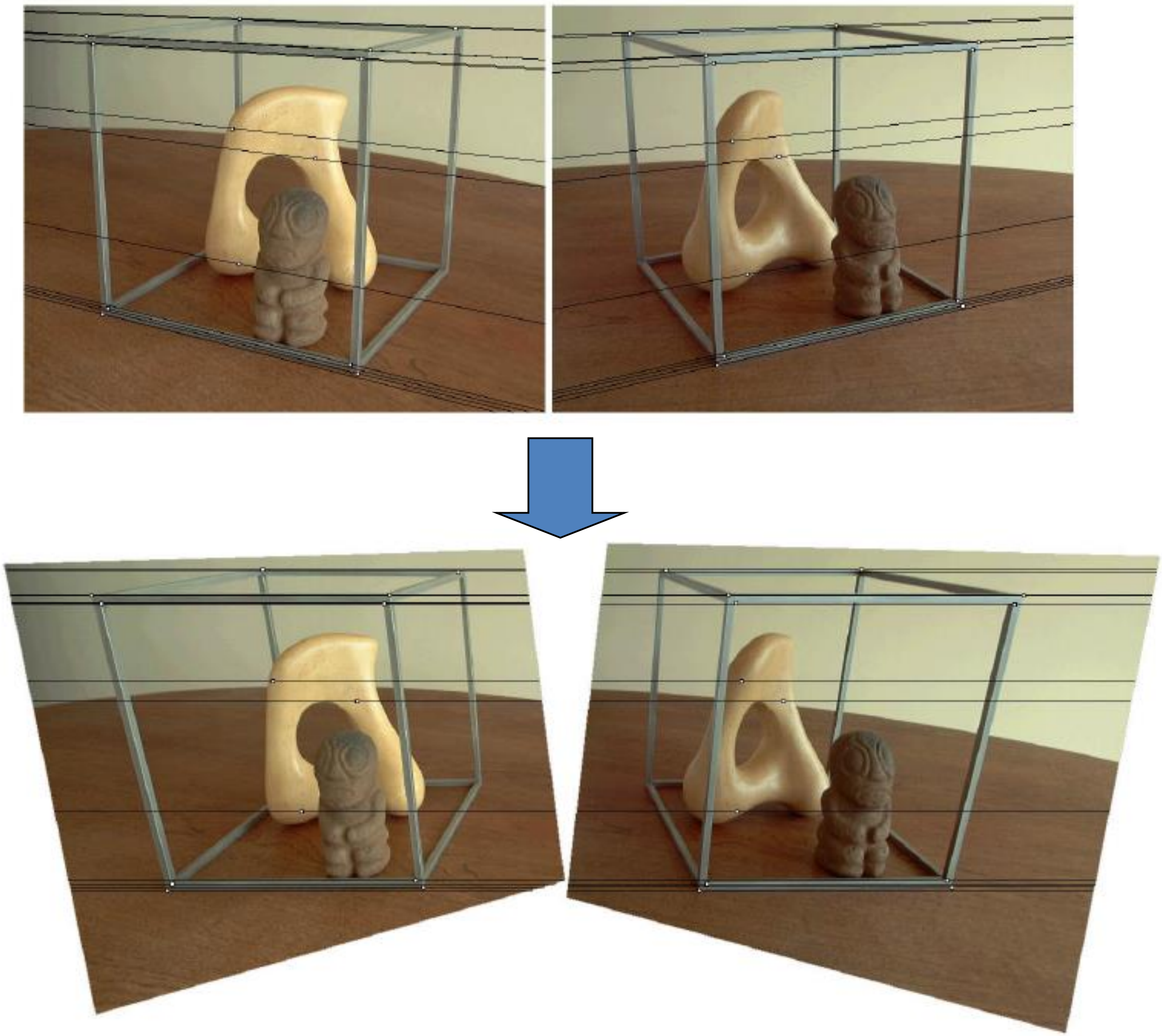
Stereo Rectification: example

- First, remove radial distortion
- Then, compute homographies and rectify

Left Right



Stereo Rectification: example

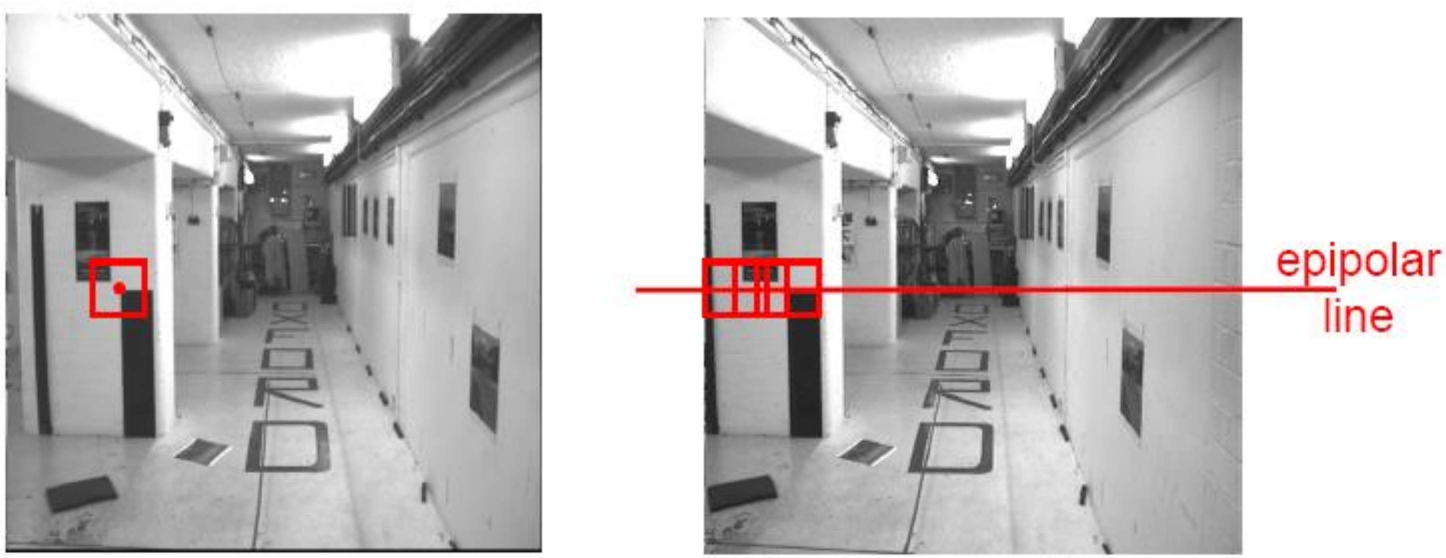


Stereo Vision

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

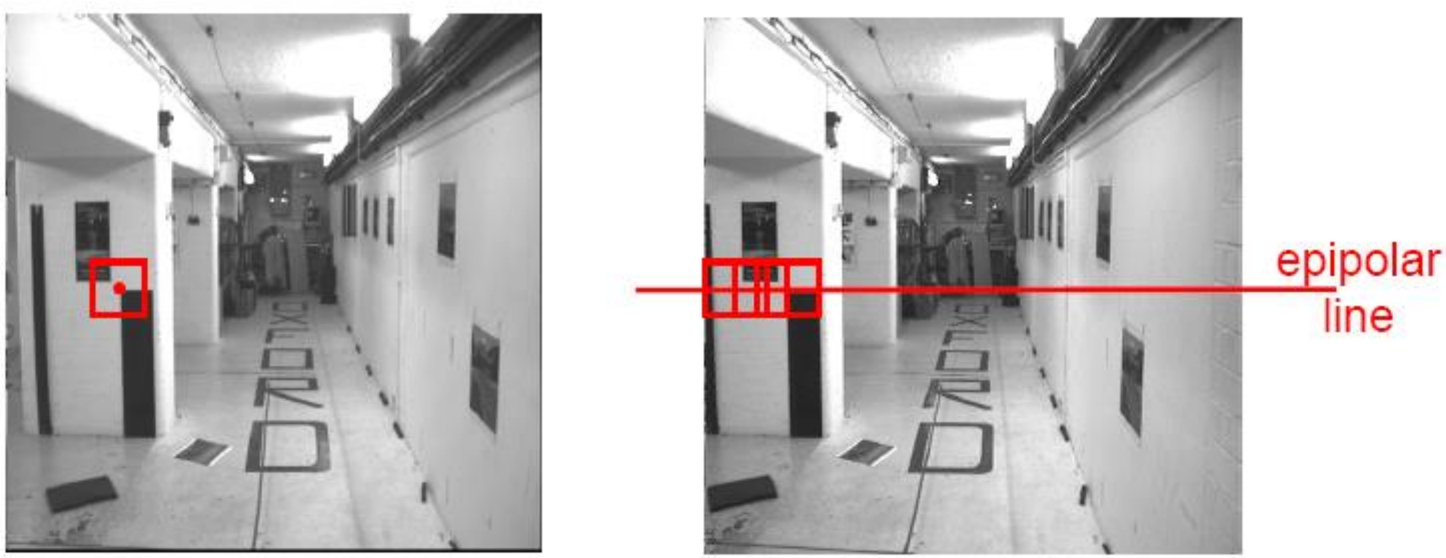
Correspondence problem

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

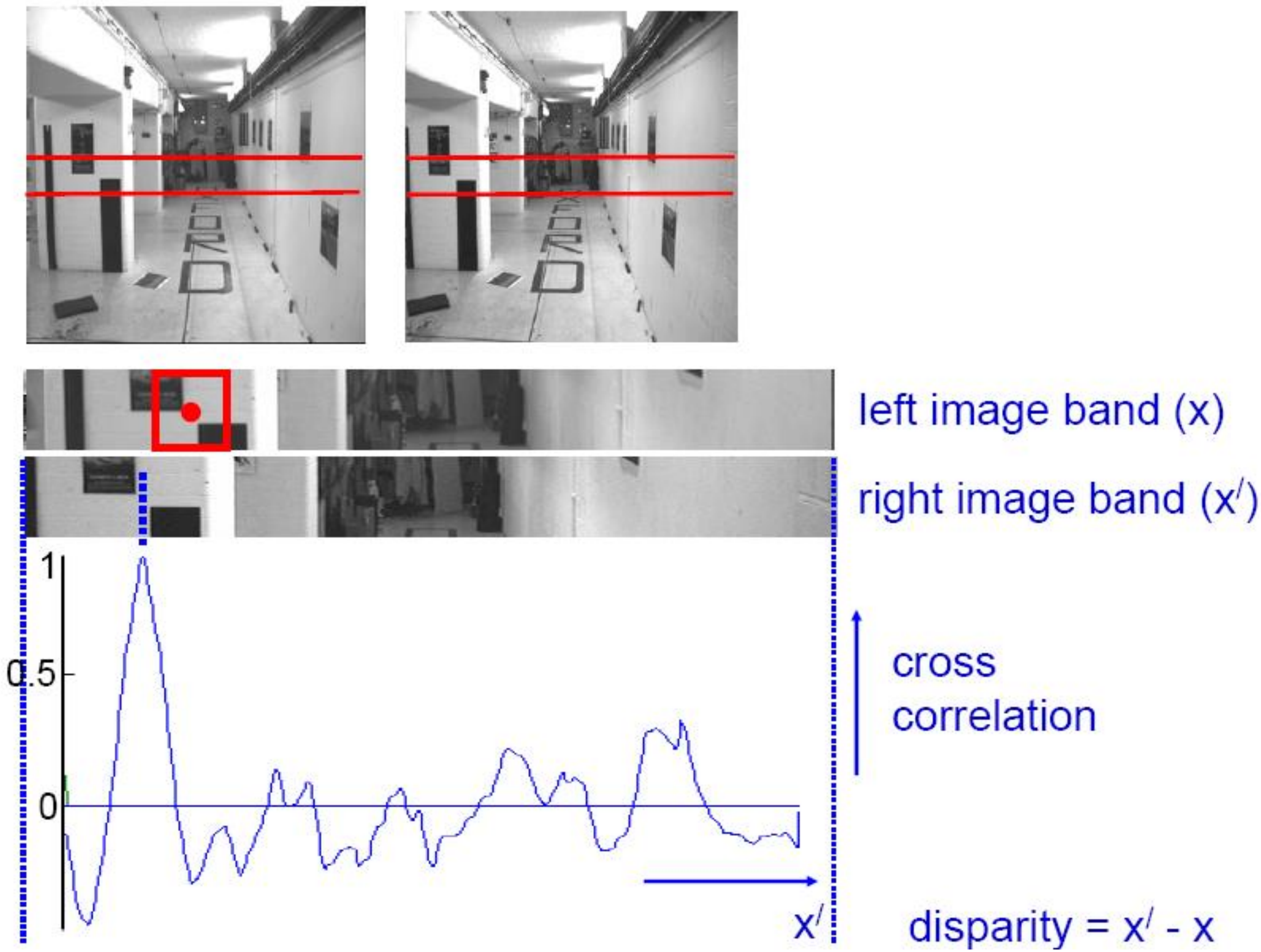


Correspondence problem

- To average noise effects, use a window around the point of interest
- Neighborhoods of corresponding points are similar in intensity patterns
- **Similarity measures**:
  - Zero-Normalized Cross-Correlation (**ZNCC**)
  - Sum of Squared Differences (**SSD**),
  - Sum of Absolute Differences (**SAD**)
  - **Census Transform** (Census descriptor plus Hamming distance)

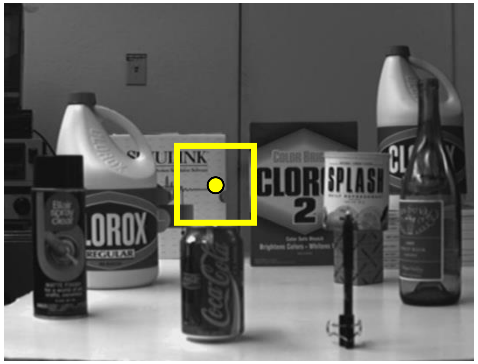


Correlation-based window matching

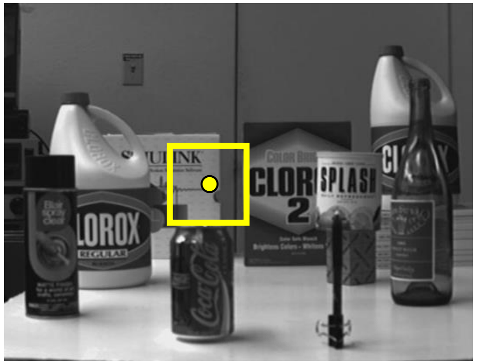


Disparity map

- Useful for dense 3D reconstruction
- Identify corresponding points of **all image pixels** in the original images
  - For each epipolar line and for each pixel (patch) in the left image
    - compare with every pixel (patch) on same epipolar line in right image
    - pick position with best similarity score (e.g., ZNCC, SSD, etc.)
- Compute the disparity for each pair of correspondences
- Usually visualised in gray-scale images



Left image



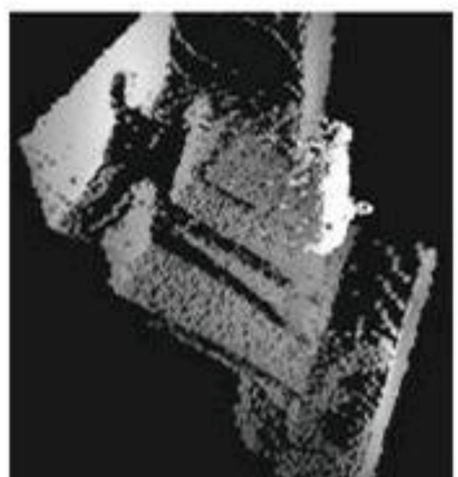
Right image

Close objects experience bigger disparity
⇨ appear brighter in disparity map

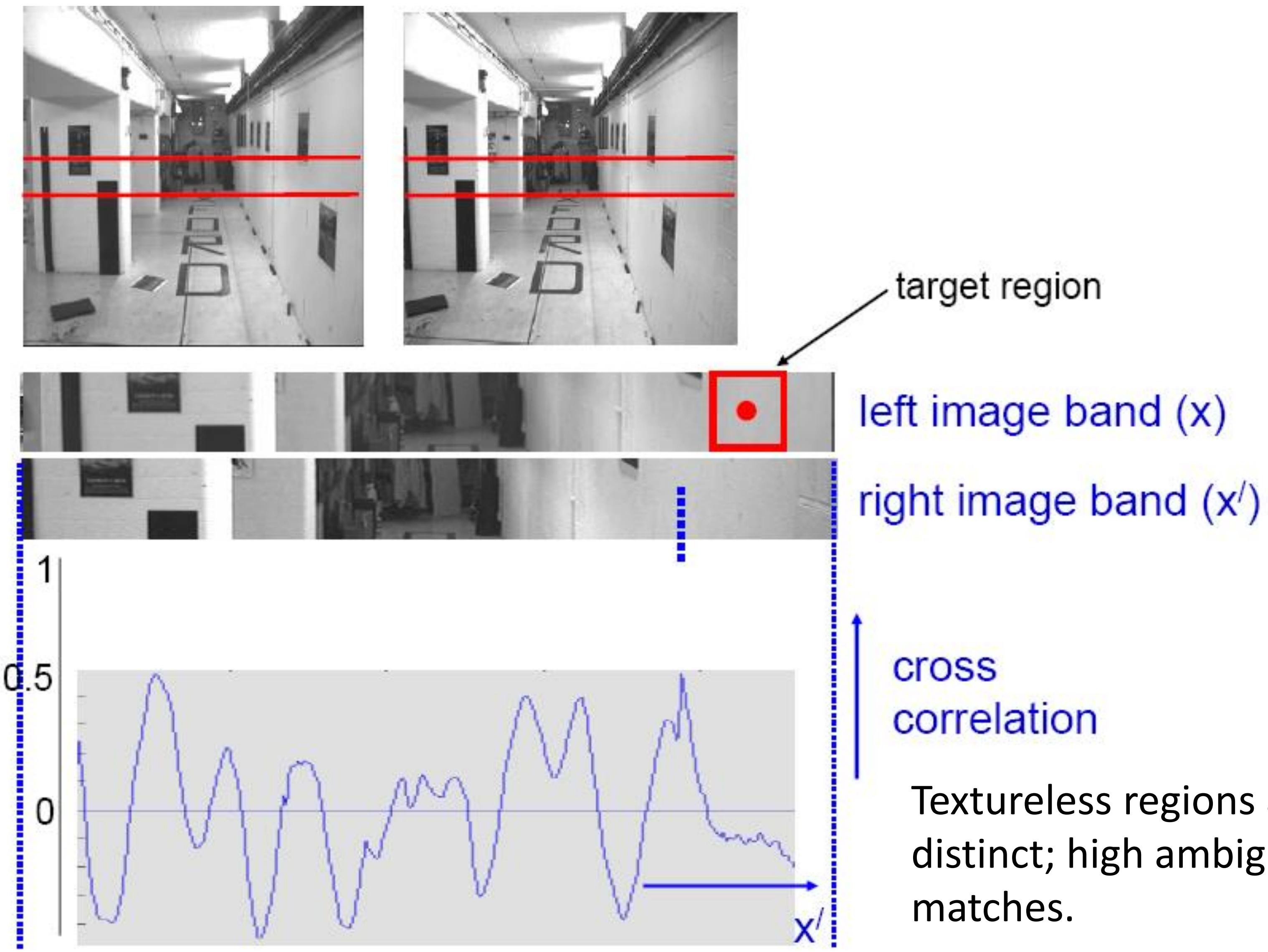
From Disparity Maps to Point Cloud

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



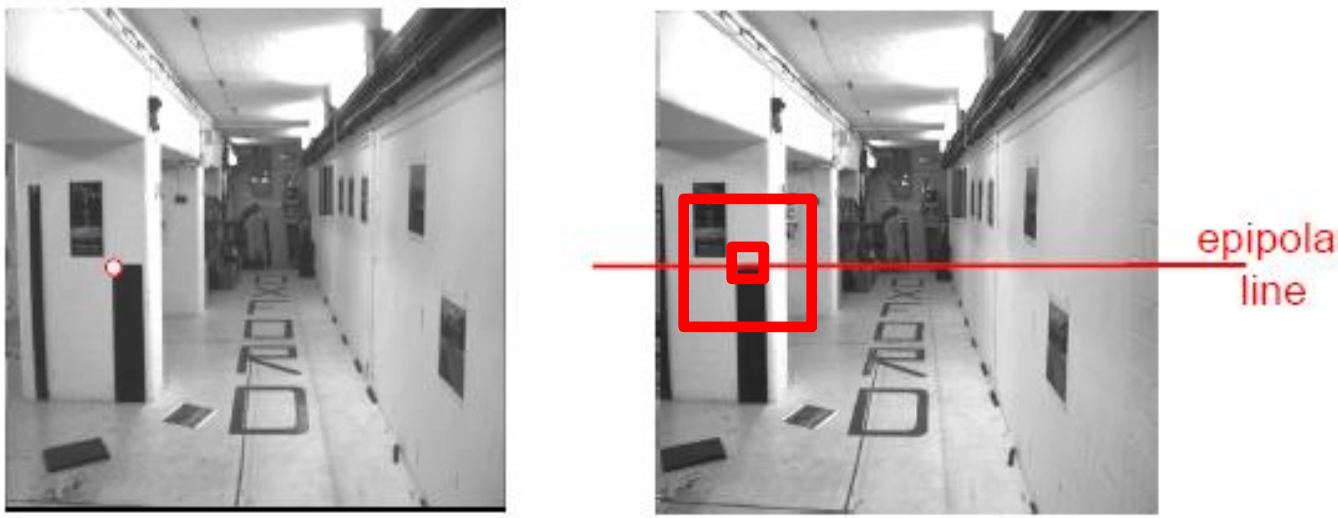


Correspondence Problems:
Textureless regions (**the aperture problem**)



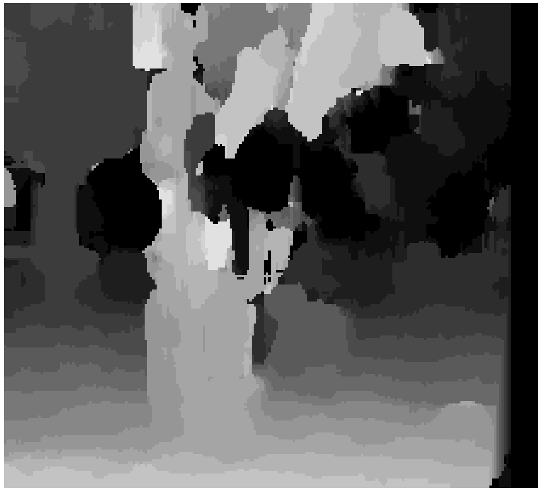
Textureless regions are non-distinct; high ambiguity for matches.

Solution: increase window size



Effects of window size

W = 3

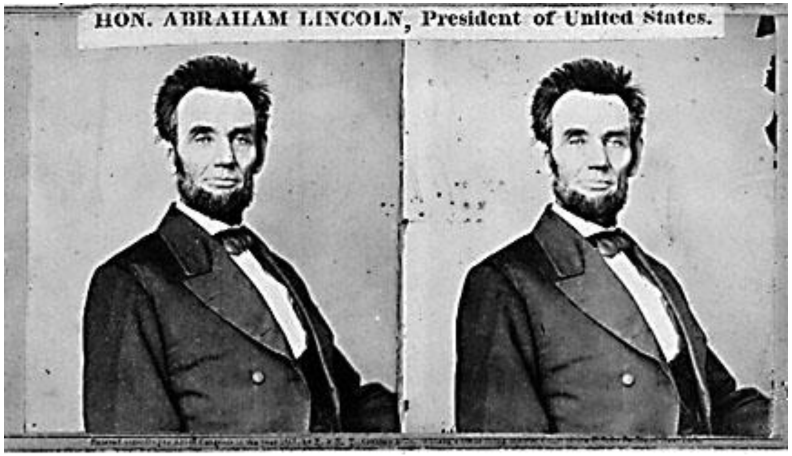


W = 20

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

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

Failures of correspondence search

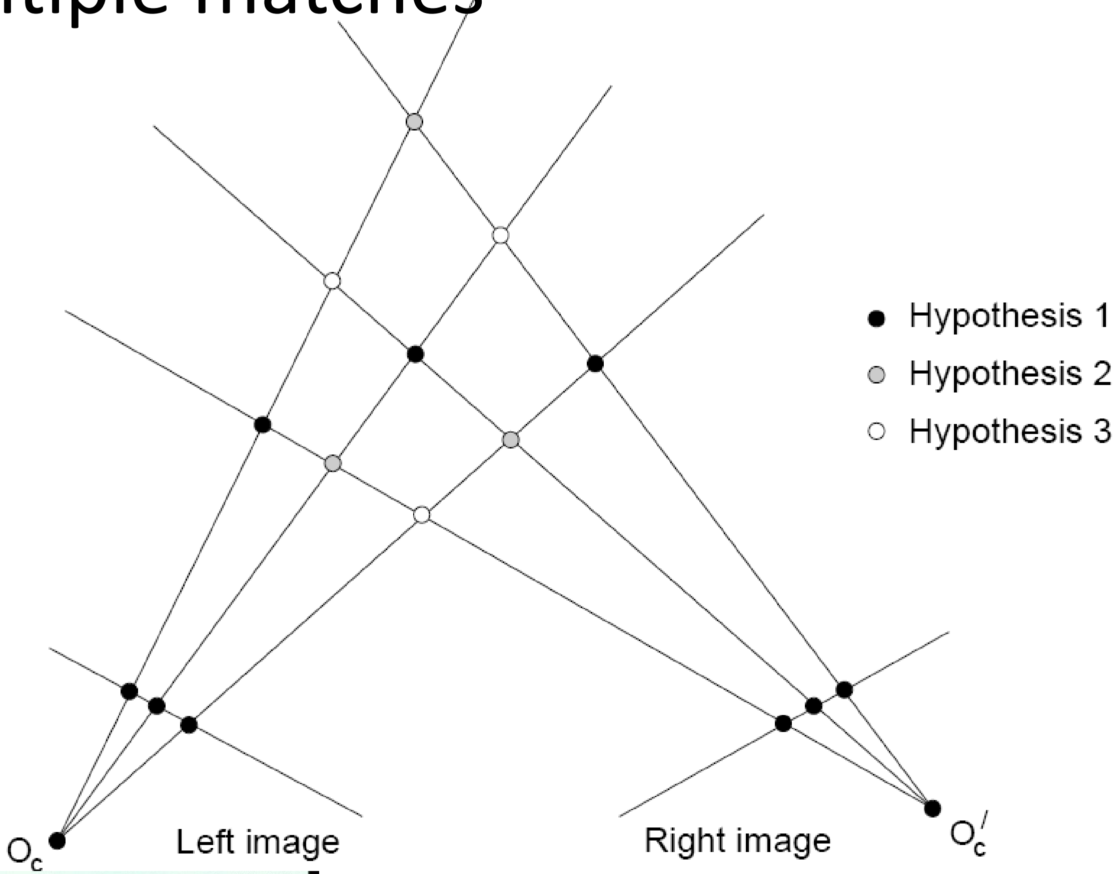


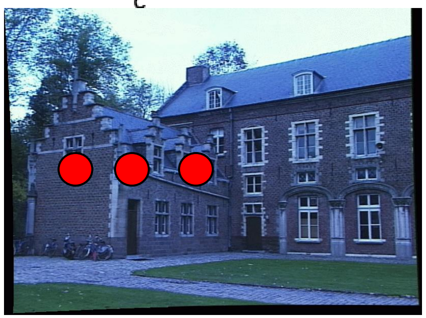
Textureless surfaces

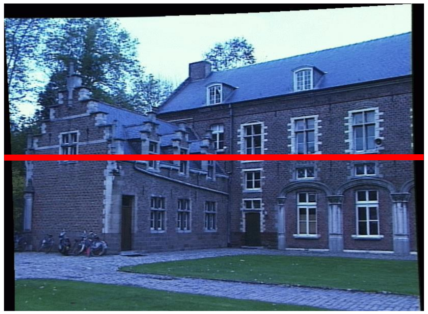
Occlusions, repetition

Non-Lambertian surfaces, specularities

Correspondence Problems: Multiple matches







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

How can we improve window-based matching?

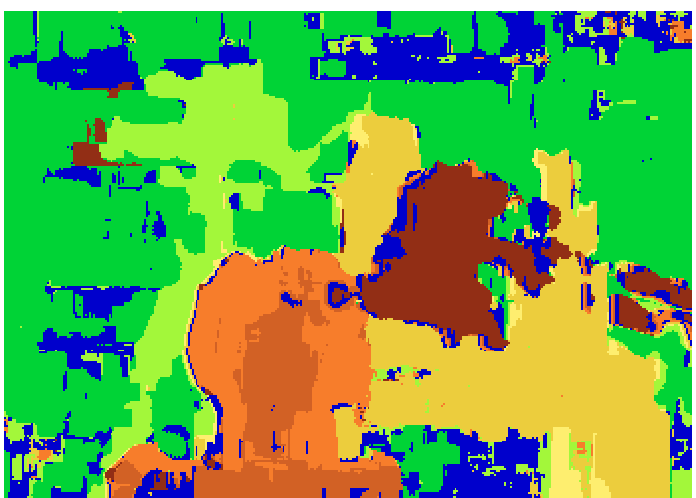
- Beyond the epipolar constraint, there are “soft” constraints to help identify corresponding points
  - Uniqueness
    - Only one match in right image for every point in left image
  - Ordering
    - Points on **same surface** will be in same order in both views
  - Disparity gradient
    - Disparity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

Grauman

Results with window search

Data

Window-based matching



Ground truth

Better methods exist...

Graph cuts

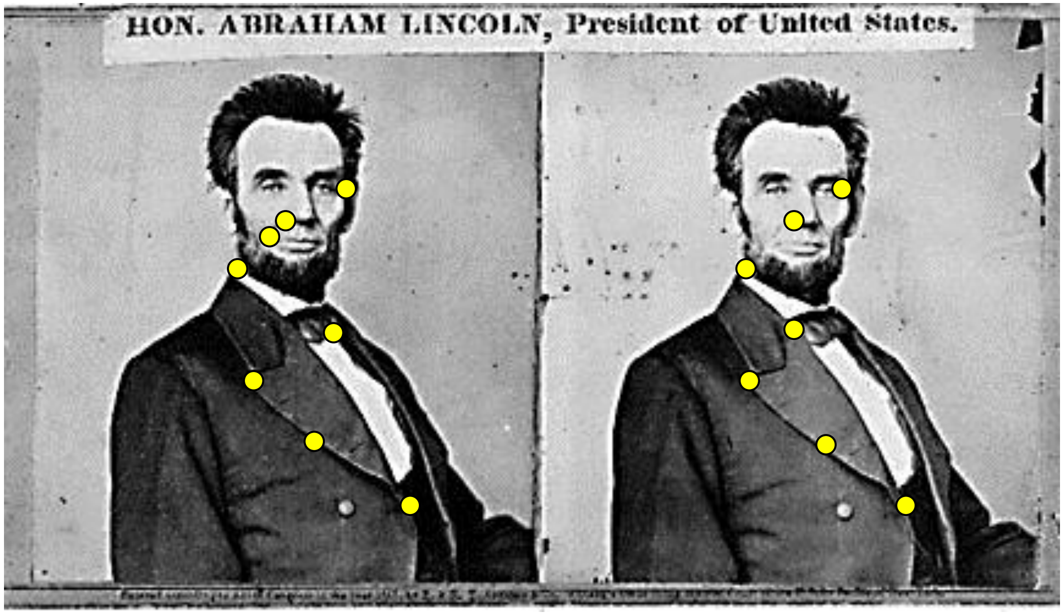
Ground truth

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 correspondence search

- Restrict search to sparse set of detected features
- Rather than pixel values (or lists of pixel values) use *feature descriptor* and an associated *similarity metrics*
- Still use epipolar geometry to narrow the search further

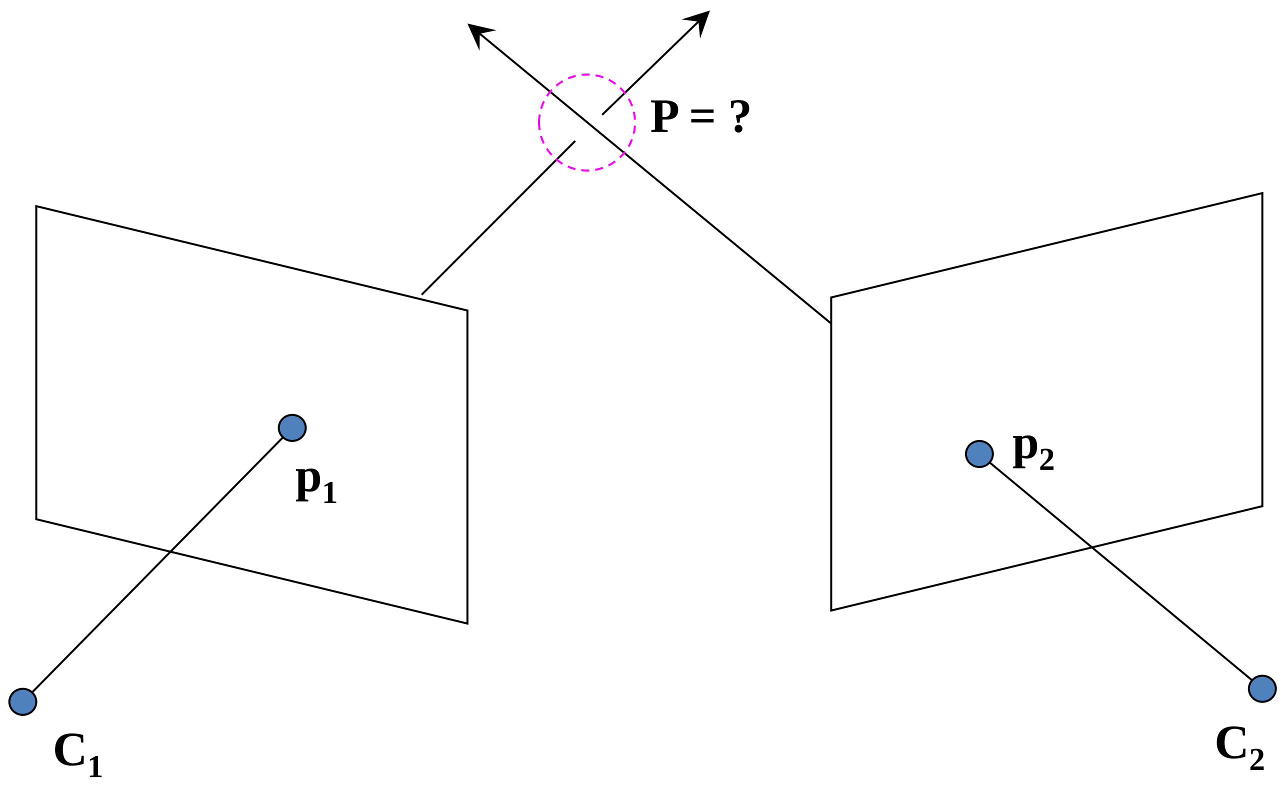


Stereo Vision

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

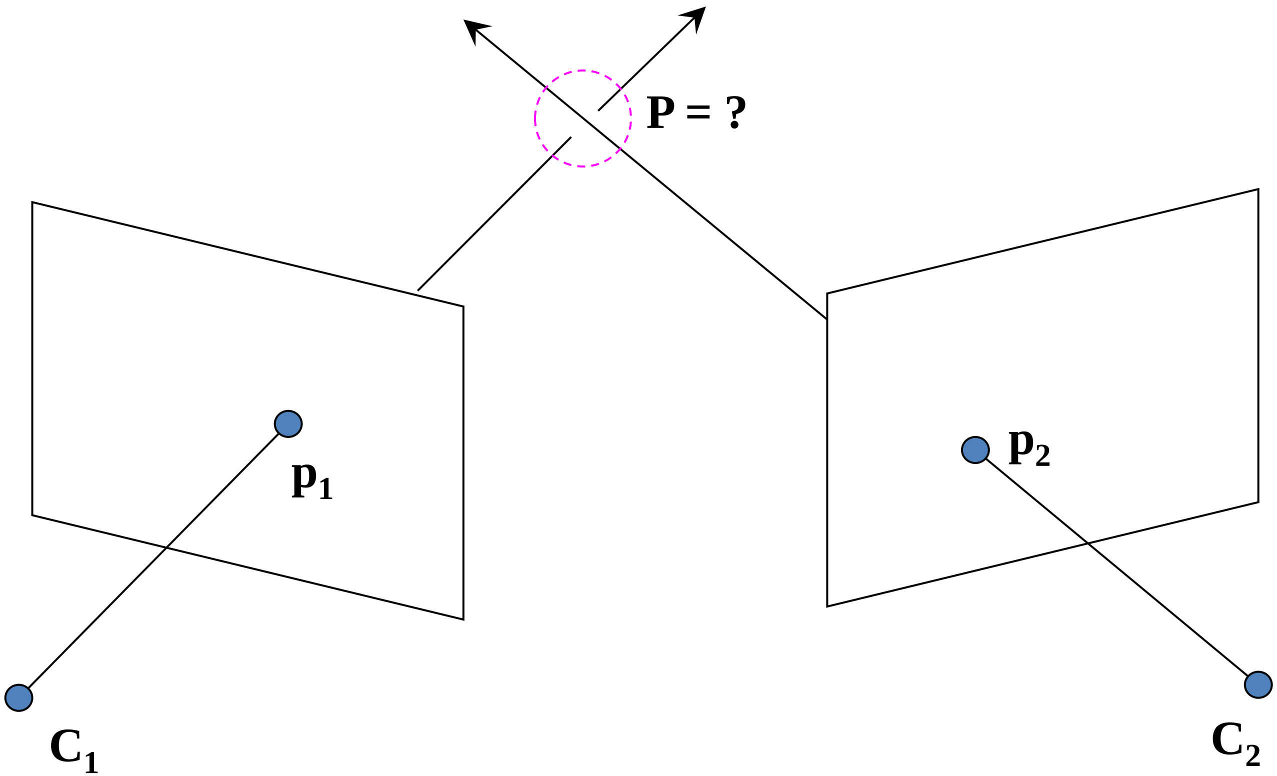
Triangulation

- Given the projections $\mathbf{p}_1$ and $\mathbf{p}_2$ of a 3D point ***P*** in two or more images (with known camera matrices R and T), find the coordinates of the 3D point



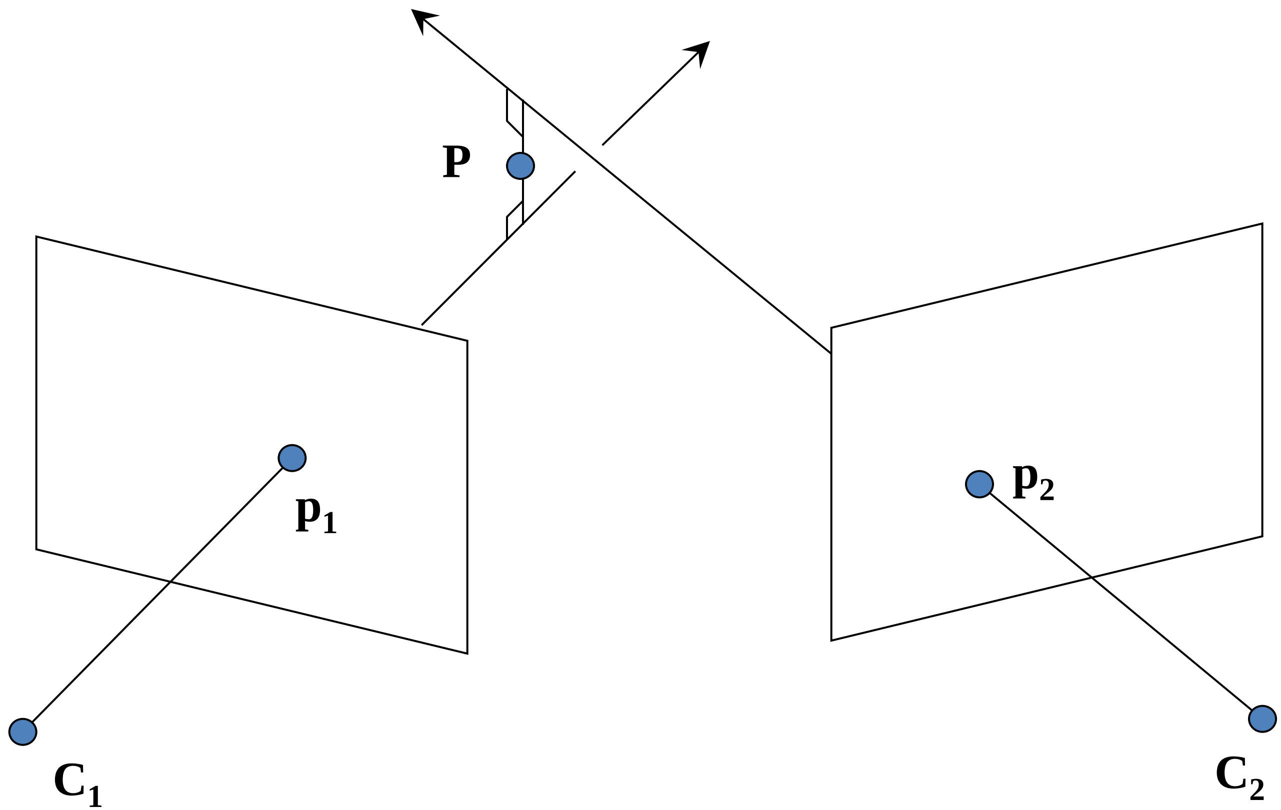
Triangulation

- 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 approach

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



Triangulation: Linear Approach

Left camera

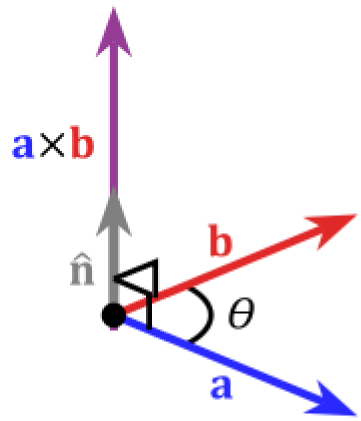
$$\lambda_1 \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} = K\left[I|0\right] \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \Rightarrow \lambda_1 p_1 = M_1 \cdot P$$

Right camera

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

Review: Cross Product (or Vector Product)

$$\vec{a} \times \vec{b} = \vec{c}$$



- Vector cross product takes two vectors and returns a third vector that is perpendicular to both inputs

$$\vec{a} \cdot \vec{c} = 0$$

$$\vec{b} \cdot \vec{c} = 0$$

- So here, $\boldsymbol{c}$ is perpendicular to both $\boldsymbol{a}$ and $\boldsymbol{b}$, which means the dot product = 0
- Also, recall that the cross product of two parallel vectors = 0

- The vector **cross product** can also be expressed as the product of a **skew-symmetric matrix** and a vector

$$\mathbf{a} \times \mathbf{b} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = [\mathbf{a}_\times]\mathbf{b}$$

Triangulation: Linear Approach

Left camera

$$\Rightarrow \lambda_1 p_1 = M_1 \cdot P \qquad \Rightarrow p_1 \times M_1 \cdot P = 0 \qquad \Rightarrow [p_{1\times}] M_1 \cdot P = 0$$

Right camera

$$\Rightarrow \lambda_2 p_2 = M_2 \cdot P \qquad \Rightarrow p_2 \times M_2 \cdot P = 0 \qquad \Rightarrow [p_{2\times}] M_2 \cdot P = 0$$

Cross product as matrix multiplication:

$$\mathbf{a} \times \mathbf{b} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = [\mathbf{a}_\times]\mathbf{b}$$

Triangulation: Linear Approach

Left camera

$$\Rightarrow \lambda_1 p_1 = M_1 \cdot P \qquad \Rightarrow p_1 \times M_1 \cdot P = 0 \qquad \Rightarrow [p_{1\times}] M_1 \cdot P = 0$$

Right camera

$$\Rightarrow \lambda_2 p_2 = M_2 \cdot P \qquad \Rightarrow p_2 \times M_2 \cdot P = 0 \qquad \Rightarrow [p_{2\times}] M_2 \cdot P = 0$$

Two independent equations each in terms of three unknown entries of $\boldsymbol{P}$

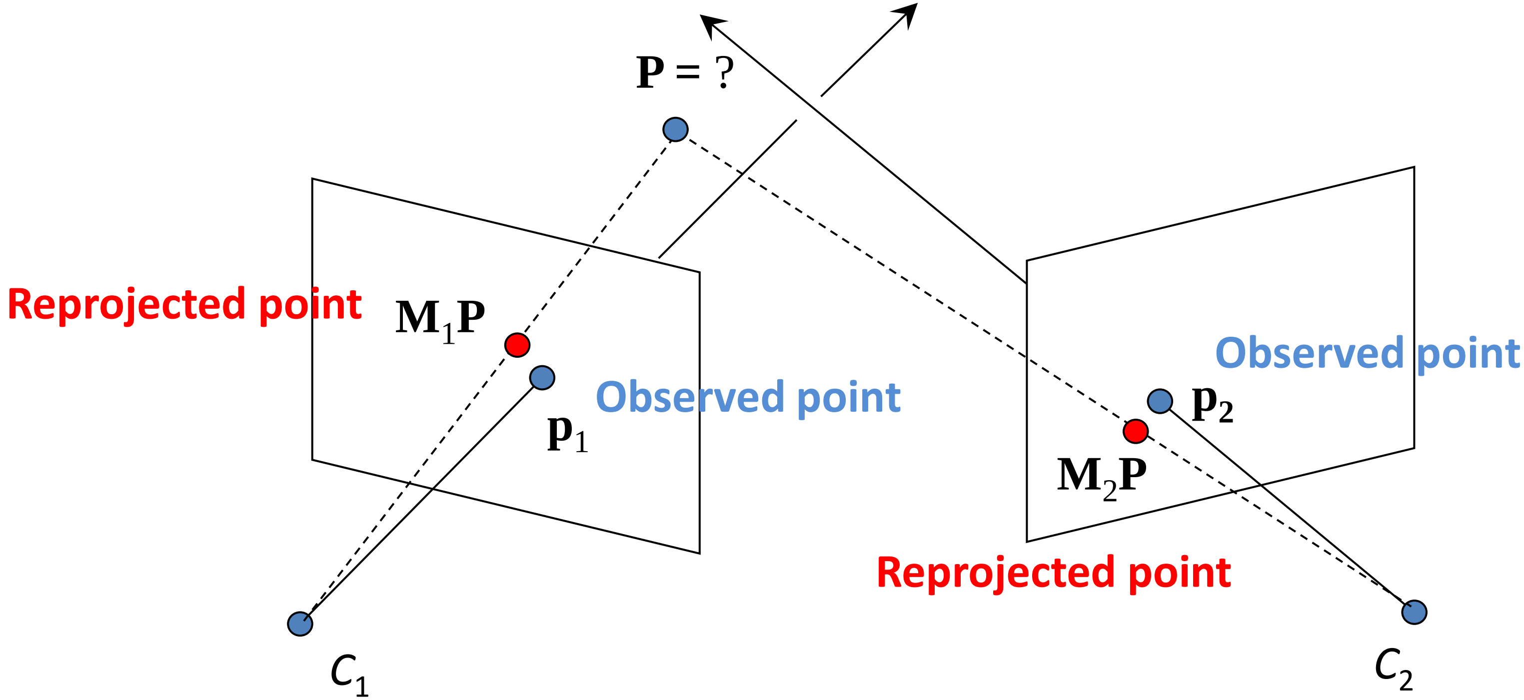
Triangulation: Nonlinear approach

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

$$SSRE = d^2(p_1, M_1P) + d^2(p_2, M_2P)$$

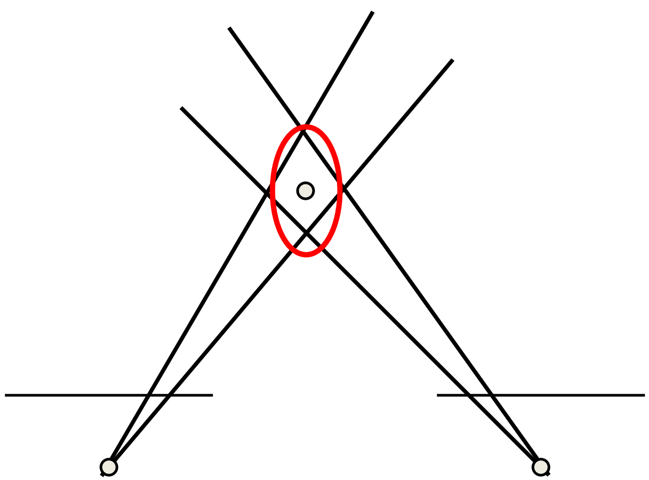
where $d(p_1, M_1P) = \|p_1 - M_1P\|$ is called ***Reprojection Error***.

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

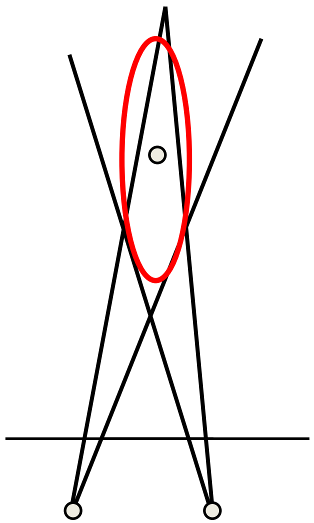


Choosing the Baseline

- 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

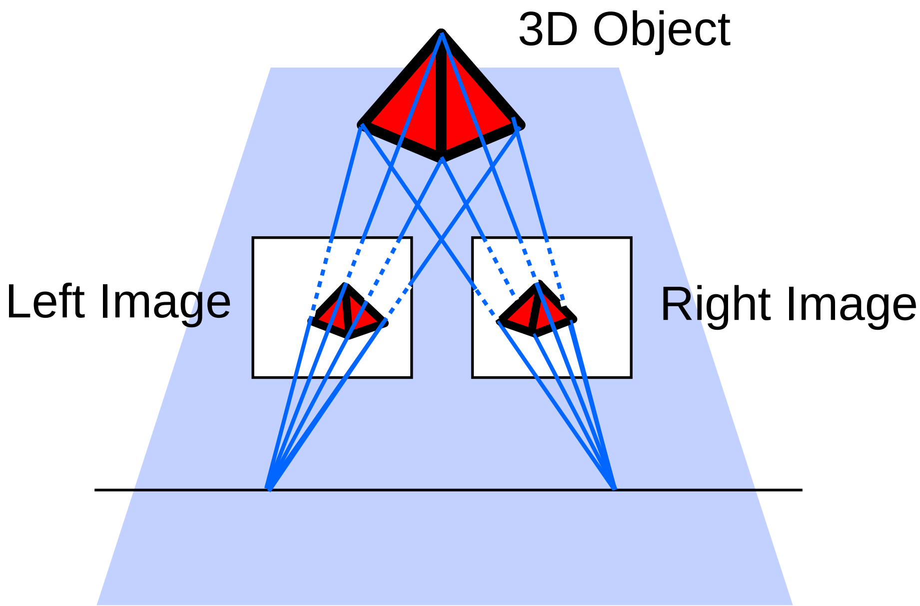


Large Baseline



Small Baseline

Stereo Vision - summary



1. Stereo camera calibration ⇨ compute camera relative pose
2. Epipolar rectification ⇨ align images & epipolar lines
3. Search for correspondences
4. Output: compute stereo triangulation or disparity map
5. Consider how baseline & image resolution affect accuracy of depth estimates

Szeliski book: Chapter 11
Autonomous Mobile Robot book: Chapter 4.2.5