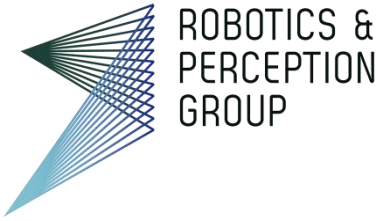
University of Zurich[UZH] | ETH zürich
Institute of Informatics – Institute of Neuroinformatics



Lecture 07
Multiple View Geometry 1

Davide Scaramuzza
http://rpg.ifi.uzh.ch/

Multiple View Geometry spans 4 lectures

19.09.2019	**Lecture 01 - Introduction to Computer Vision and Visual Odometry**	Davide Scaramuzza
26.09.2019	**Lecture 02 - Image Formation 1: perspective projection and camera models** **Exercise 01 - Augmented reality wireframe cube**	Davide Scaramuzza Daniel & Mathias Gehrig
03.10.2019	**Lecture 03 - Image Formation 2: camera calibration algorithms** **Exercise 02 - PnP problem**	Davide Scaramuzza Daniel & Mathias Gehrig
10.10.2019	**Lecture 04 - Filtering & Edge detection**	Davide Scaramuzza
17.10.2019	**Lecture 05 - Point Feature Detectors, Part 1** **Exercise 03 - Harris detector + descriptor + matching**	Davide Scaramuzza Daniel & Mathias Gehrig
24.10.2019	**Lecture 06 - Point Feature Detectors, Part 2** **Exercise 04 - SIFT detector + descriptor + matching**	Davide Scaramuzza Daniel & Mathias Gehrig
31.10.2019	**Lecture 07 - Multiple-view geometry** **Exercise 05 - Stereo vision: rectification, epipolar matching, disparity, triangulation**	Davide Scaramuzza Daniel & Mathias Gehrig
07.11.2019	**Lecture 08 - Multiple-view geometry 2** **Exercise 06 - Eight-Point Algorithm**	Antonio Loquercio Daniel & Mathias Gehrig
14.11.2019	**Lecture 09 - Multiple-view geometry 3 (Part 1)**	Davide Scaramuzza
21.11.2019	**Lecture 10 - Multiple-view geometry 3 (Part 2)** **Exercise session: Intermediate VO Integration**	Davide Scaramuzza Daniel & Mathias Gehrig
28.11.2019	**Lecture 11 - Optical Flow and Tracking (Lucas-Kanade)** **Exercise 08 - Lucas-Kanade tracker**	Davide Scaramuzza Daniel & Mathias Gehrig
05.12.2019	**Lecture 12 - Place recognition and 3D Reconstruction** **Exercise session: Deep Learning Tutorial**	Davide Scaramuzza Daniel & Mathias Gehrig
12.12.2019	**Lecture 13 - Visual inertial fusion** **Exercise 09 - Bundle Adjustment**	Davide Scaramuzza Daniel & Mathias Gehrig
19.12.2019	**Lecture 14 - Event based vision** After the lecture, we will Scaramuzza's lab. Departure from lecture room at 12:00 via tram 10. **Exercise session: Final VO Integration**	Davide Scaramuzza Daniel & Mathias Gehrig

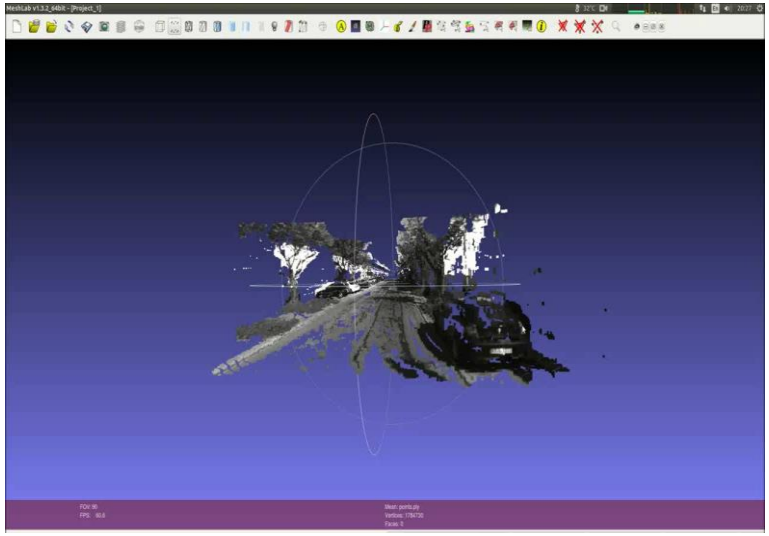
Next week lecture given by Antonio Loquercio

19.09.2019	**Lecture 01 - Introduction to Computer Vision and Visual Odometry**	Davide Scaramuzza
26.09.2019	**Lecture 02 - Image Formation 1: perspective projection and camera models** **Exercise 01 - Augmented reality wireframe cube**	Davide Scaramuzza Daniel & Mathias Gehrig
03.10.2019	**Lecture 03 - Image Formation 2: camera calibration algorithms** **Exercise 02 - PnP problem**	Davide Scaramuzza Daniel & Mathias Gehrig
10.10.2019	**Lecture 04 - Filtering & Edge detection**	Davide Scaramuzza
17.10.2019	**Lecture 05 - Point Feature Detectors, Part 1** **Exercise 03 - Harris detector + descriptor + matching**	Davide Scaramuzza Daniel & Mathias Gehrig
24.10.2019	**Lecture 06 - Point Feature Detectors, Part 2** **Exercise 04 - SIFT detector + descriptor + matching**	Davide Scaramuzza Daniel & Mathias Gehrig
31.10.2019	**Lecture 07 - Multiple-view geometry** **Exercise 05 - Stereo vision: rectification, epipolar matching, disparity, triangulation**	Davide Scaramuzza Daniel & Mathias Gehrig
07.11.2019	**Lecture 08 - Multiple-view geometry 2** **Exercise 06 - Eight-Point Algorithm**	Antonio Loquercio Daniel & Mathias Gehrig
14.11.2019	**Lecture 09 - Multiple-view geometry 3 (Part 1)**	Davide Scaramuzza
21.11.2019	**Lecture 10 - Multiple-view geometry 3 (Part 2)** **Exercise session: Intermediate VO Integration**	Davide Scaramuzza Daniel & Mathias Gehrig
28.11.2019	**Lecture 11 - Optical Flow and Tracking (Lucas-Kanade)** **Exercise 08 - Lucas-Kanade tracker**	Davide Scaramuzza Daniel & Mathias Gehrig
05.12.2019	**Lecture 12 - Place recognition and 3D Reconstruction** **Exercise session: Deep Learning Tutorial**	Davide Scaramuzza Daniel & Mathias Gehrig
12.12.2019	**Lecture 13 - Visual inertial fusion** **Exercise 09 - Bundle Adjustment**	Davide Scaramuzza Daniel & Mathias Gehrig
19.12.2019	**Lecture 14 - Event based vision** After the lecture, we will Scaramuzza's lab. Departure from lecture room at 12:00 via tram 10. **Exercise session: Final VO Integration**	Davide Scaramuzza Daniel & Mathias Gehrig

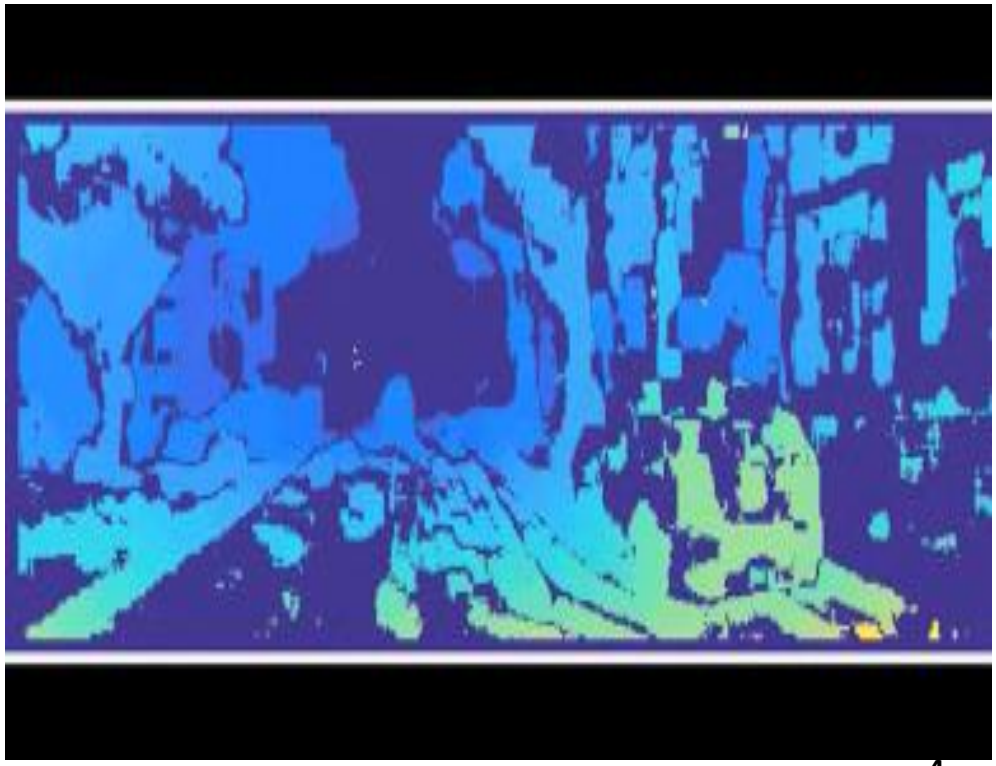
Lab Exercise 5 - Today afternoon

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

3D point cloud



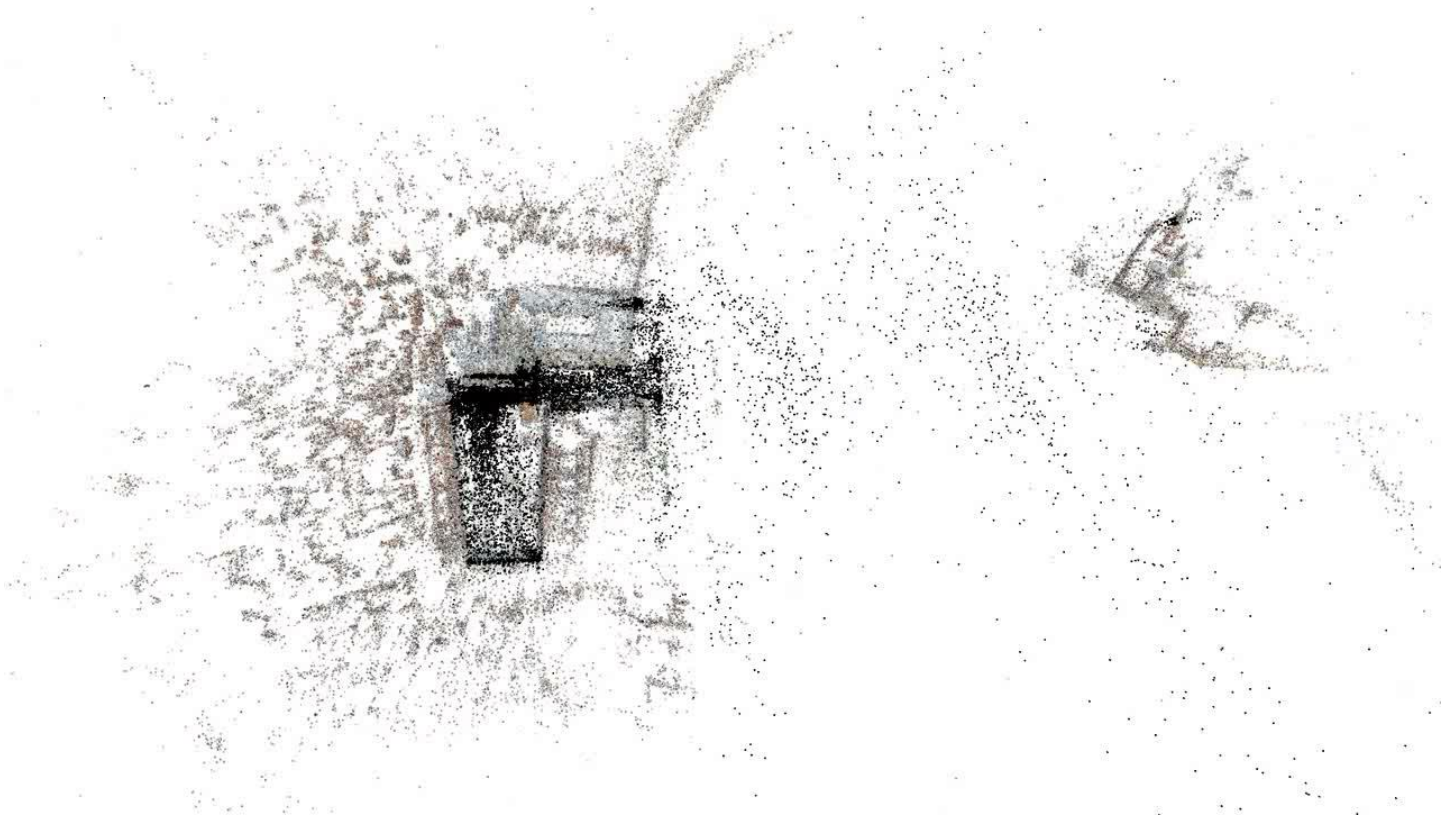
Disparity map



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



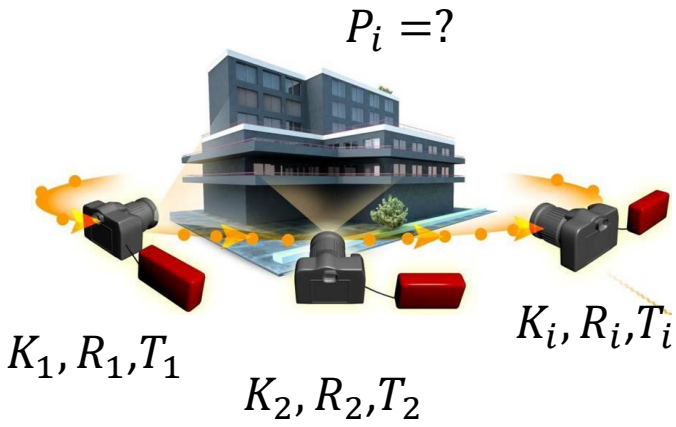
San Marco square, Venice

14,079 images, 4,515,157 points

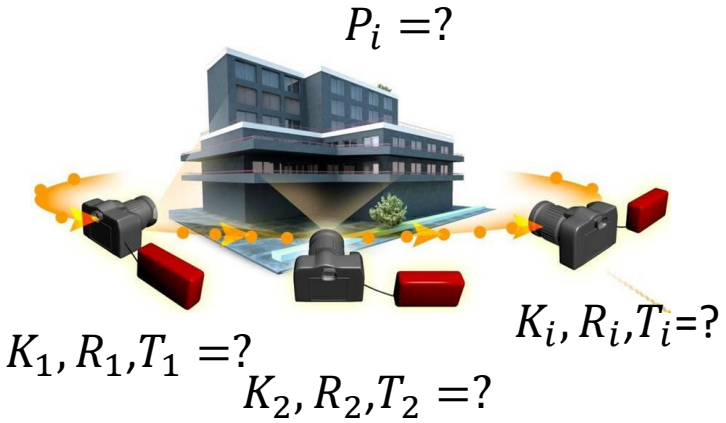
Paper: Building Rome in a Day, University of Washington, 2009 – Most influential paper of 2009

Multiple View Geometry

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

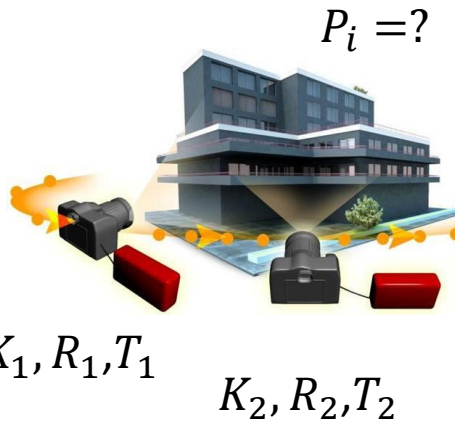


- **Structure From Motion:**
  - **Assumptions**: none (K, T, and R are **unknown**).
  - **Goal**: Recover simultaneously 3D scene structure and camera poses (up to scale) from multiple images

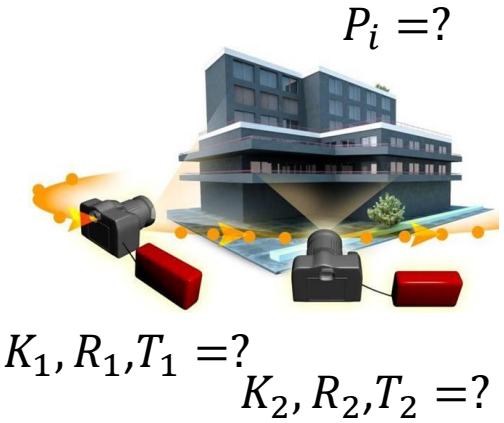


2-View Geometry

- **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

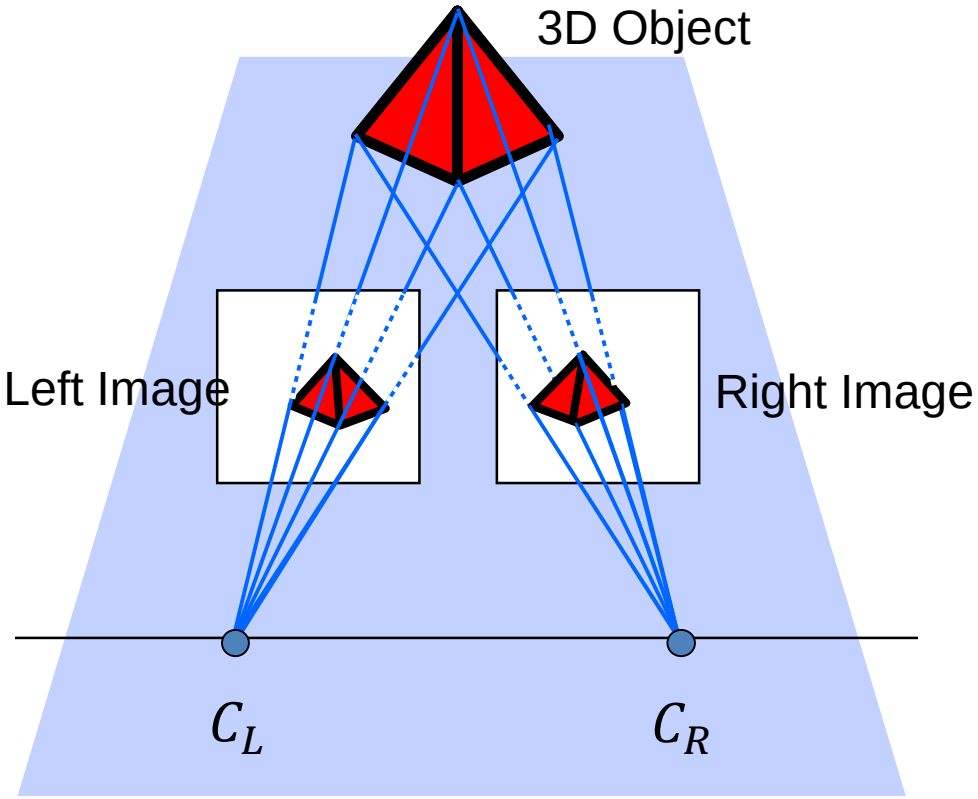


Today's outline

- Stereo Vision
- Epipolar Geometry

Depth from Stereo

- From a single camera, we can ***back-project*** the **ray** on which an 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?

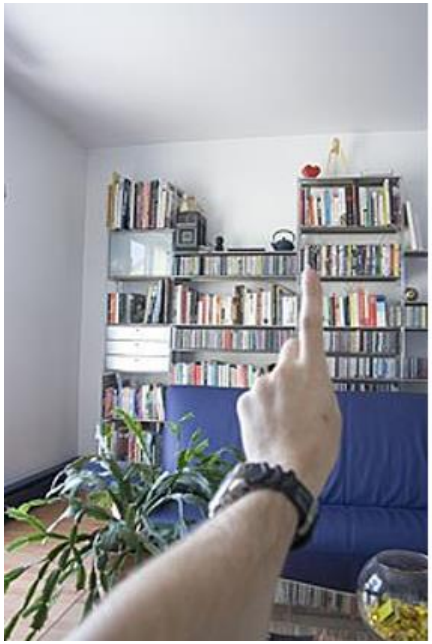


Image from the left eye

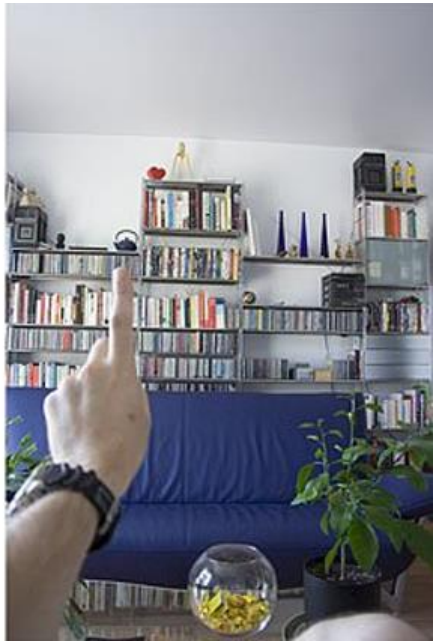
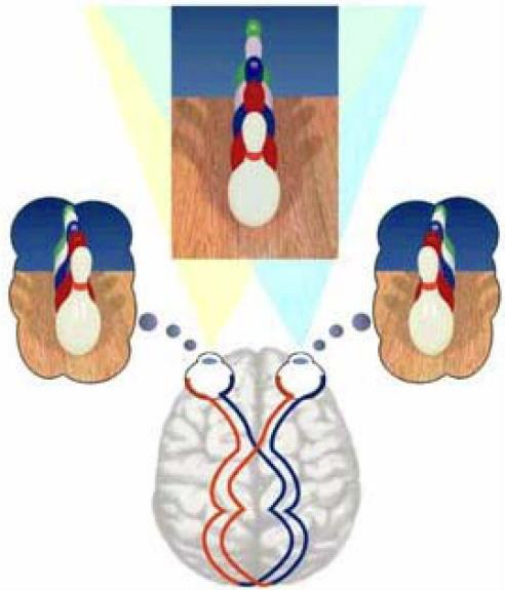
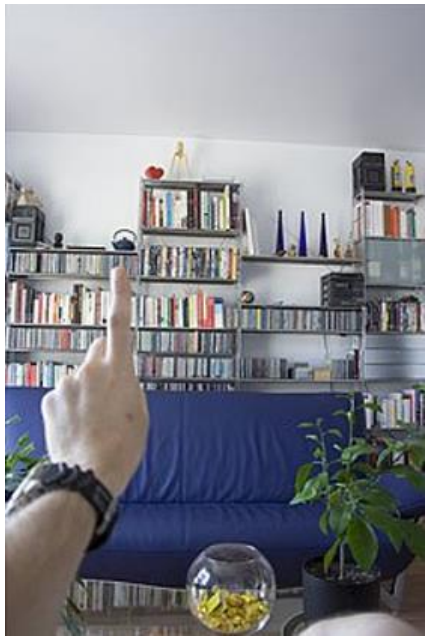


Image from the right eye



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 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?



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 distortion is also removed. This process is called «**rectification**». What happens if you wear a pair of mirrors for a week?

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

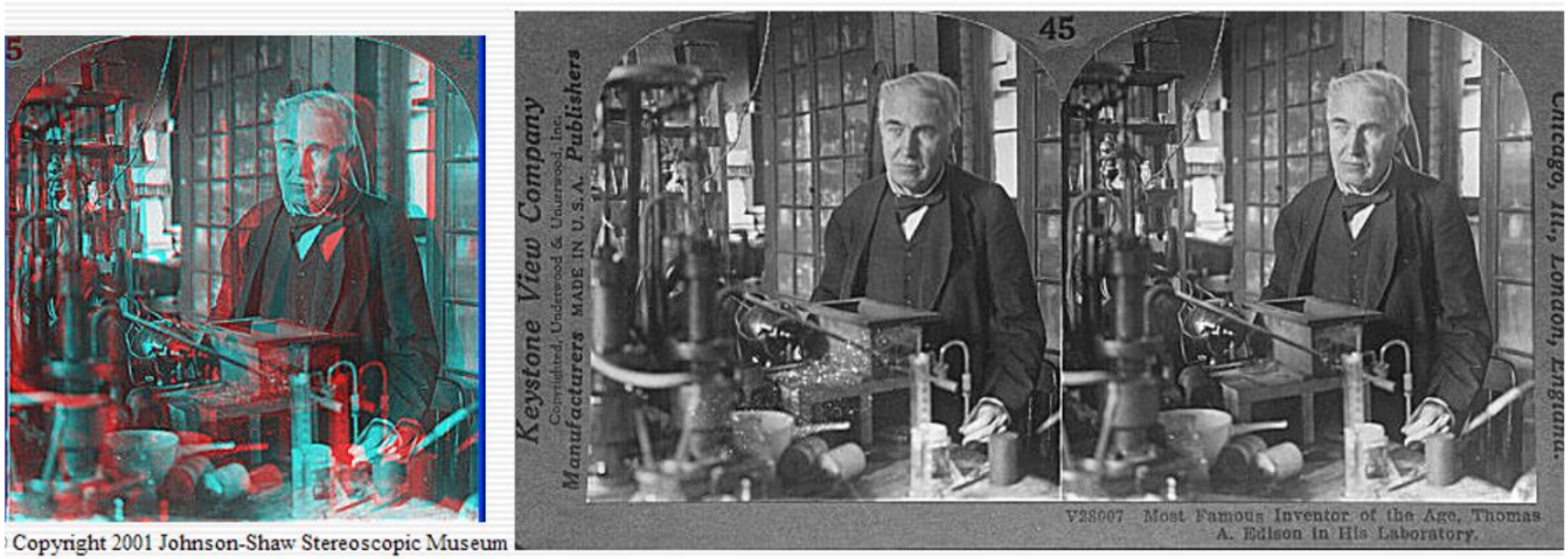
Applications: Stereo photography and stereo viewers

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



Applications: Stereograms

Exploit disparity as depth cue using single image

Image from magiceye.com

Applications: Stereograms

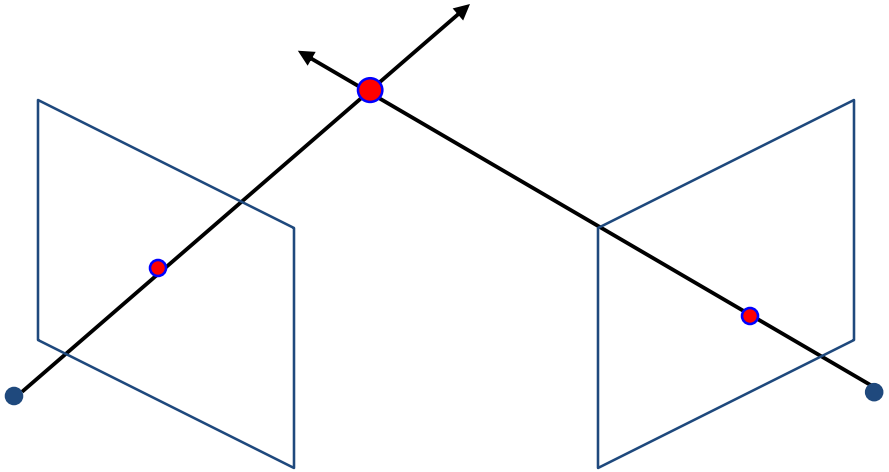
Exploit disparity as depth cue using single image

Image from magiceye.com

Stereo Vision

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

Stereo Vision: basic idea



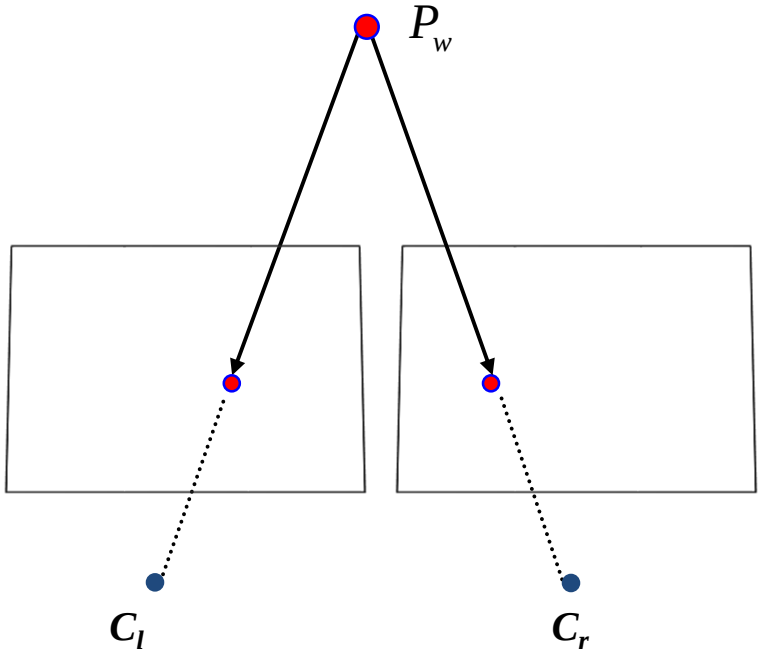
Basic Principle: Triangulation

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

Stereo Vision: basic idea

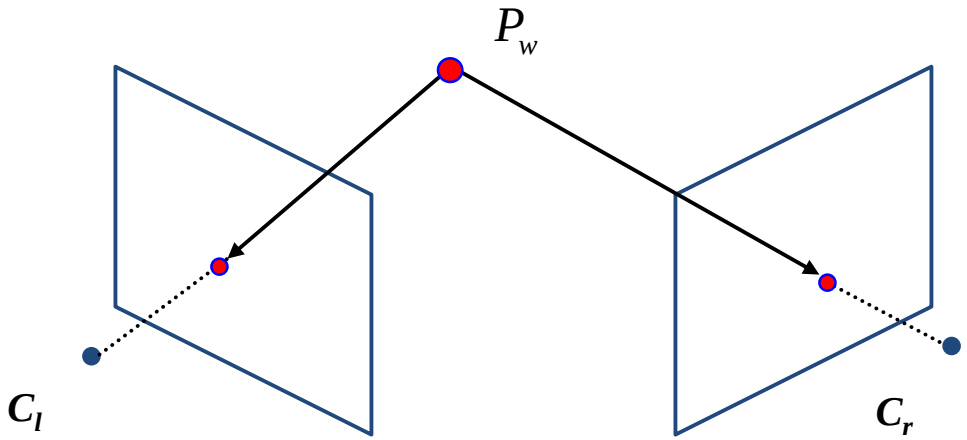
Simplified case

(identical cameras and aligned)



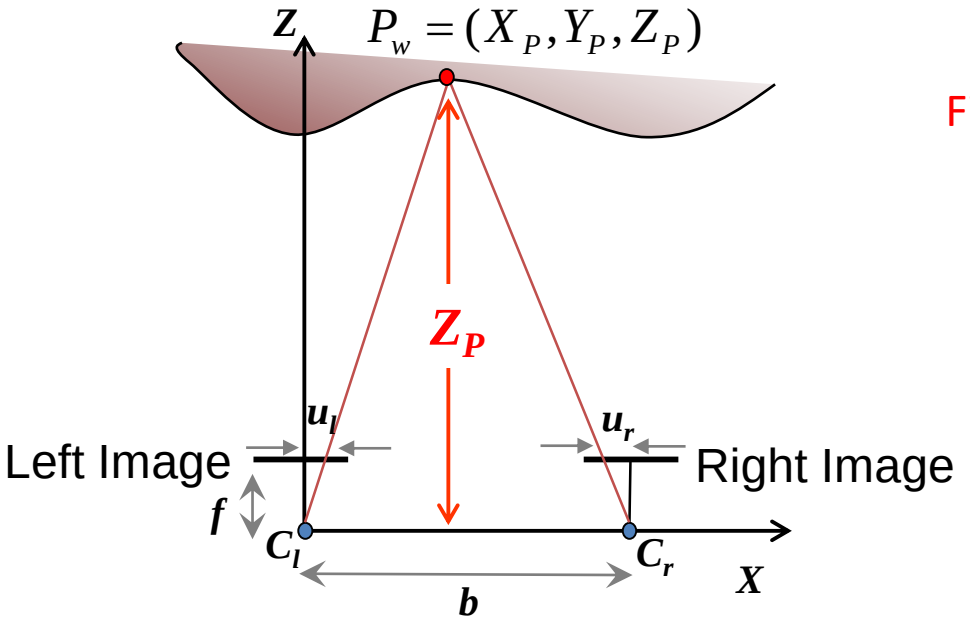
General case

(non identical cameras and not aligned)



Stereo Vision - The simplified case

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



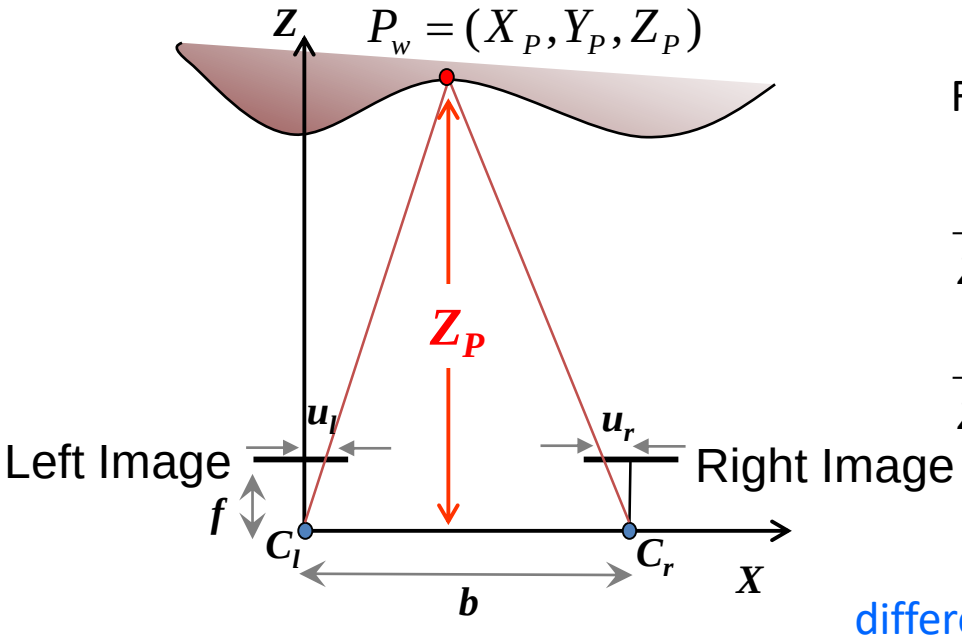
Baseline

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:

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

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

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

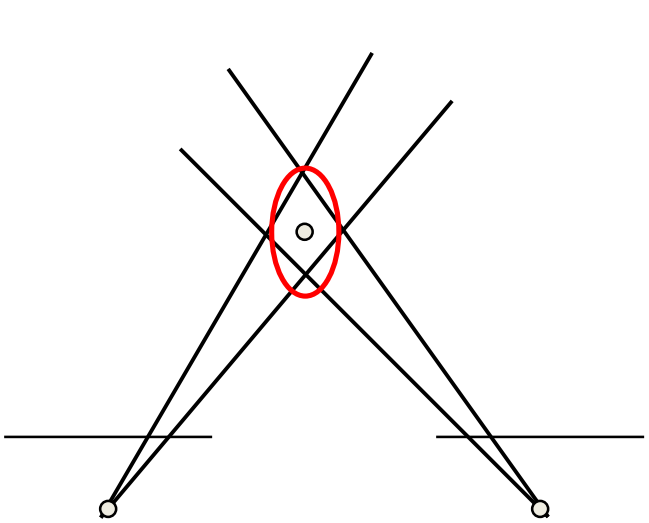
Disparity

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

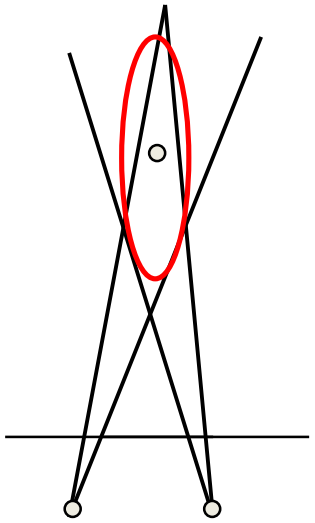
1. What's the max disparity of a stereo camera?
2. What's the disparity of a point at infinity?
3. How does the depth uncertainty depend on the disparity?
4. How does it depend on the depth estimate?
5. How can we increase the accuracy of a stereo system?

Choosing the Baseline

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



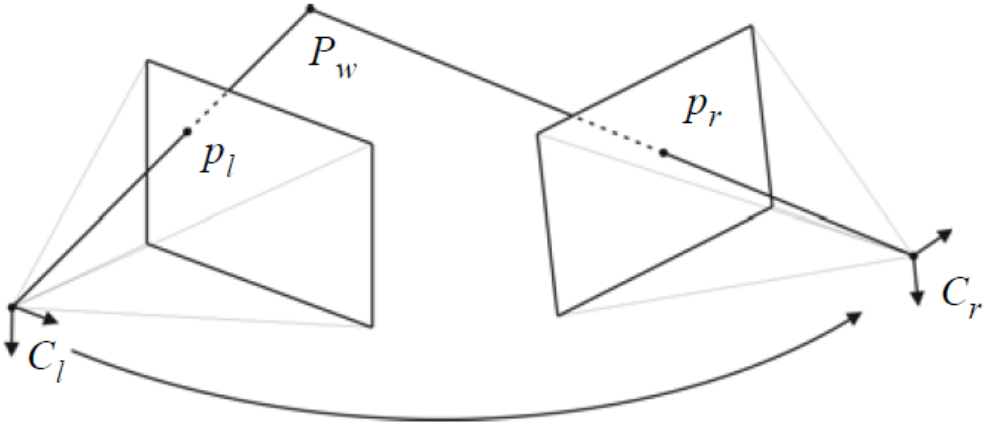
Large Baseline



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?



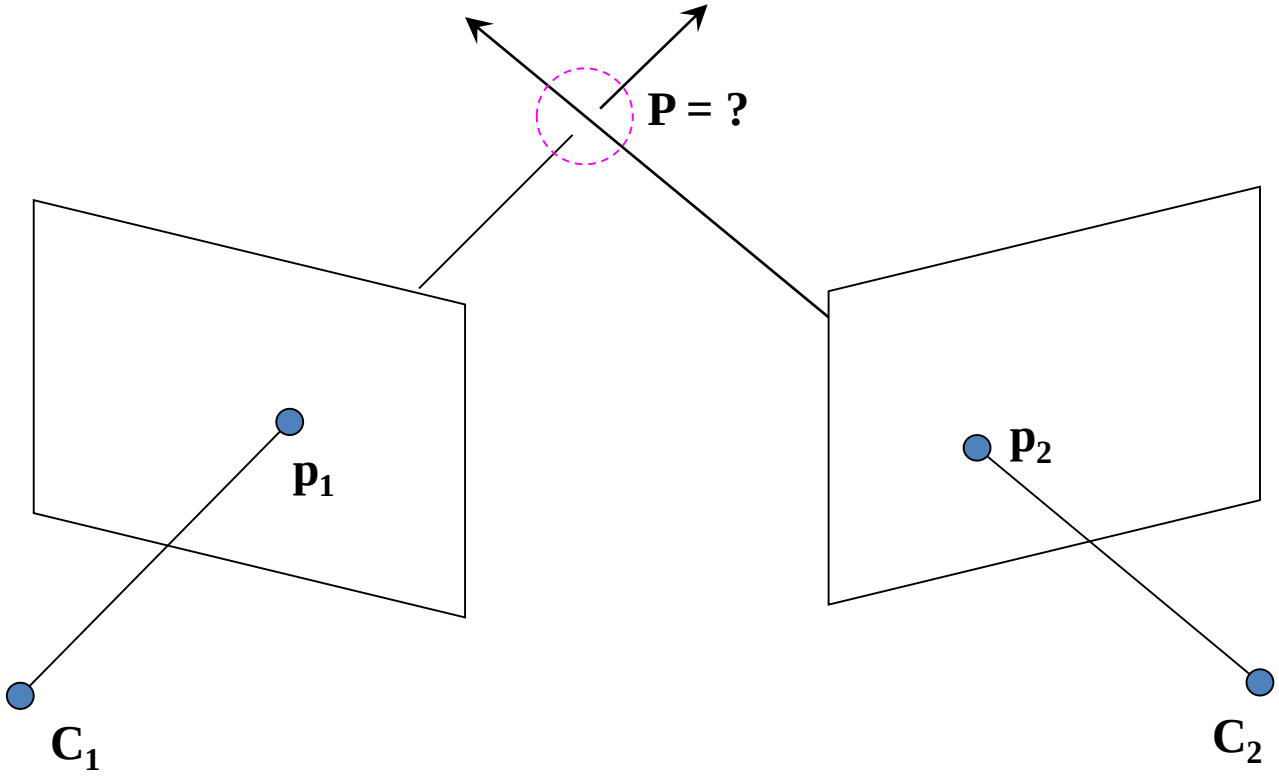
- In order to be able to use a stereo camera, we need the
  - **Extrinsic parameters** (relative rotation and translation)
  - **Instrinsic parameters** (focal length, optical center, radial distortion of each camera)

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

  ⇨ How do we compute the relative pose?

Triangulation

- "Triangulation" is the problem of determining the 3D position of a point given a set of corresponding image locations and known camera poses.
- 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 won't meet exactly, so we can only compute an approximation.



Triangulation: linear approximation

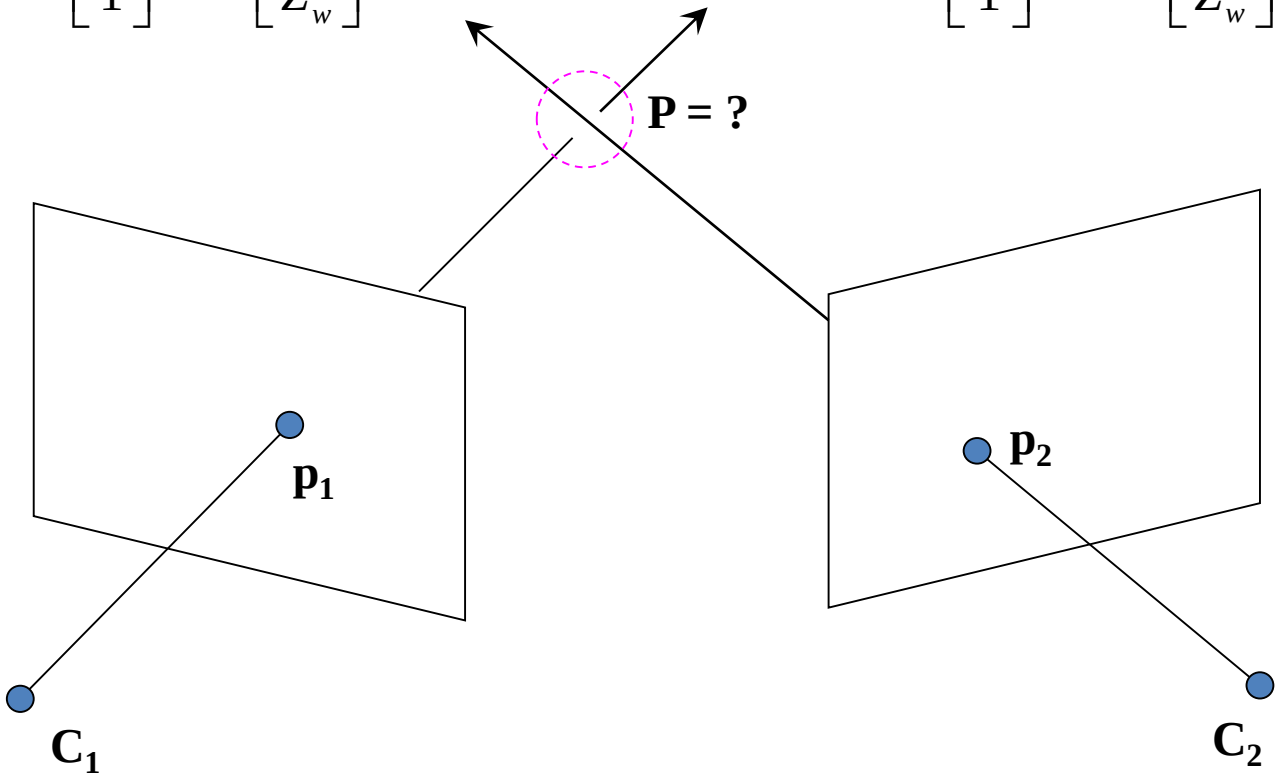
- We construct the system of equations of the left and right cameras, and solve it:

Left camera (assumed as world frame)

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

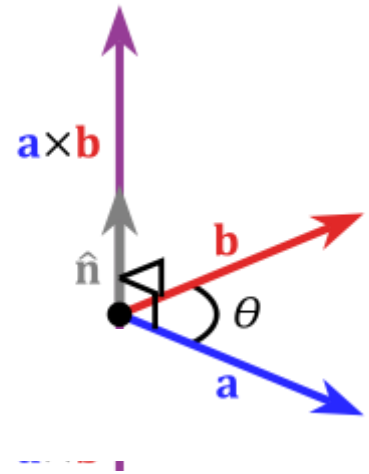
Right camera:

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



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 approximation

Left camera

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

Right camera

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

Triangulation: linear approximation

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:

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

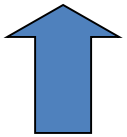
Triangulation: linear approximation

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

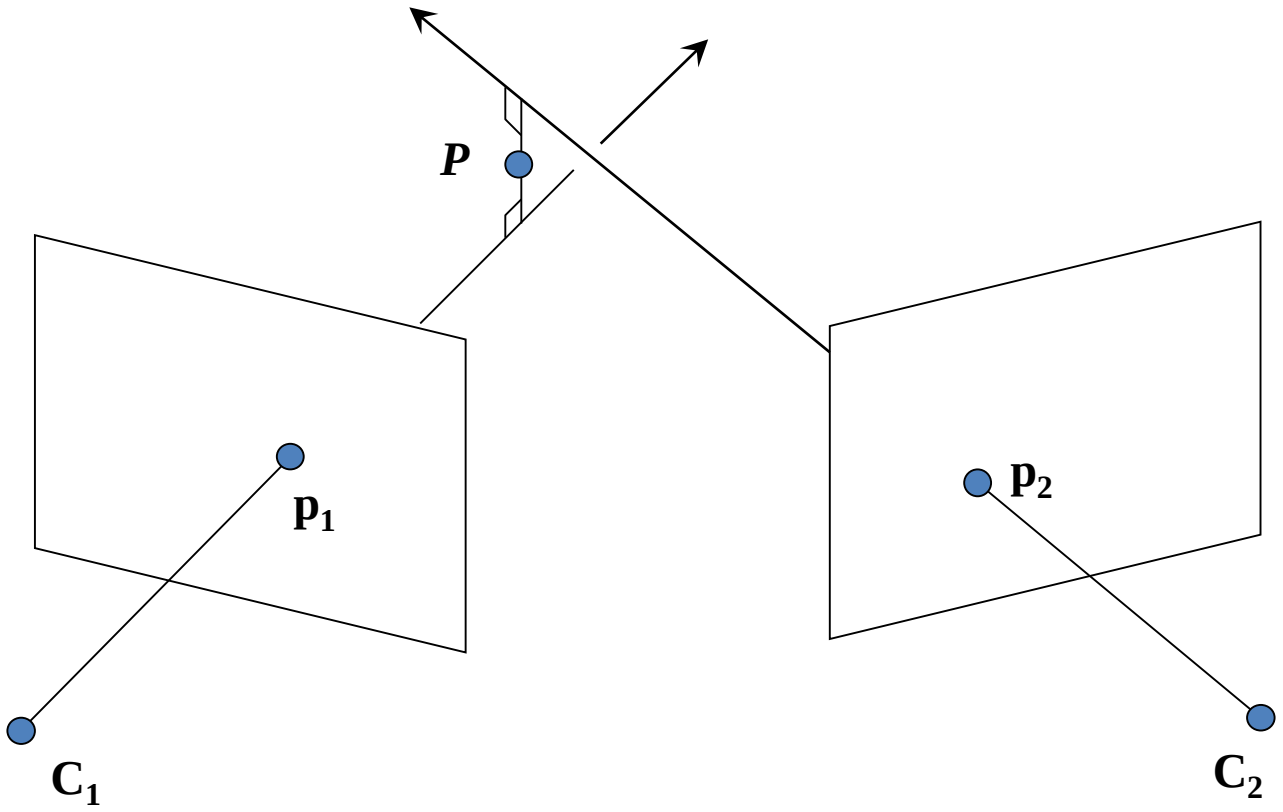
$$\Rightarrow \lambda_2 p_2 = M_2 \cdot P \qquad \Rightarrow p_2 \times M_2 \cdot P = 0 \qquad \Rightarrow [p_{2\times}] M_2 \cdot P = 0$$



Two independent equations each in terms of the three unknown elements of $\boldsymbol{P}$.
$\boldsymbol{P}$ can be determined using SVD, as we already did when we talked about DLT (see Lecture 03)

Geometric interpretation of linear approximation

P is computed as the midpoint of the shortest segment connecting the two back-projected rays.

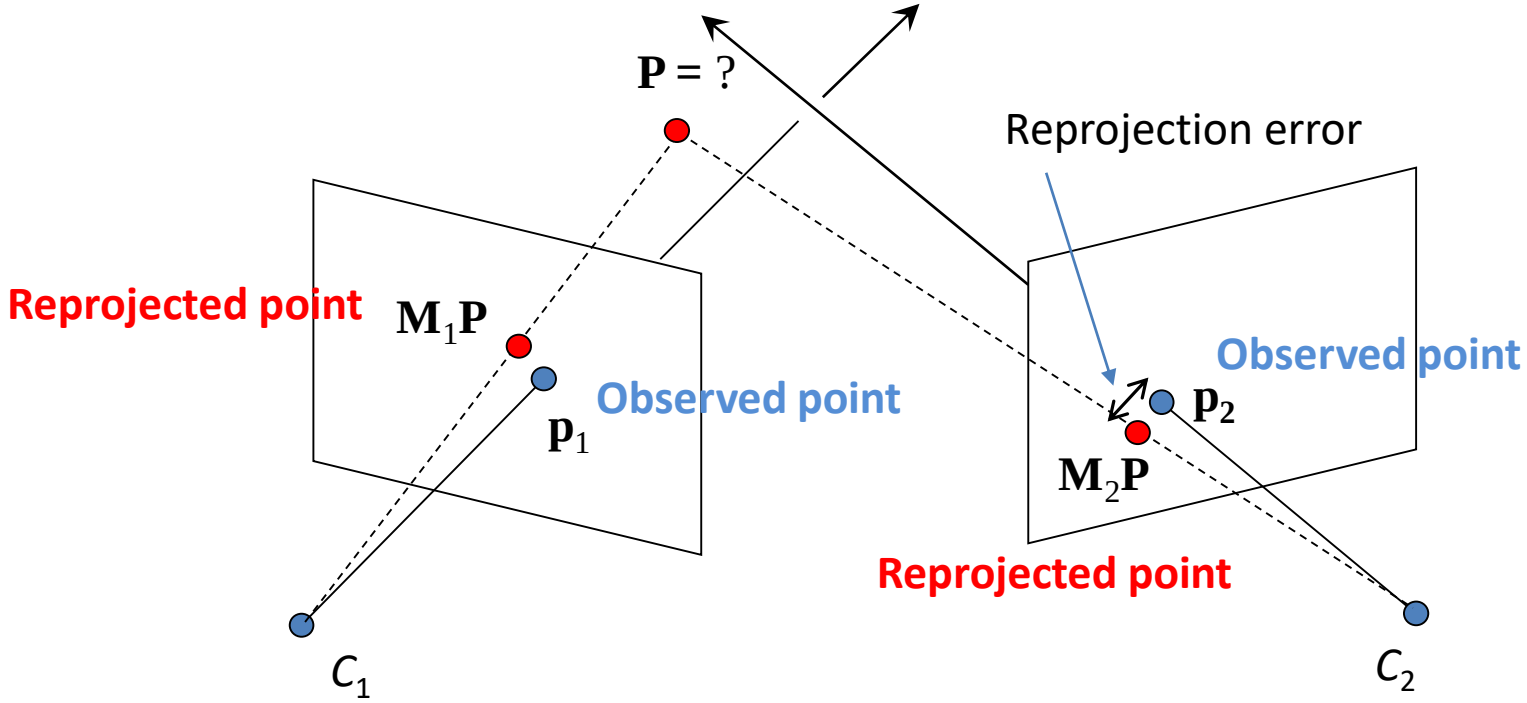


Triangulation: Nonlinear approximation

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

$$SSRE = \|p_1 - \pi_1(P)\|^2 + \|p_2 - \pi_2(P)\|^2$$

where $\|p_i - \pi_i(P)\|$ is called ***Reprojection Error***. $\pi_i(P)$ is the projection of P onto the i^{th} camera plane. In practice, we **initialize *P* using the linear approximation** and **then we refine it by minimizing the SSRE** (using Gauss-Newton or Levenberg-Marquardt).



Stereo Vision

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

Correspondence Problem

Given a point p_L on left image, how do we find its correspondent p_R on the right image?

Left image

Right image

Correspondence Problem

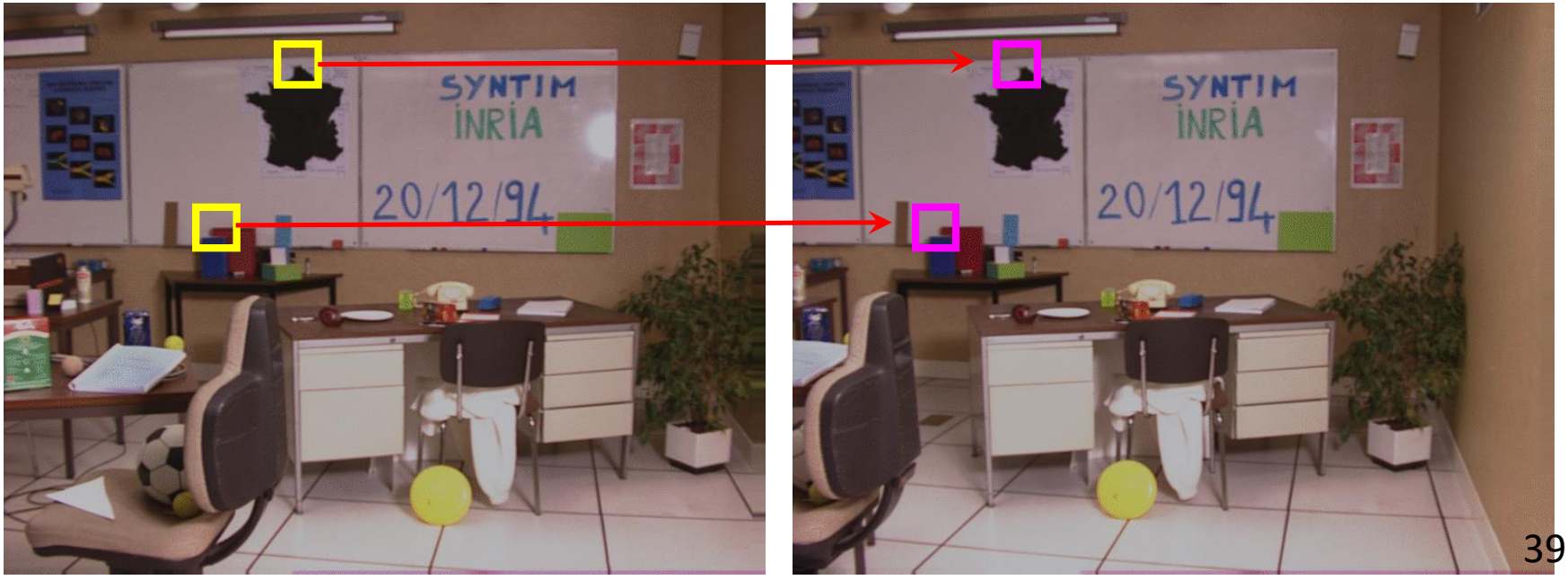
Given a point p_L on left image, how do we find its correspondent p_R on the right image?

Left image

Right image

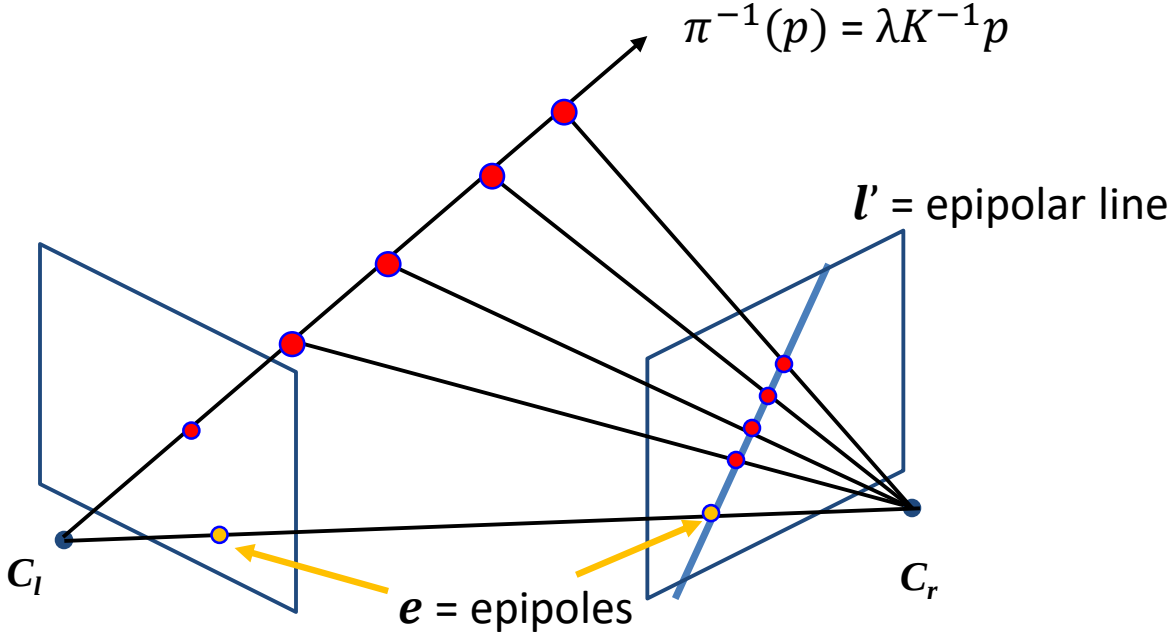
Correspondence Problem

- **Correspondence search**: identify image patches on the left & right images, which correspond to the same scene point.
- **Most used similarity measures for stereo correspondence search**
  - **(Z)NCC**
  - **(Z)SSD**
  - **(Z)SAD**
  - **Census Transform** (Census descriptor plus Hamming distance)



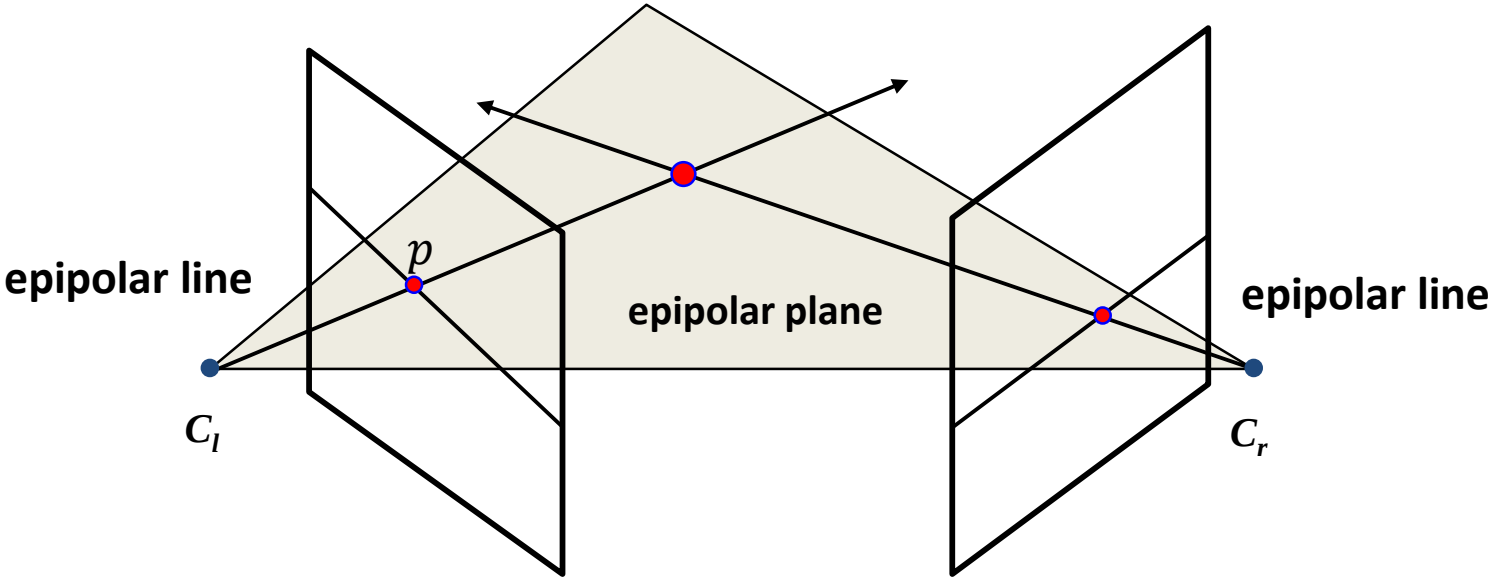
Correspondence Problem

- **Exhaustive** 2D image search can be computationally very expensive!
- Can we make the correspondence search 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



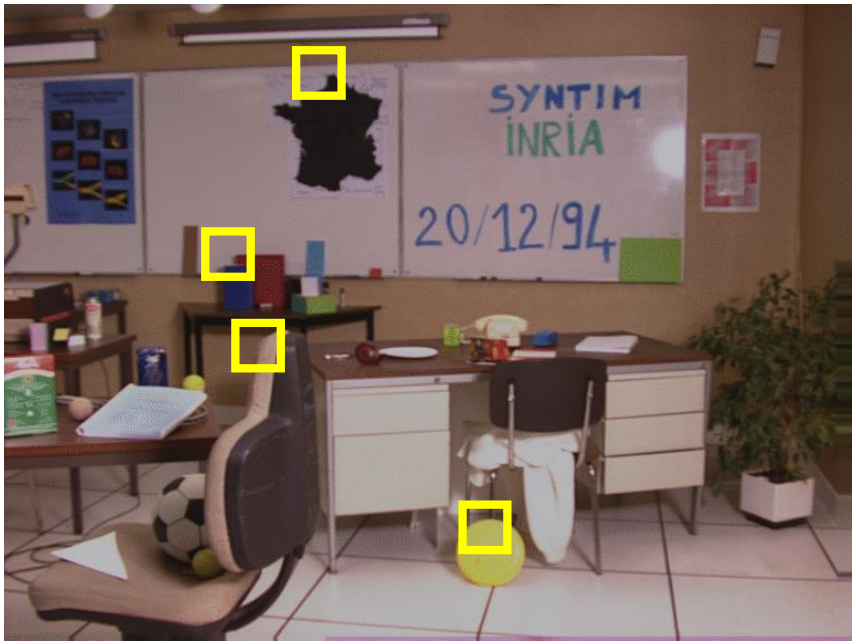
The Epipolar Constraint

- Given C_l, C_r and one image point p, an **epipolar plane** can be uniquely defined
- The intersections of the epipolar plane with the two image planes are called **epipolar lines**
- The epipolar lines determine the so-called **epipolar constraint:** the location of the corresponding point has to lie along the epipolar line.
- Why is this useful?
  - It reduces correspondence problem to 1D search along *the epipolar line*

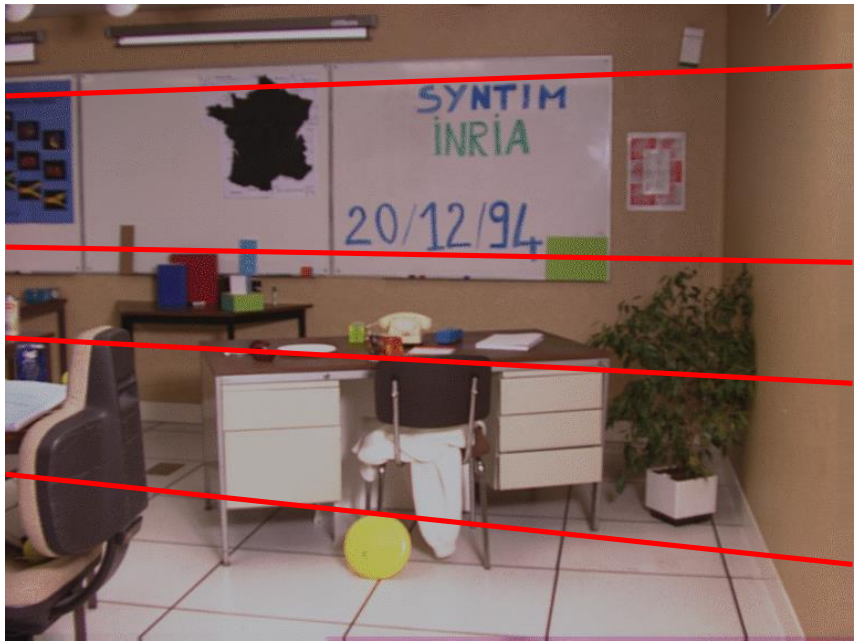


1D Correspondence Search via Epipolar Constraint

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



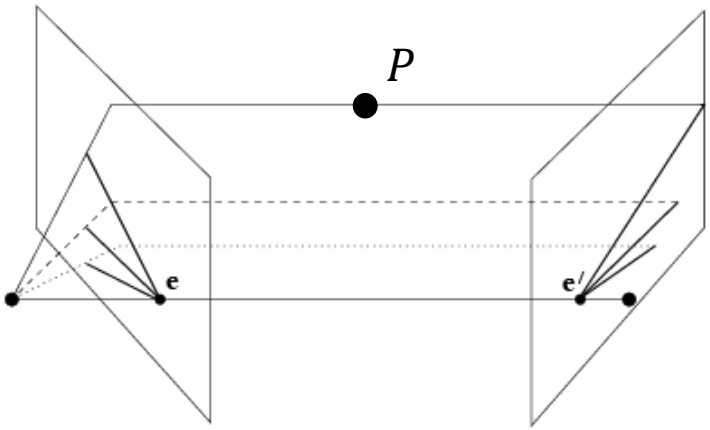
Left image

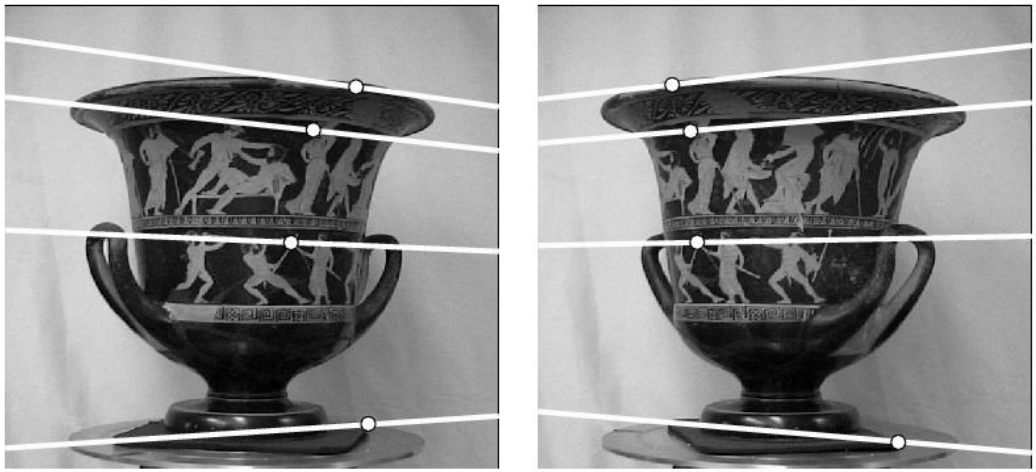


Right image

Example: converging cameras

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

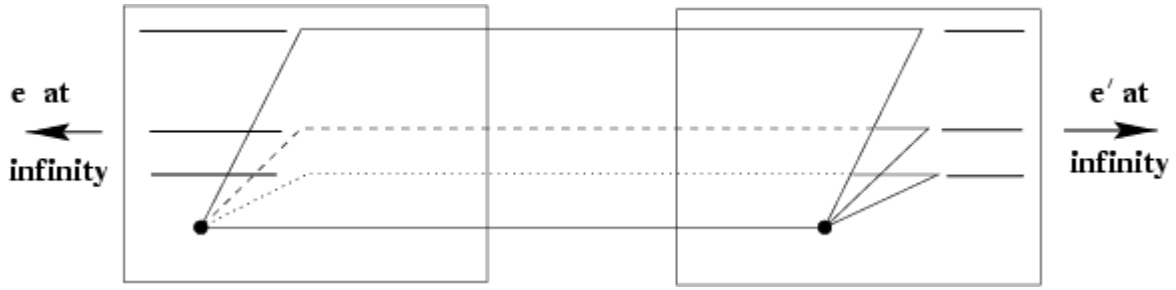




Left image

Right image

Example: identical and horizontally-aligned cameras

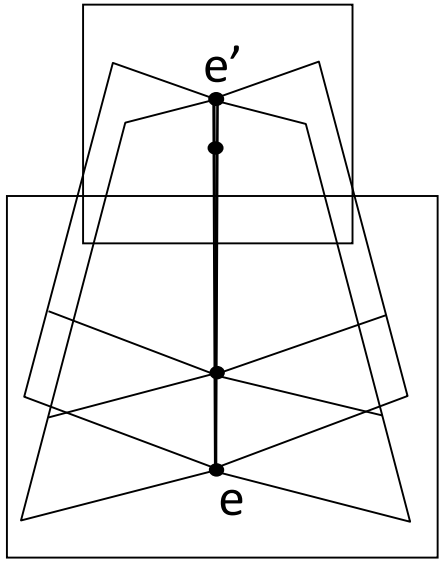


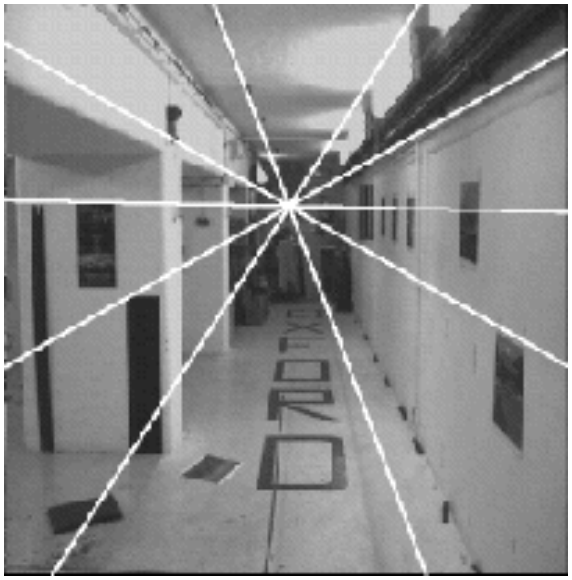
Left image

Right image

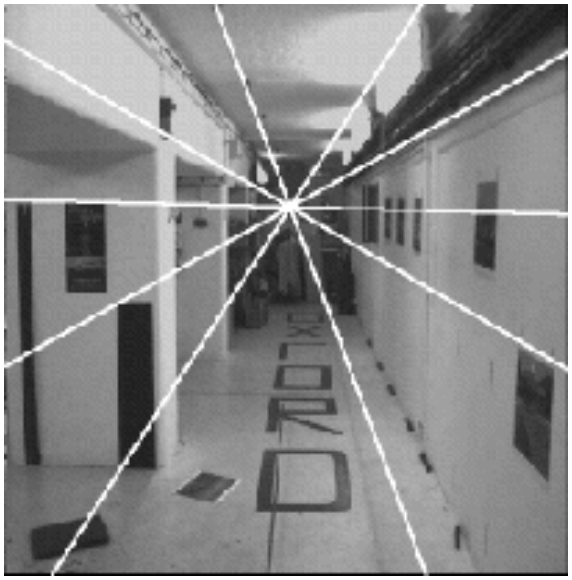
Example: forward motion (parallel to the optical axis)

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





Left image



Right image

Stereo Vision

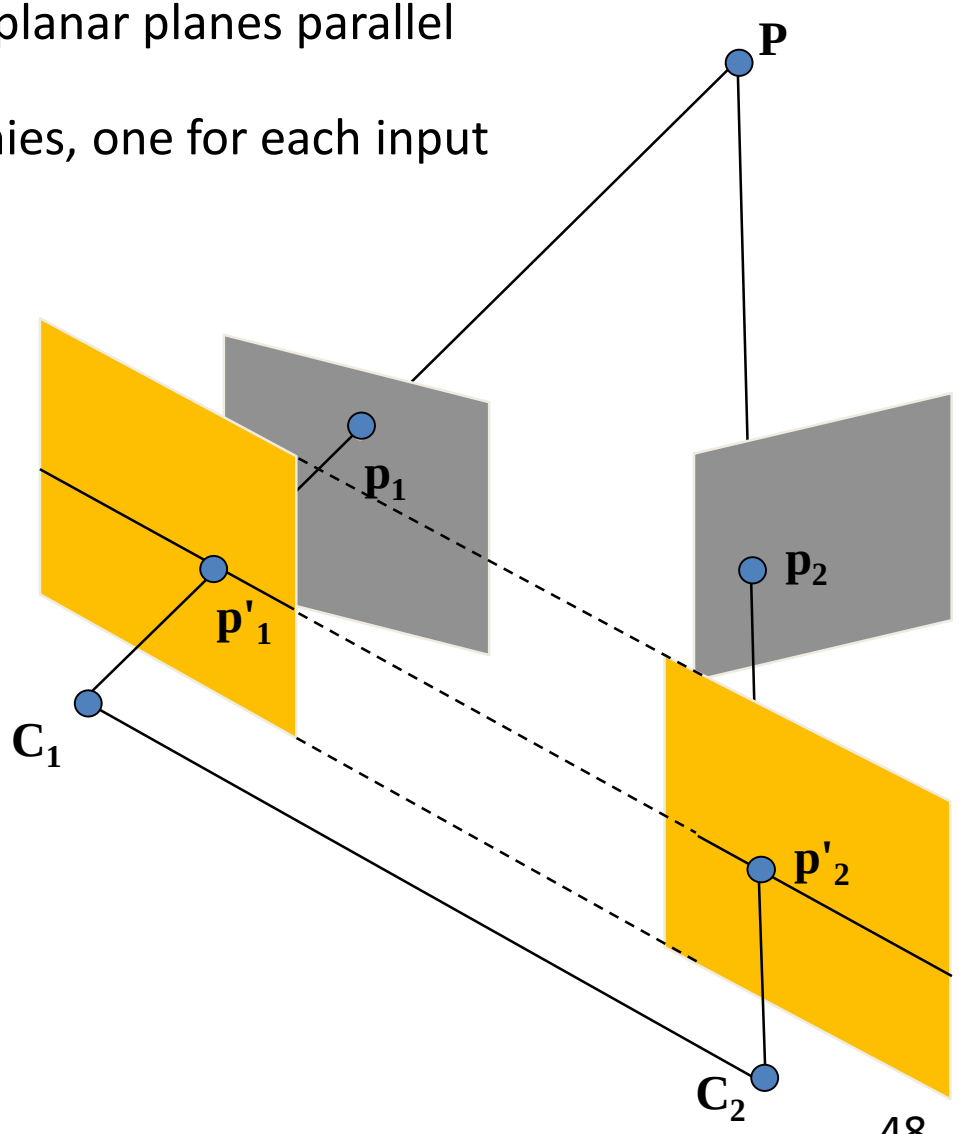
- Simplified case
- General case
- Correspondence problem
- Stereo rectification
- Triangulation

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 because then the correspondence search can be made very efficient (only search the point along the same scanlines)
- Stereo rectification warps the left and right images into new “rectified” images, whose epipolar lines are aligned to the baseline.

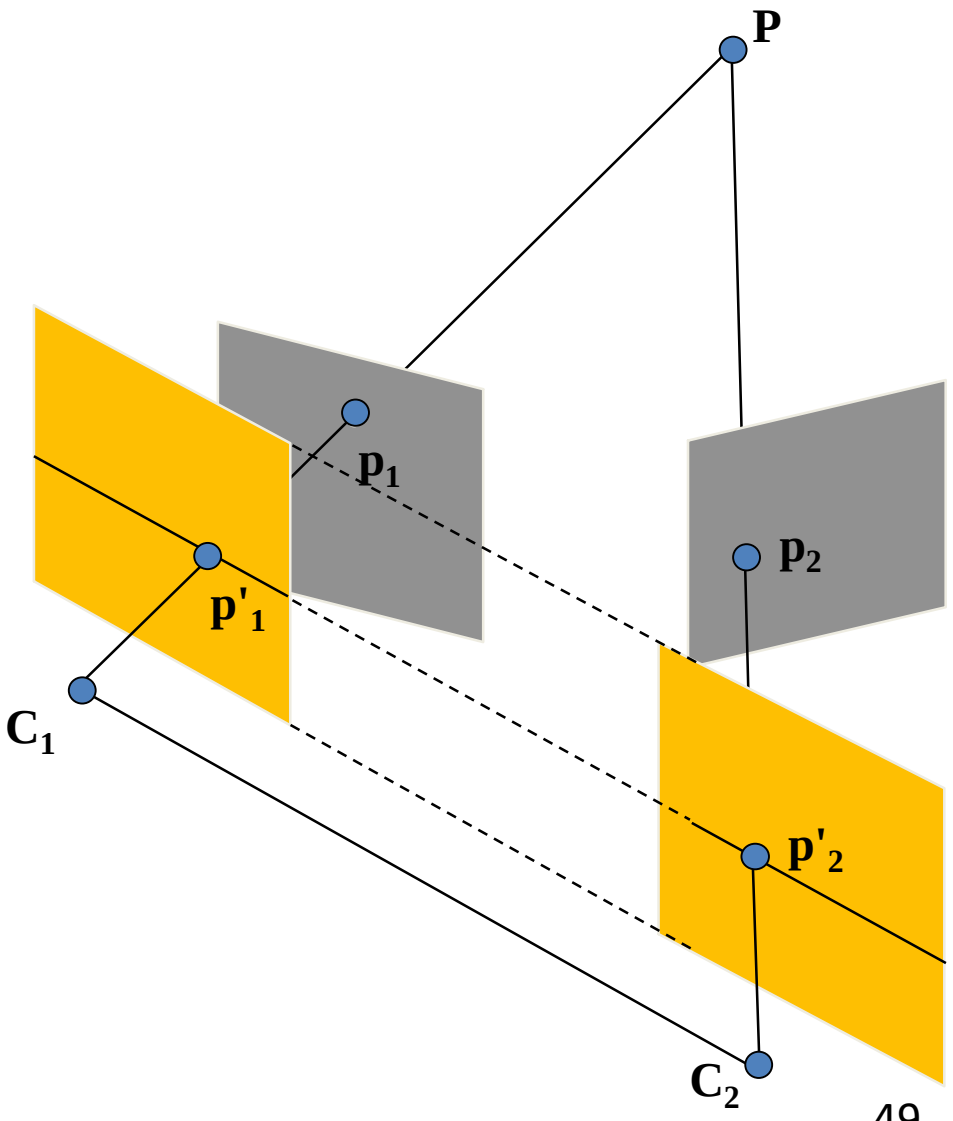
Stereo Rectification

- Warps original image planes onto a coplanar planes 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 (PPMs) obtained by rotating the old ones around their optical centers until focal planes become parallel to each other.
- 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

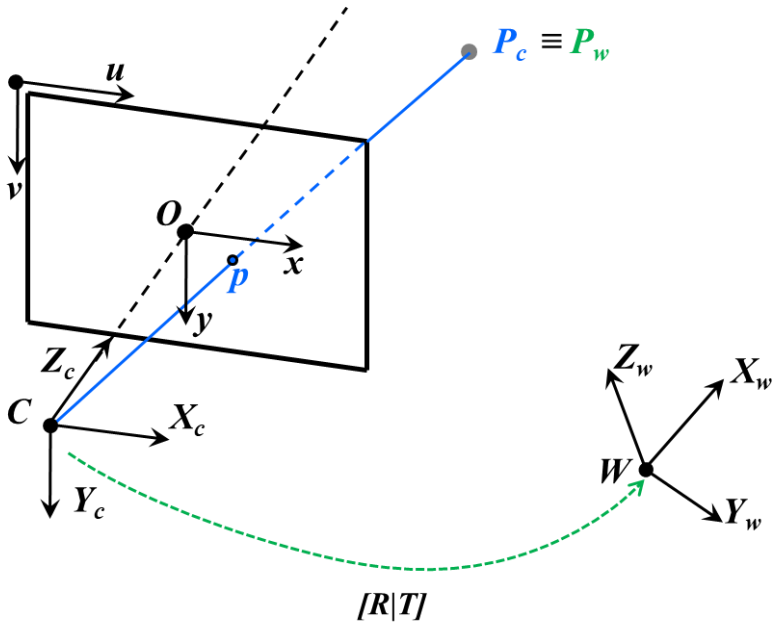


Stereo Rectification (1/5)

In Lecture 02, we have seen that the Perspective Equation for a point P_w in the world frame is

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = KR \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} + T$$

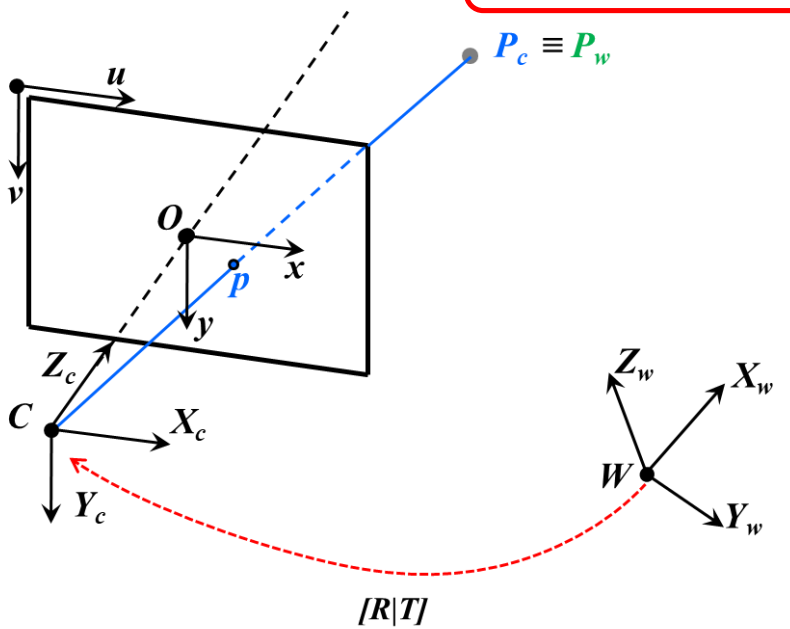
where $R = R_{cw}$ and $T = T_{cw}$ transform points **from the World frame to the Camera frame**.



Stereo Rectification (1/5)

For Stereo Vision, however, it is more common to use $\boldsymbol{R} \equiv \boldsymbol{R_{wc}}$ and $\boldsymbol{T} \equiv \boldsymbol{T_{wc}}$, where now R, and T transform points **from the Camera frame to the World frame**. This is more convenient because $\boldsymbol{T} \equiv \boldsymbol{C}$ directly represents the **world coordinates of the camera center**.
The projection equation can be re-written as:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = KR^{-1} \left(\begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} - T \right) \quad \rightarrow \quad \lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = KR^{-1} \left(\begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} - C \right)$$



Stereo Rectification (2/5)

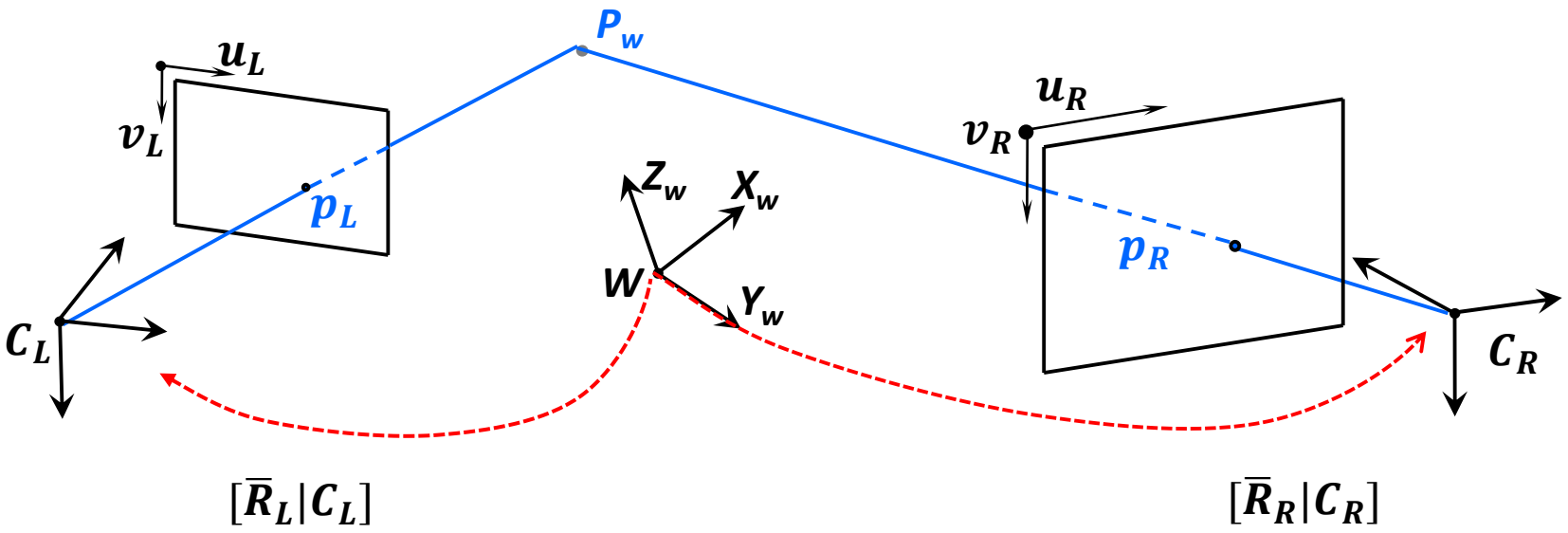
We can now write the Perspective Equation for the Left and Right cameras. For generality, we assume that Left and Right cameras have different intrinsic parameters.

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 **image planes** are **coplanar (i.e., same $\hat{R}$)** and their **intrinsic** parameters are **identical (i.e., same $\hat{K}$)**

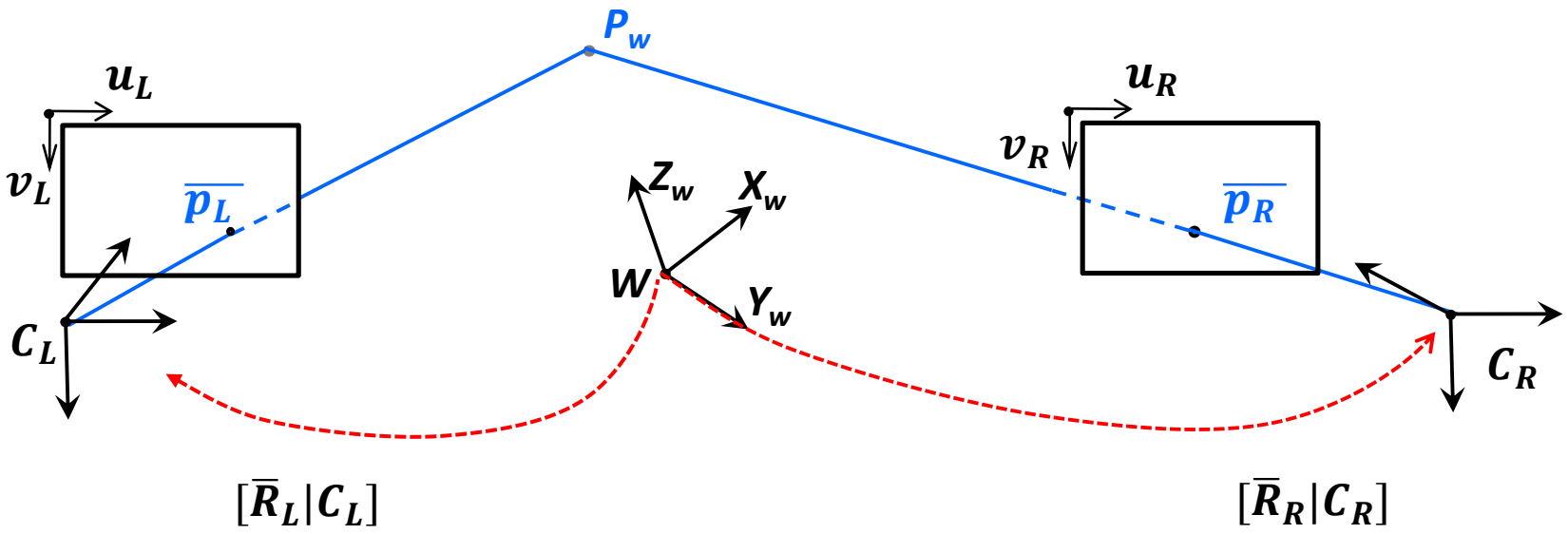
$$\lambda_L \begin{bmatrix} u_L \\ v_L \\ 1 \end{bmatrix} = K_L R_L{}^{-1} \left(\begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} - C_L \right)$$ **Old Left camera**

$$\rightarrow \hat{\lambda}_L \begin{bmatrix} \hat{u}_L \\ \hat{v}_L \\ 1 \end{bmatrix} = \hat{K} \hat{R}^{-1} \left(\begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} - C_L \right)$$ **New Left camera**

$$\lambda_R \begin{bmatrix} u_R \\ v_R \\ 1 \end{bmatrix}$$ **Old Right camera**

$$= K_R R_R{}^{-1} \left(\begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} - C_R \right)$$

$$\rightarrow \hat{\lambda}_R \begin{bmatrix} \hat{u}_R \\ \hat{v}_R \\ 1 \end{bmatrix} = \hat{K} \hat{R}^{-1} \left(\begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} - C_R \right)$$ **New Right camera**

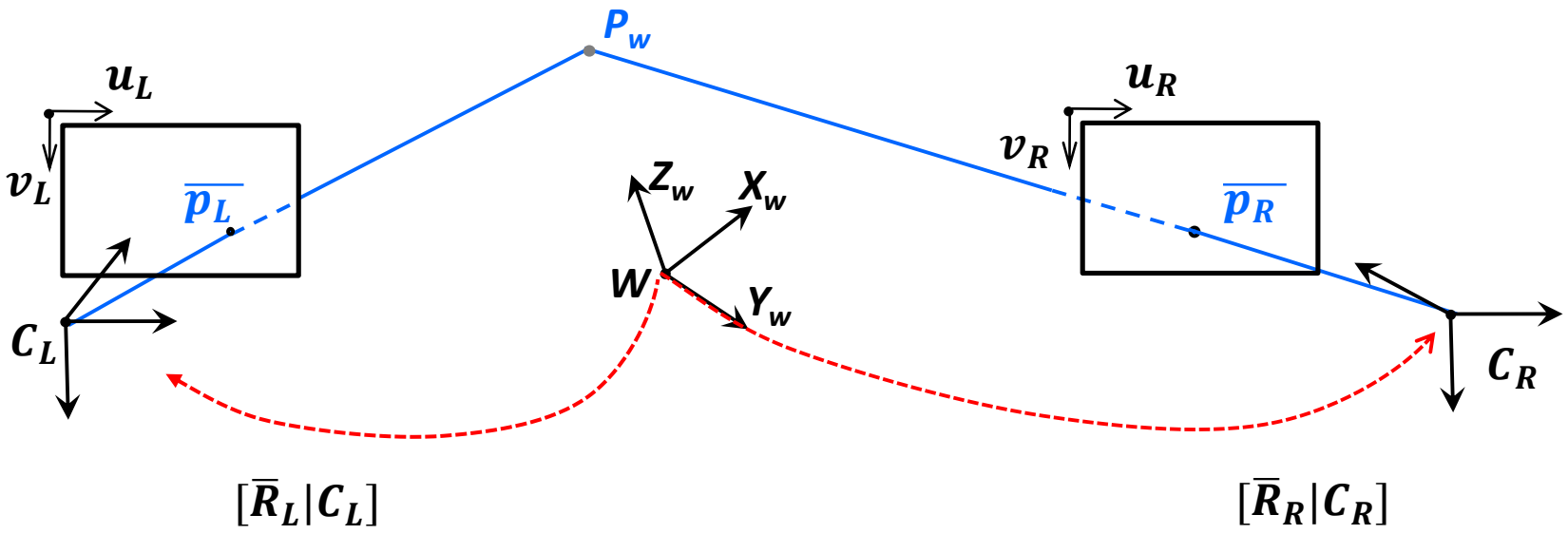


Stereo Rectification (4/5)

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

$$\hat{\lambda}_L \begin{bmatrix} \hat{u}_L \\ \hat{v}_L \\ 1 \end{bmatrix} = \underbrace{\lambda_L \hat{K} \hat{R}^{-1} R_L K_L{}^{-1}}_{\text{Homography of Left Camera}} \begin{bmatrix} u_L \\ v_L \\ 1 \end{bmatrix}$$

$$\hat{\lambda}_R \begin{bmatrix} \hat{u}_R \\ \hat{v}_R \\ 1 \end{bmatrix} = \underbrace{\lambda_R \hat{K} \hat{R}^{-1} R_R K_R{}^{-1}}_{\text{Homography of Right Camera}} \begin{bmatrix} u_R \\ v_R \\ 1 \end{bmatrix}$$



Stereo Rectification (5/5)

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

$$\hat{K} = \frac{K_L + K_R}{2}$$

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

with $\widehat{r_1}, \widehat{r_2}, \widehat{r_3}$ being the column vectors of $\hat{R}$, where:

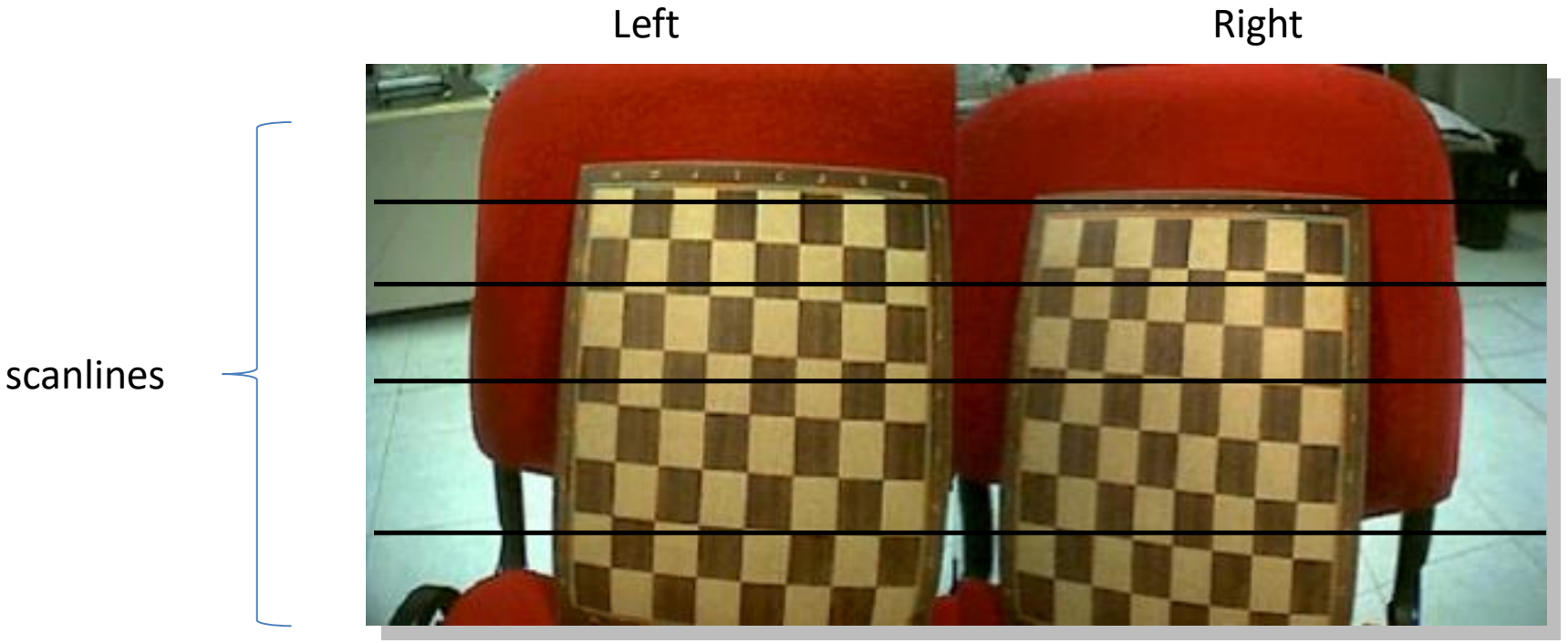
$$\widehat{r_1} = \frac{C_2 - C_1}{\|C_2 - C_1\|}$$

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

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

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

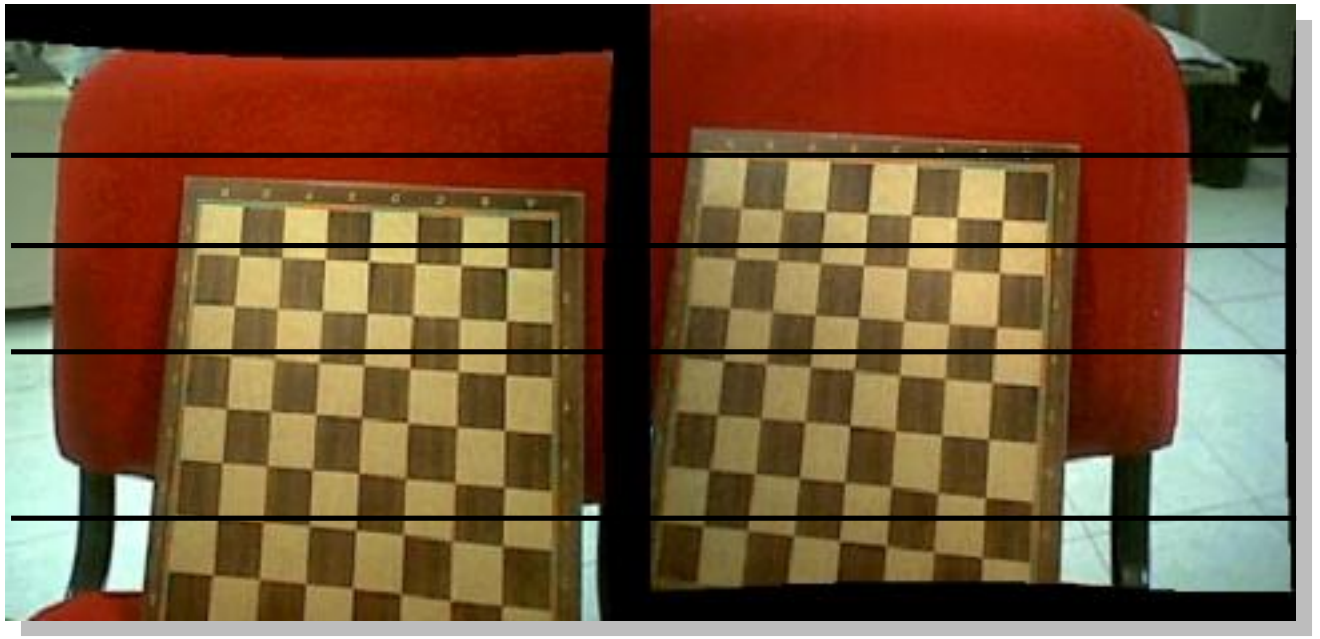
Stereo Rectification: example



Stereo Rectification: example

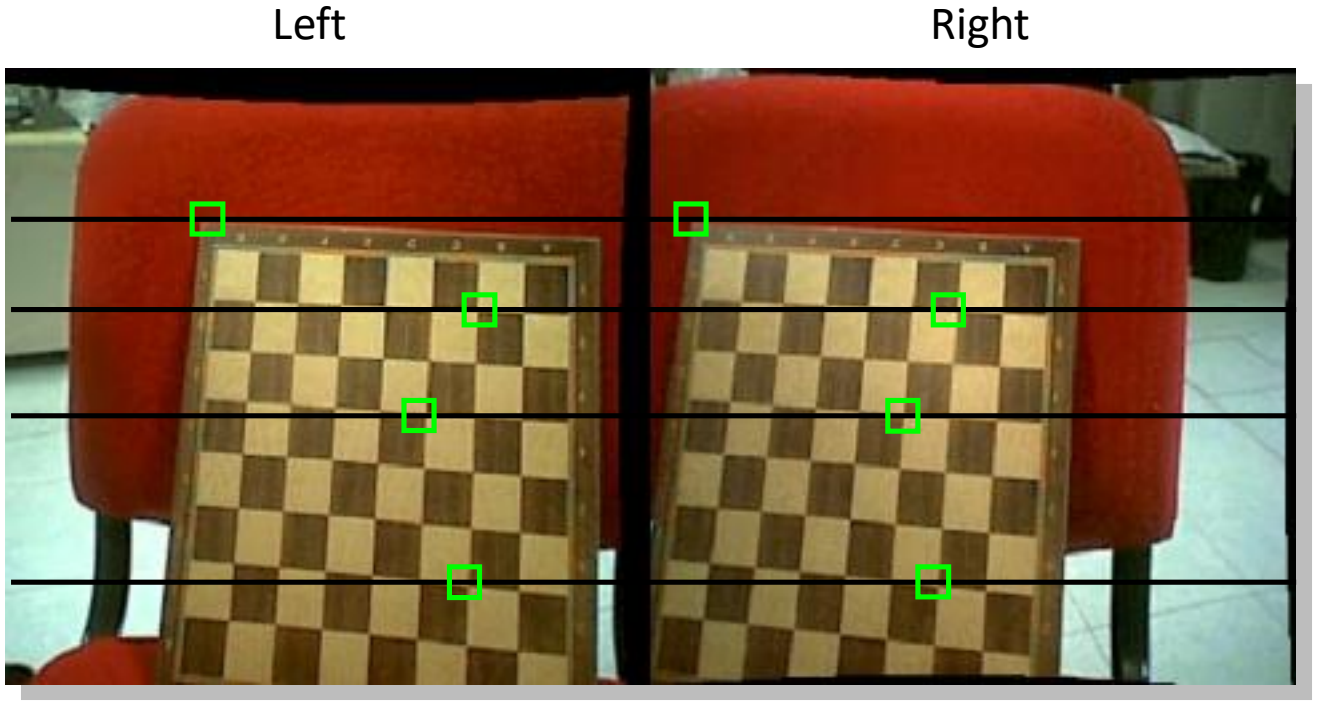
- First, **remove radial distortion** (use **bilinear interpolation** (see lect. 06))

Left Right

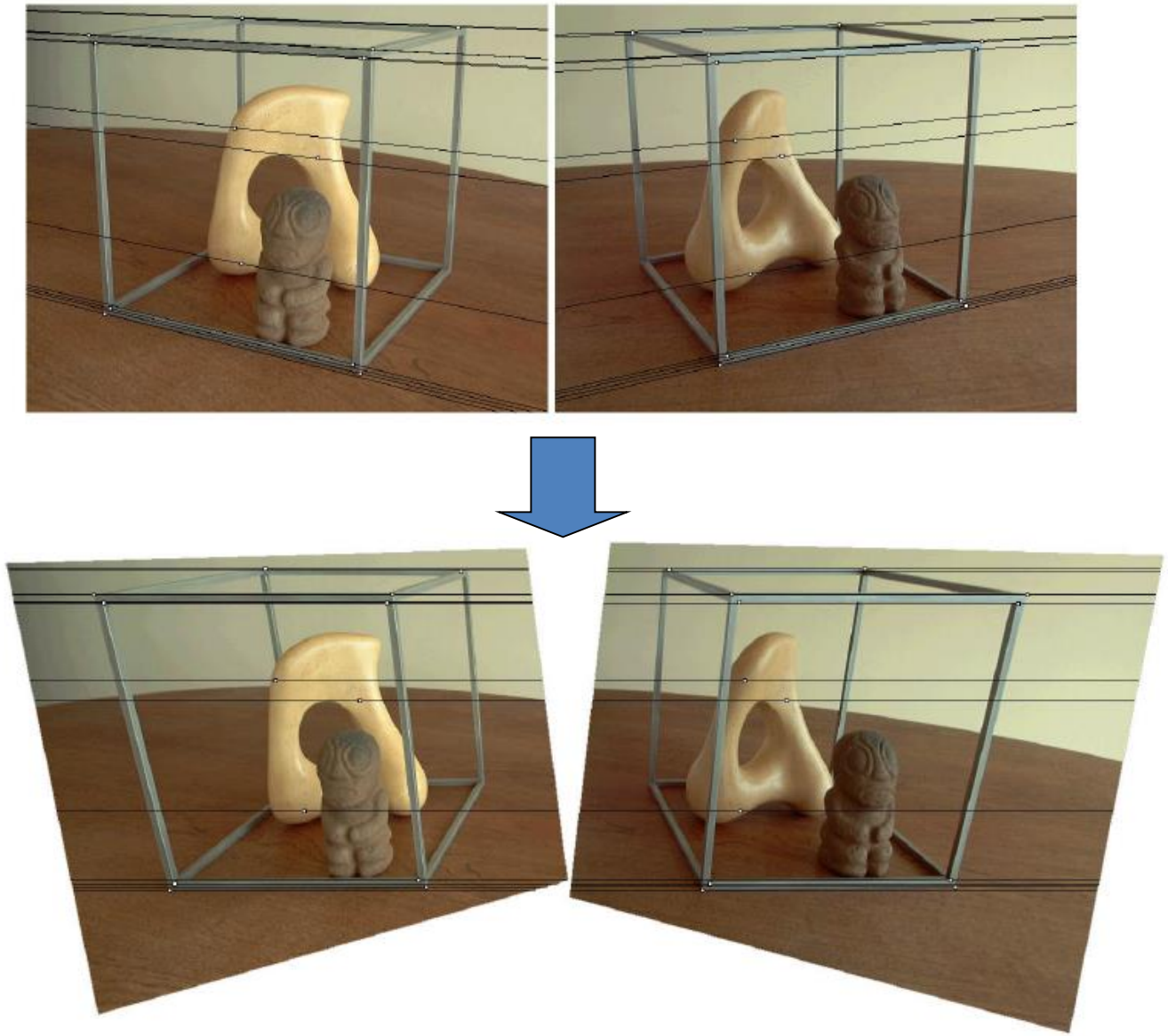


Stereo Rectification: example

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



Stereo Rectification: example



Stereo Vision

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

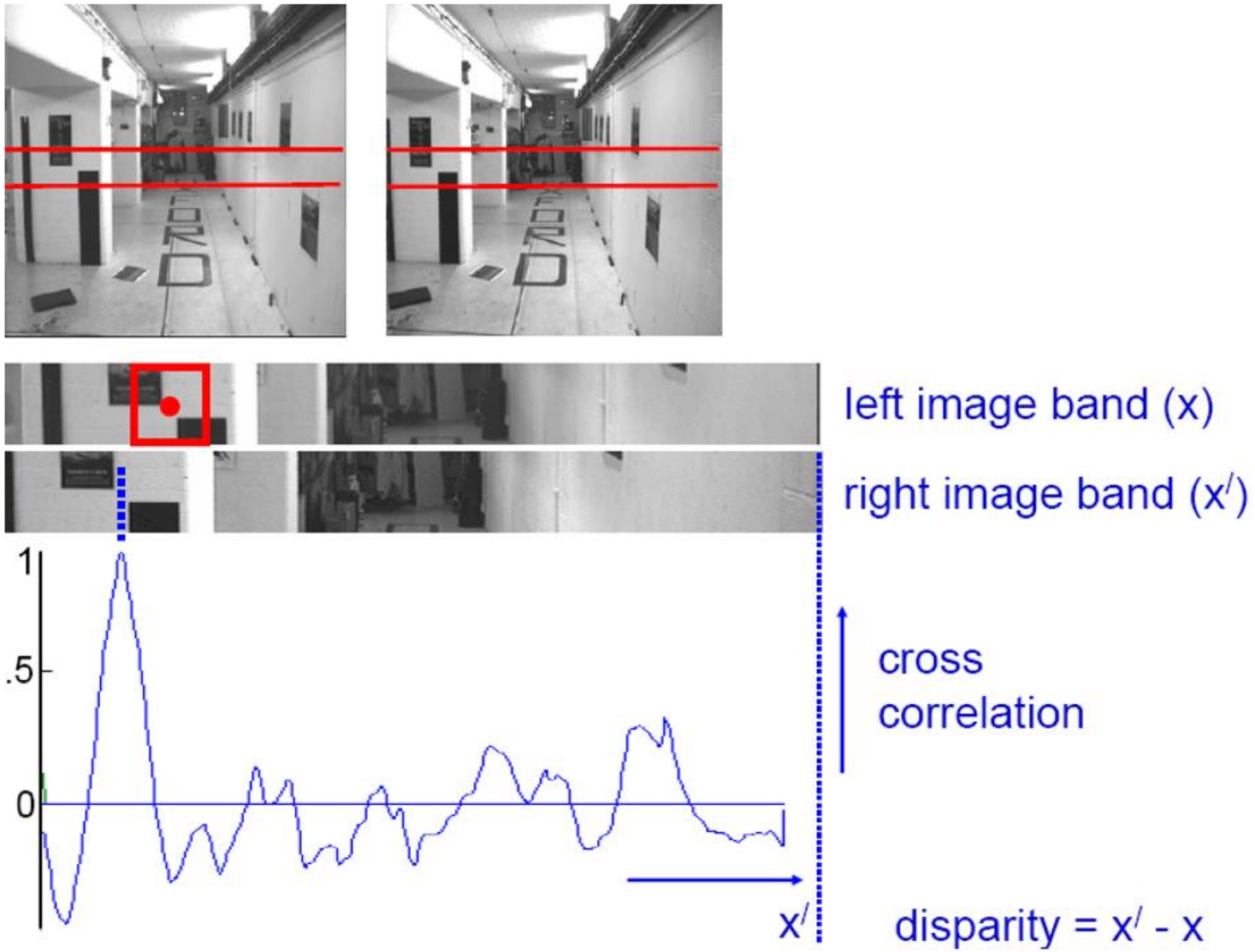
Correspondence problem

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

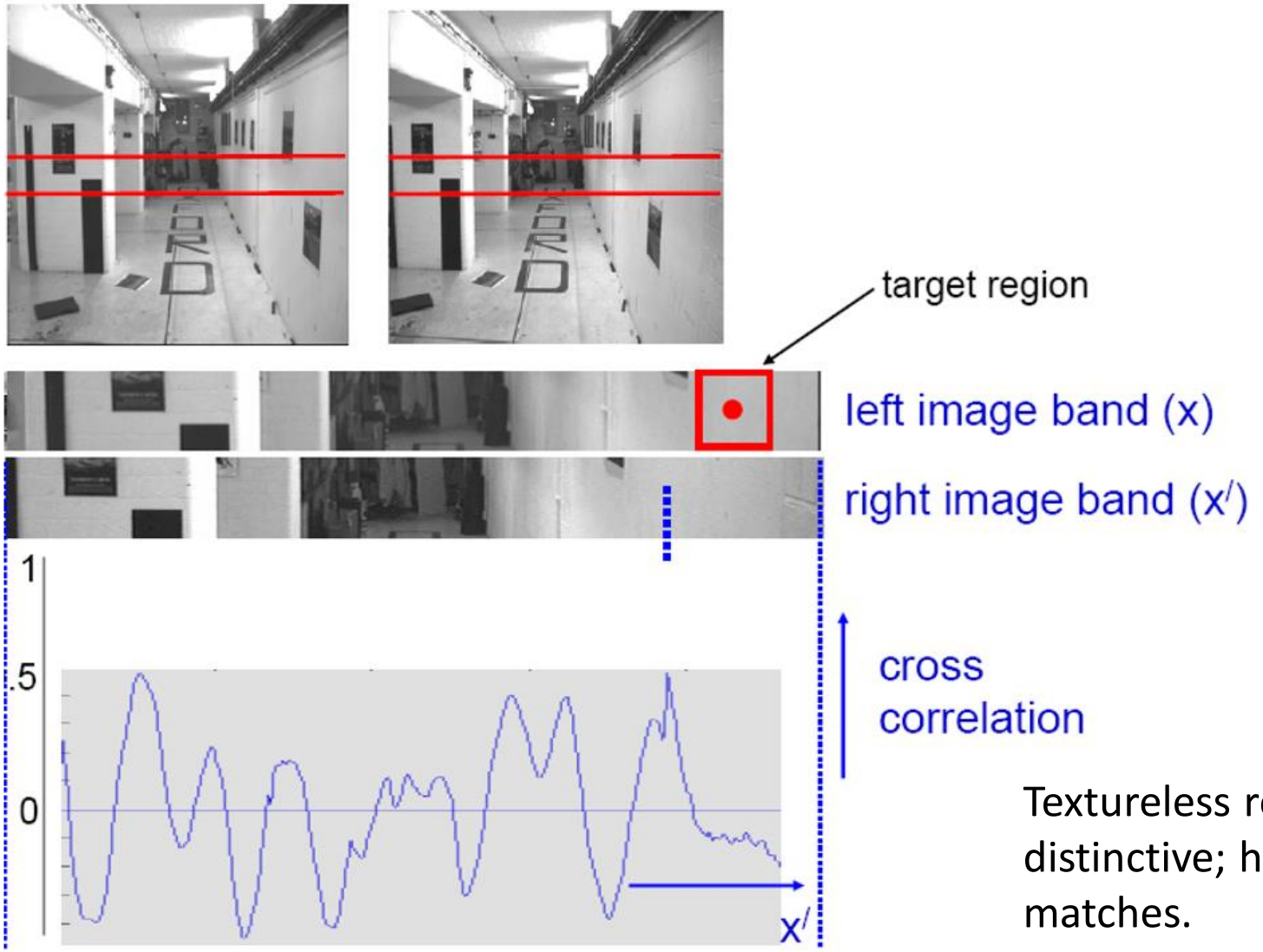
Correspondence problem

- To average noise effects, use a window around the point of interest (assumption: 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

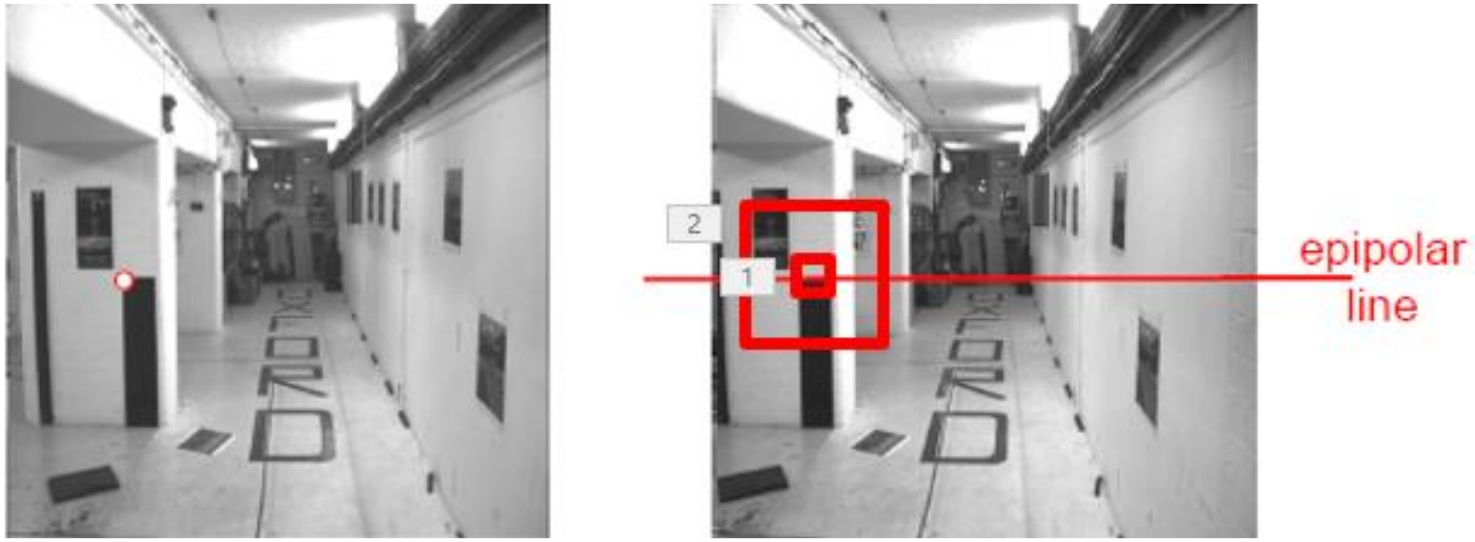


Correspondence Problems:
Textureless regions (**the aperture problem**)



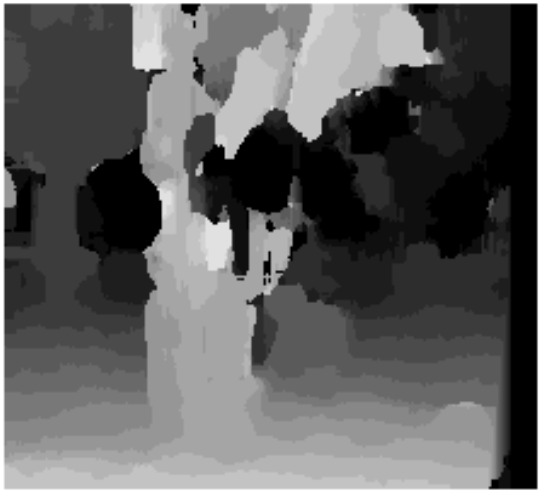
Textureless regions are not distinctive; high ambiguity for matches.

Solution: increase window size



Effects of window size

W = 3



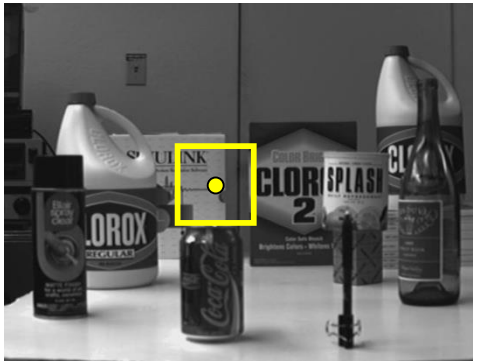
W = 20

- Smaller window
  - \+ More detail
  - – More noise
- Larger window
  - \+ Smoother disparity maps
  - – Less detail

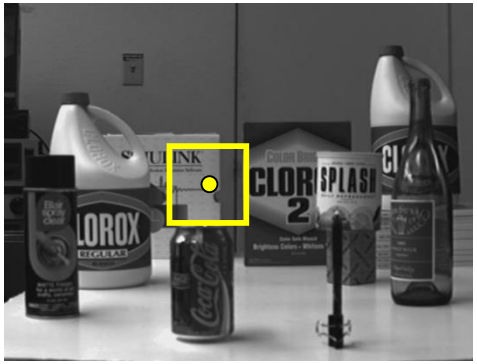
Disparity map

Input to dense 3D reconstruction

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



Left image

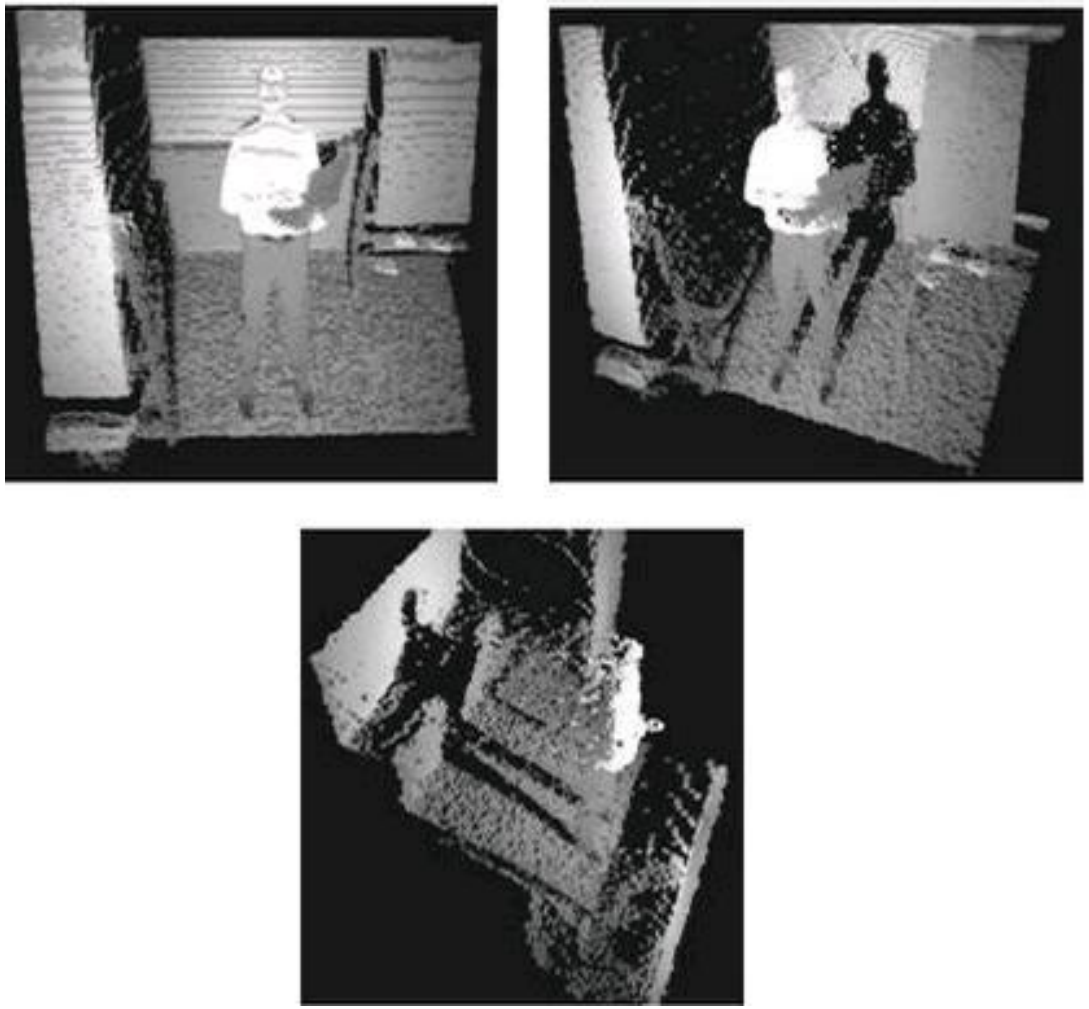


Right image

Close objects experience bigger disparity
⇨ appear brighter in disparity map

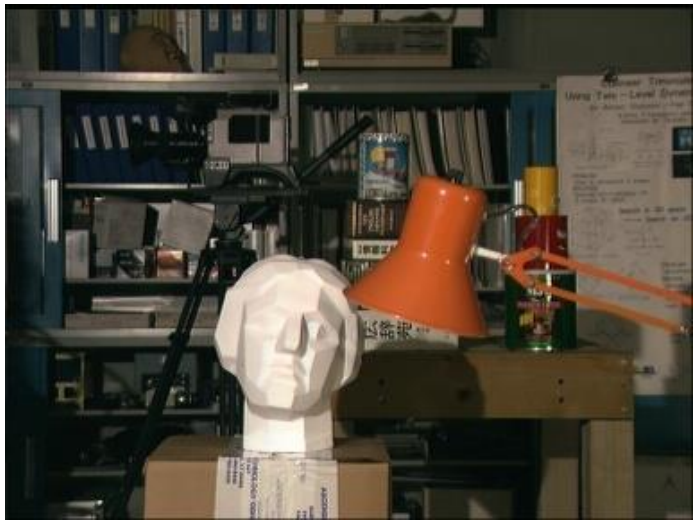
From Disparity Maps to Point Cloud

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

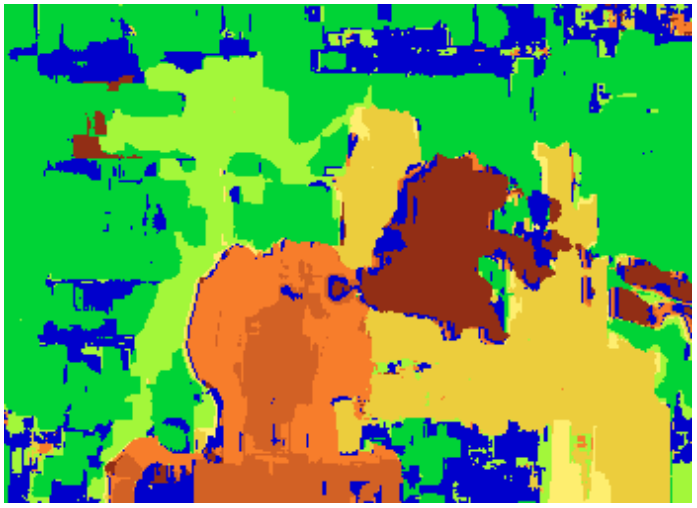


Accuracy

Data



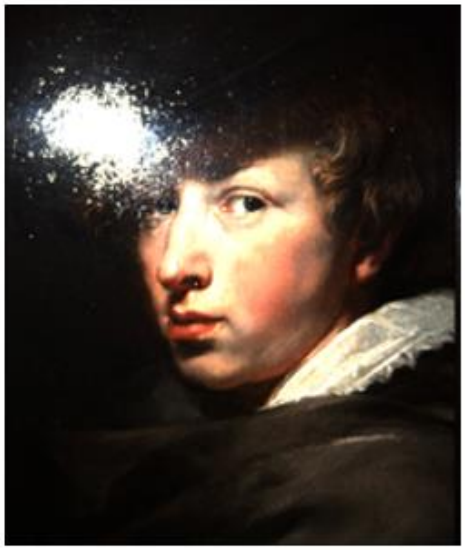
Window-based matching



Ground truth

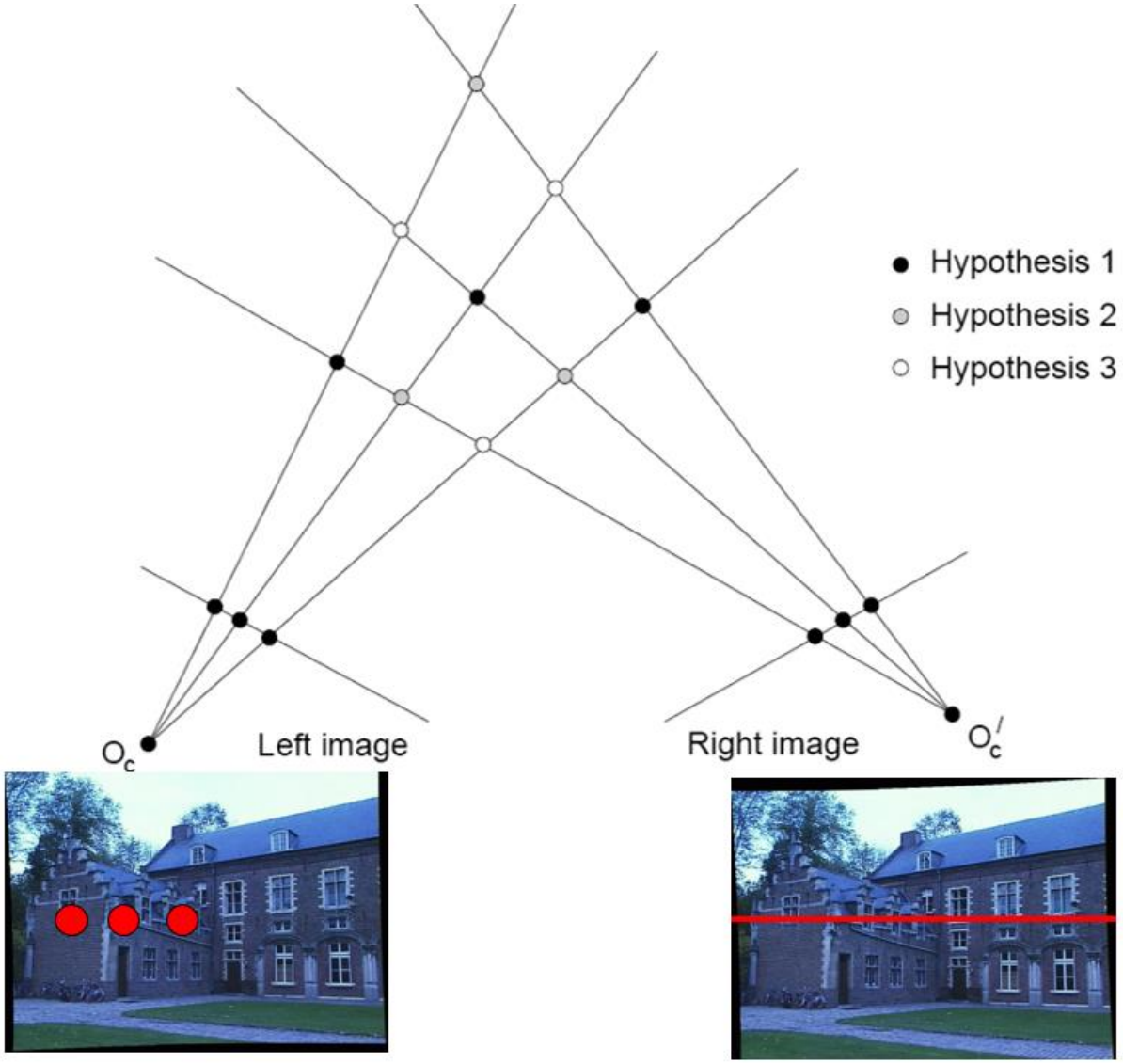
Challenges

Occlusions and repetitive patterns



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...

Using Deep Learning

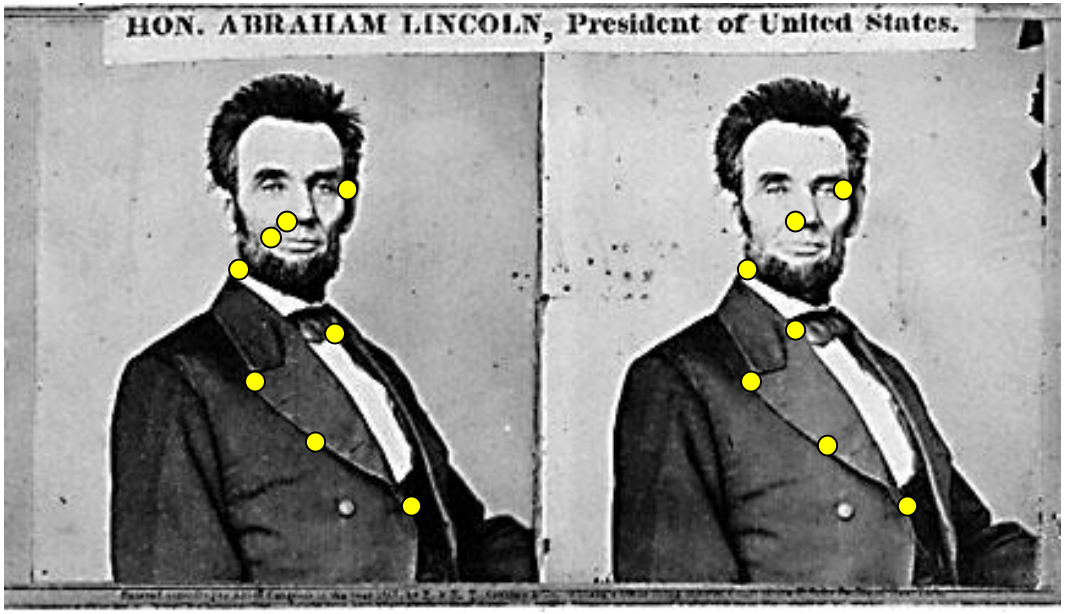
Ground truth

Jia-Ren Chang Yong-Sheng Chen, Pyramid Stereo Matching Network, CVPR'18

For the latest and greatest:
http://vision.middlebury.edu/stereo/ and
http://www.cvlibs.net/datasets/kitti/eval_scene_flow.php?benchmark=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

Understanding Check

Are you able to answer the following questions?

- Can you relate Structure from Motion to 3D reconstruction? In what they differ?
- Can you define disparity in both the simplified and the general case?
- Can you provide a mathematical expression of depth as a function of the baseline, the disparity and the focal length?
- Can you apply error propagation to derive an expression for depth uncertainty? How can we improve the uncertainty?
- Can you analyze the effects of a large/small baseline?
- What is the closest depth that a stereo camera can measure?
- Are you able to show mathematically how to compute the intersection of two lines (linearly and non-linearly)?
- What is the geometric interpretation of the linear and non-linear approaches and what error do they minimize?
- Are you able to provide a definition of epipole, epipolar line and epipolar plane?
- Are you able to draw the epipolar lines for two converging cameras, for a forward motion situation, and for a side-moving camera?
- Are you able to define stereo rectification and to derive mathematically the rectifying homographies?
- How is the disparity map computed?
- How can one establish stereo correspondences with subpixel accuracy?
- Describe one or more simple ways to reject outliers in stereo correspondences.
- Is stereo vision the only way of estimating depth information? If not, are you able to list alternative options?