DTU

Daniel Olesen, DTU Space

30540 – Photogrammetry (3)

Todays Lecture

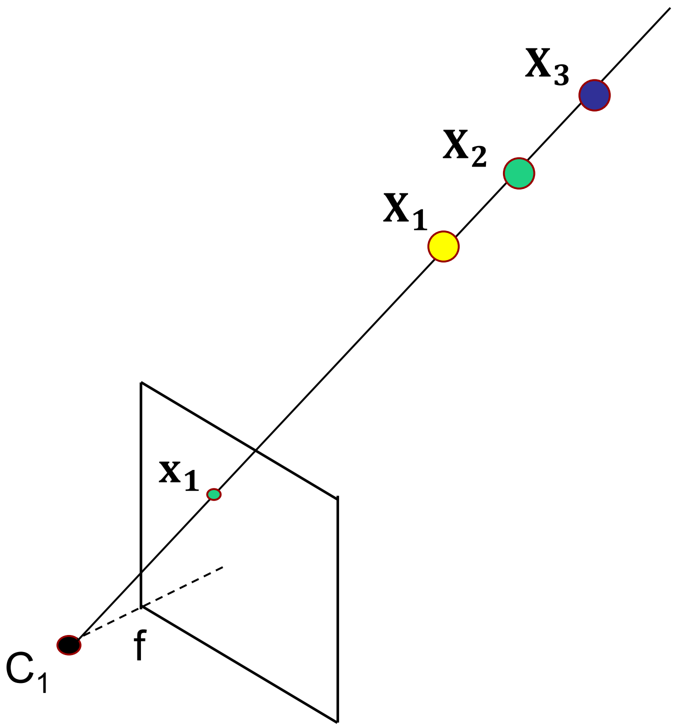
- How can we perceive in 3D with cameras?

- Epipolar Geometry
  - Relative and Absolute Orientation
  - Essential and Fundamental Matrix
  - 8-point algorithm

- Triangulation methods
  - Middle-point algorithm
  - Linear Triangulation

- Stereo-calibration and rectification
  - Triangulation by Disparity

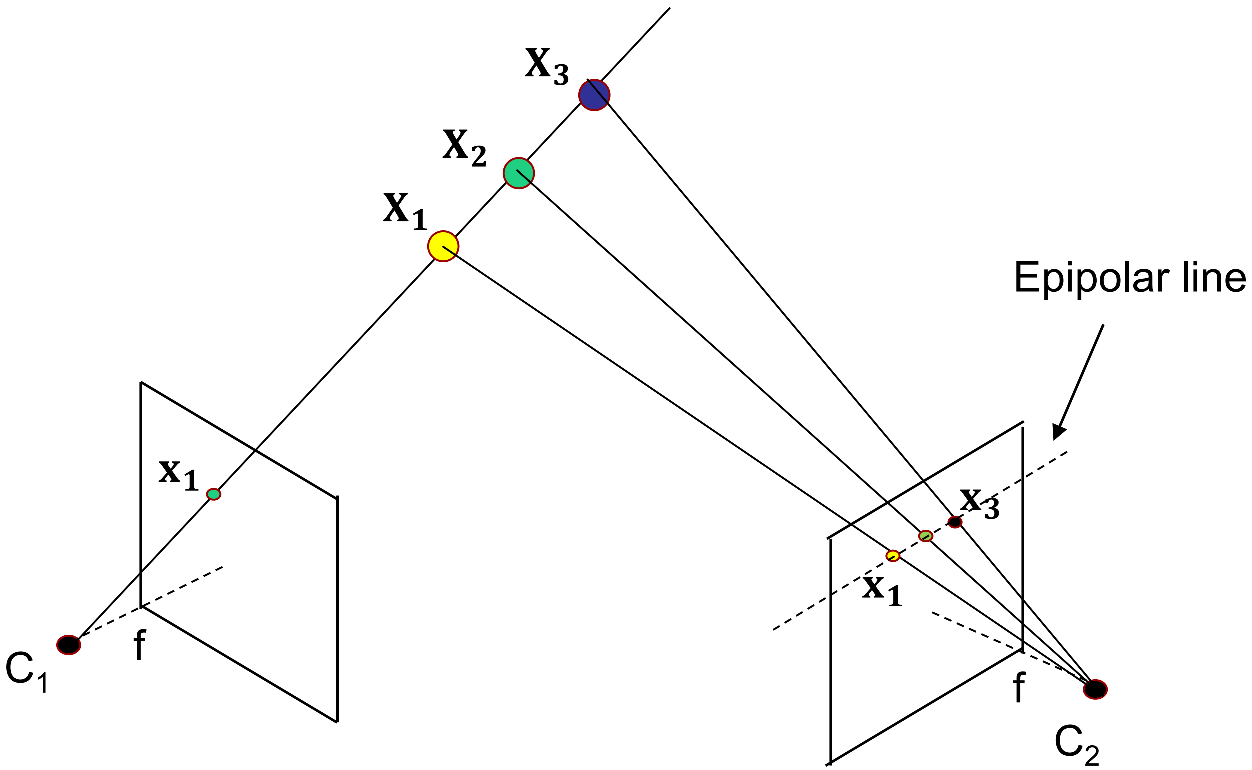
Can we reconstruct 3D points from a single camera?



If the position and orientation of a camera is known, we are from an image point able to construct a line in which the corresponding world-point must reside.

In other words, from a single view we can not distinguish between the collinear points X1,X2,X3.

What if we use two cameras?

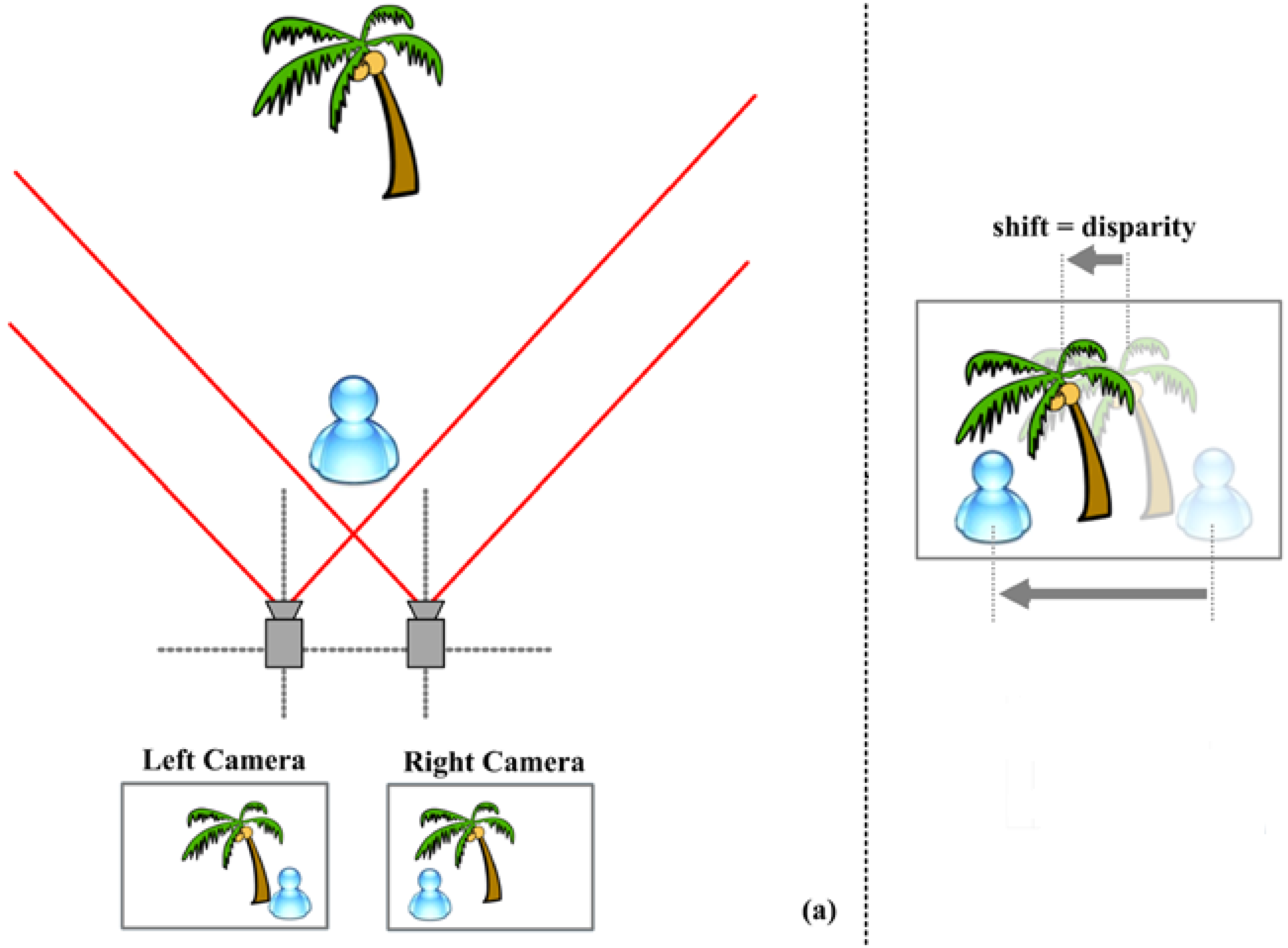


In case we have a second camera with known position and orientation (pose), we can from the second view distinguish between X1, X2 and X3

X1, X2 and X3 all lies on what is known as a epipolar line in the imageplane in the second view. We will later on establish a geometrical constraint between this line and the point in the left view.

Using two cameras to view the same scene is called stereo vision

Perceiving depth from Disparity (parallel view)

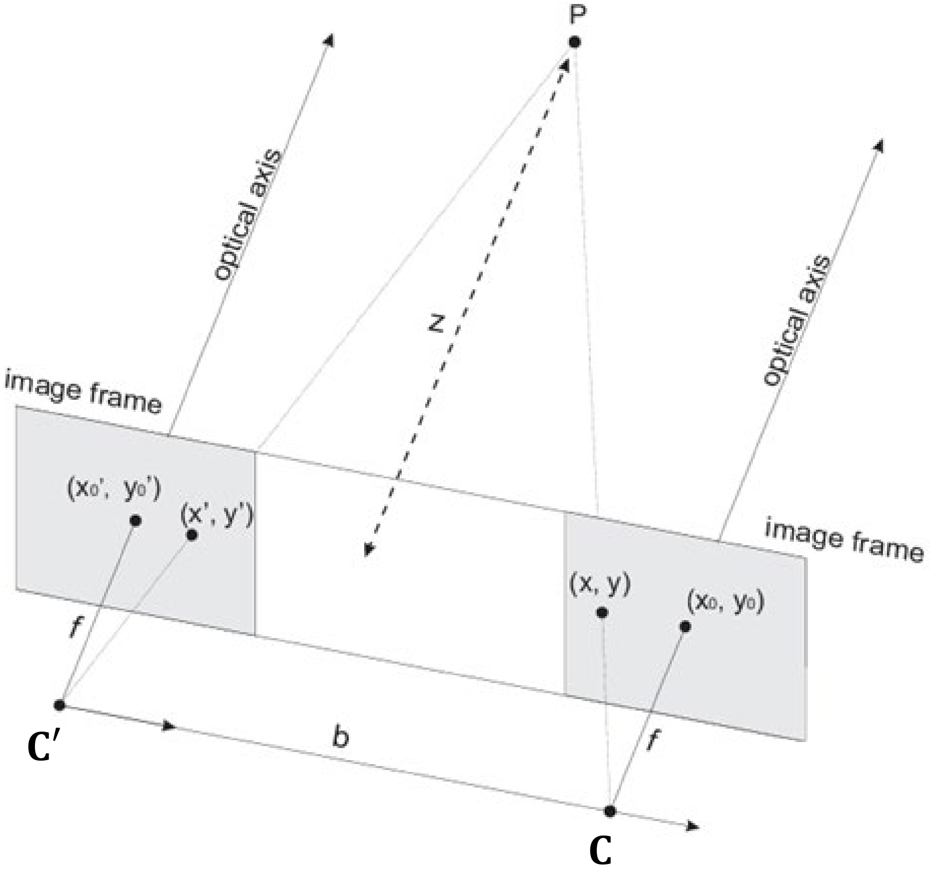


https://medium.com/mini-distill/pps-efficient-deep-learning-for-stereo-matching-de253fc411d4

Perceiving depth from Disparity (parallel view)

Demo with Intel Realsense D455

Recover depth from a parallel view



By using two parallel cameras with known absolute orientation, it is fairly easy to recover 3D coordinates as the depth is inversely proportional to the disparity.

It is generally possible to reconstruct 3D points from two arbitrary viewpoints. This process is known as triangulation.

Source: Anderson A. S. Souza, Rosiery Maia and Luiz M. G. Gonçalves: 3D Probabilistic Occupancy Grid to Robotic Mapping with Stereo Vision

Todays Lecture

✓How can we perceive in 3D with cameras?

- Epipolar Geometry
  - Relative and Absolute Orientation
  - Essential and Fundamental Matrix
  - 8-point algorithm

- Triangulation methods
  - Mid-point method algorithm
  - Linear Triangulation

- Stereo-calibration and rectification
  - Triangulation by Disparity

Stereo Orientation

- In order to obtain 3D world measurements from stereo images, 12 parameters has to be determined for calibrated cameras (intrinsics known) and 22 for uncalibrated cameras.

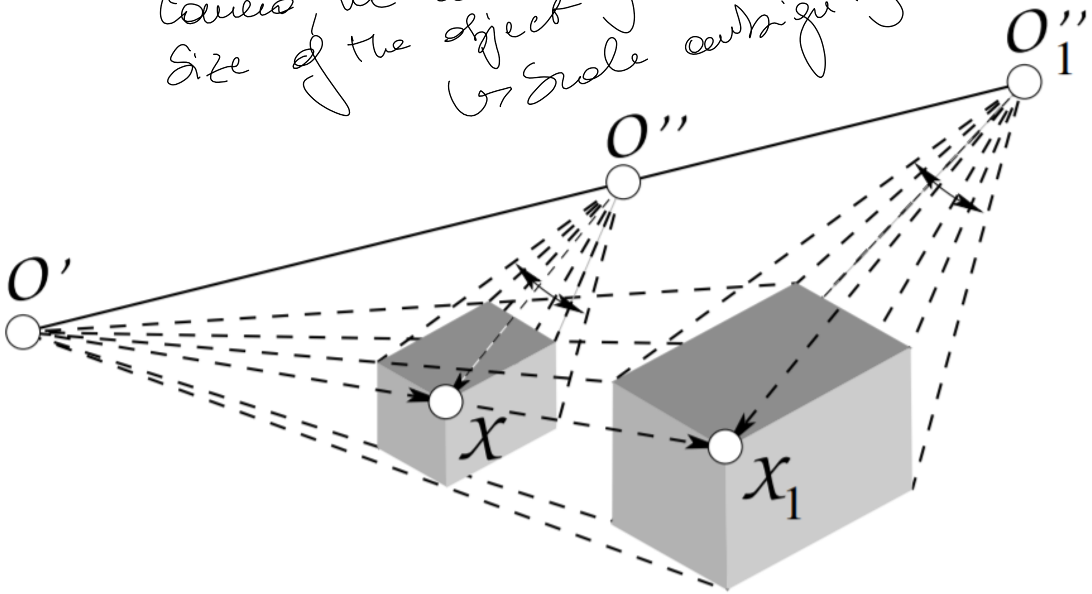
- These are for the calibrated case:
  - Rotation + translation of first view (6 params)
  - Rotation + translation of second view (6 params)

- How many parameters do we need to determine if we are using the same camera from two viewpoints?

Relative Orientation (Scale ambiguity)

- Suppose we only have two views of a scene, what would be able to recover from this alone?

If we don't have the positions of the camera, we can't say much on the size of the object
↳ scale ambiguity



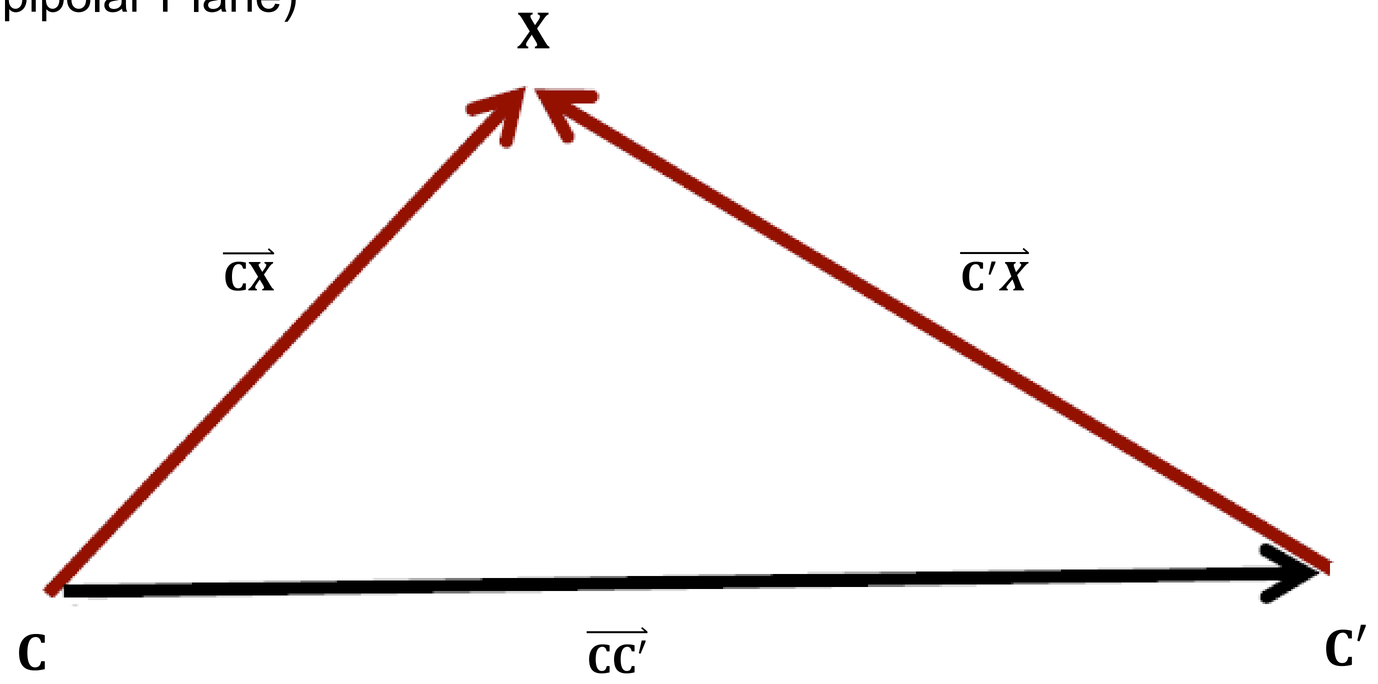
Given enough point-correspondes between the two views we can recover the geometry of the photographed object up to a similarity transform (ambigous scale)

Why?

The image formed from the big box and O_1'' is the same as for the small box and O''

Epipolar Geometry (RO)

- The Fundamental geometric relationship between a world-point captured by a stereo-view and the the perspective centres of the camera is that they together form a common plane (Epipolar Plane)



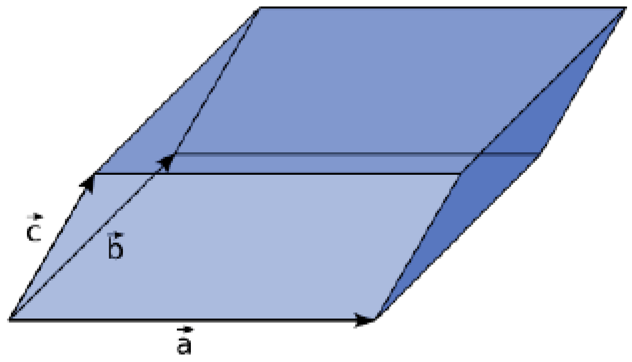
Epipolar Geometry (RO)

→ Computing the volume, but since we are in a plane: Volume is zero

- Mathematically, we can formulate the relationship between the three vectors as:

$$\overrightarrow{\mathbf{CX}} \cdot \left(\overrightarrow{\mathbf{CC'}} \times \overrightarrow{\mathbf{C'X}}\right) = \mathbf{0}$$

- Recall here, that the triple-product of vectors {a,b and c} define the volume of the parallellepiped



- If the product is zero, the three vectors must reside in a <u>common plane!</u>

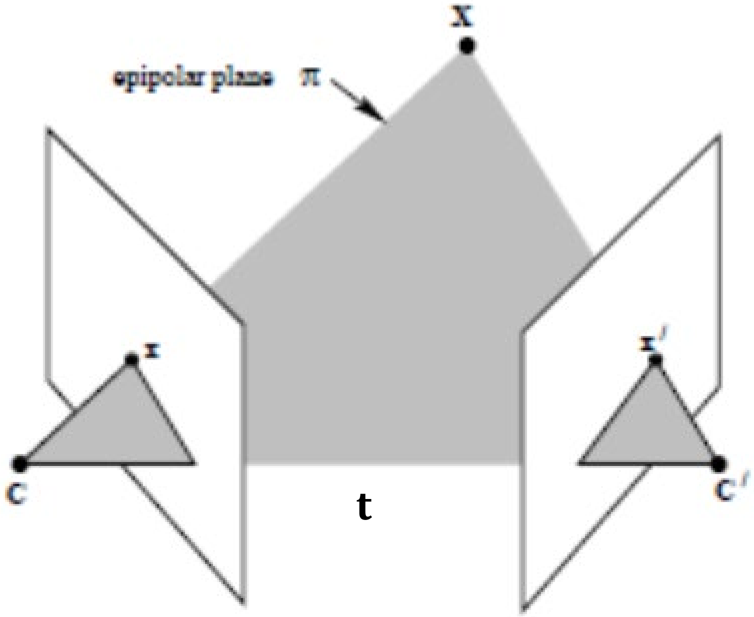
Essential Matrix : Purpose is to decode cameras positions

- For simplicity, we assume that the left camera centre is the origin of the world coordinate system.

- If we want to express, $\mathbf{X}$, in the coordinate frame of the second camera, we have:

$$\mathbf{X}' = \mathbf{R}\mathbf{X} + \mathbf{t}$$

- In order to rearrange the equation to prove the coplanarity constraint, we first premultiply (cross-product) with, $\mathbf{t}$,



$$\mathbf{t} \times \mathbf{X}' = \mathbf{t} \times (\mathbf{R}\mathbf{X} + \mathbf{t}) \Rightarrow \mathbf{t} \times \mathbf{X}' = \mathbf{t} \times \mathbf{R}\mathbf{X} + \underbrace{\mathbf{t} \times \mathbf{t}}_{0} \Rightarrow \mathbf{t} \times \mathbf{X}' = \mathbf{t} \times \mathbf{R}\mathbf{X}$$

(cont…)

Essential Matrix

- Hereafter, we premultiply (dot-product) with, $\mathbf{X}'^{T}$,

$$\mathbf{X}'^{T} \cdot (\mathbf{t} \times \mathbf{X}') = \mathbf{X}'^{T} \cdot (\mathbf{t} \times \mathbf{R}\mathbf{X}) \Rightarrow$$
$$\mathbf{0} = \mathbf{X}'^{T} \cdot (\mathbf{t} \times \mathbf{R}\mathbf{X})$$

We can express, $(\mathbf{t} \times \mathbf{R}\mathbf{X})$ as

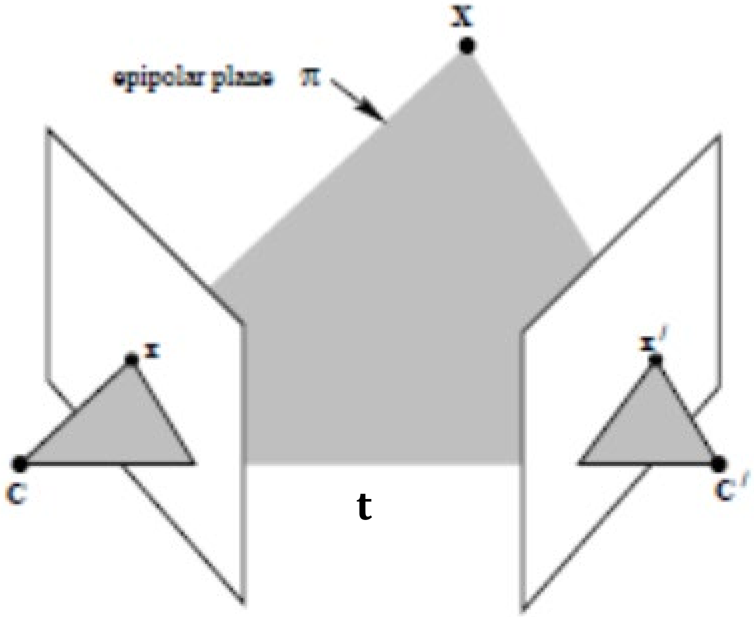
$$(\mathbf{t} \times \mathbf{R}\mathbf{X}) = \mathbf{t}_{\times}\mathbf{R}\mathbf{X}$$

where, $\mathbf{t}_{\times}$, is the skew-symmetric form of $\mathbf{t}$ and define, $\mathbf{E} = \mathbf{t}_{\times}\mathbf{R}$, to obtain

$$\mathbf{0} = \mathbf{X}'^{T} \cdot \mathbf{E} \cdot \boldsymbol{X}$$

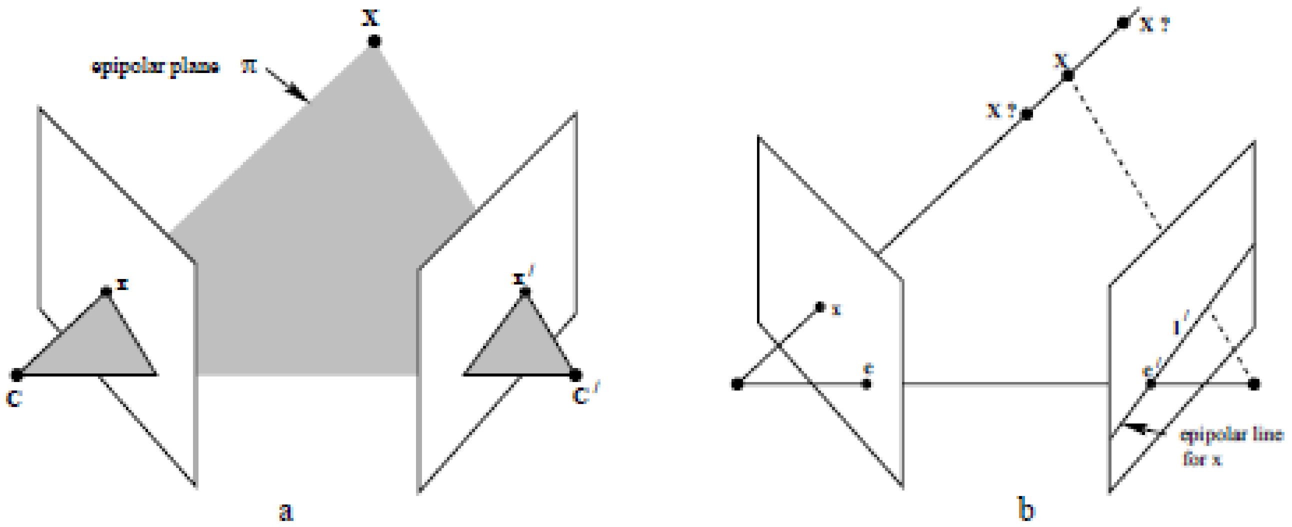
Which furthermore can be expressed from the image coordinates

$$\mathbf{0} = \mathbf{x}'^{T} \cdot \mathbf{E} \cdot \mathbf{x}$$



$$\mathbf{t}_{\times} = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}$$

Epipolar lines



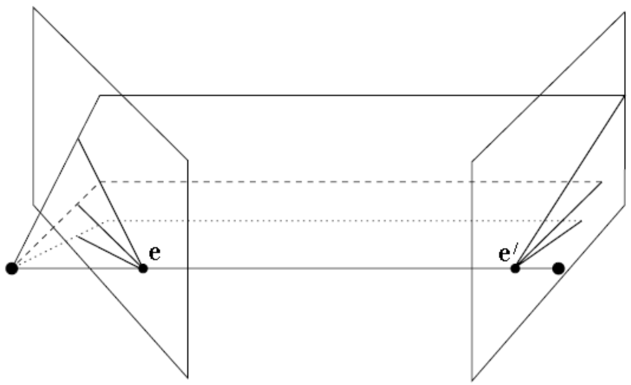
If the Essential Matrix is known, it can restrict search of a point correspondence to one line in the second image.

Epipolar lines:

$$l' = E^T x$$

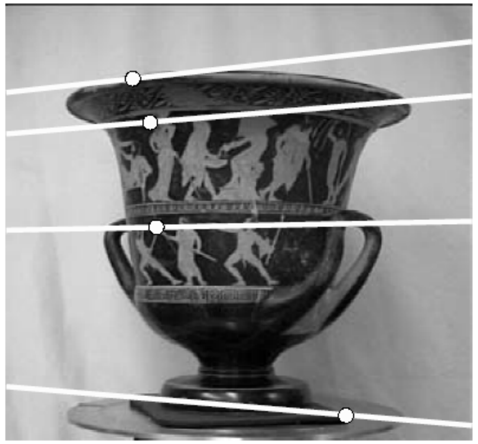
$$l = Ex'$$

Epipolar planes



a

b



c

Essential Matrix

- The essential matrix ”encodes” the relative translation and rotation between 2 camera poses – but only up to a scale ambiguity
- Is usually solved by point-correspondences in both views
  - Longuet-Higgins (1981): 8-point algorithm
  - Nistér (2003): 5-point algorithm
- Applies only to the calibrated case where intrinsics are known!
  - A similar relationshop for uncalibrated cameras is known as the **Fundamental** Matrix.

Fundamental Matrix

- The Fundamental matrix are similar to the essential matrix but also encodes the intrinsic parameteres of each camera

$$\mathbf{0} = \mathbf{x}'^{T} \cdot \mathbf{F} \cdot \mathbf{x}$$

where

$$\mathbf{F} = \mathbf{K}'^{-\mathbf{T}} \mathbf{E} \mathbf{K}^{-1}$$

Estimating the Fundamental Matrix (8-point algorithm)

Point 2nd image

Point 1st image

$$\mathbf{x}_2^T \mathbf{F} \mathbf{x}_1 = 0$$

$$[x_2, y_2, 1] \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = 0 \Rightarrow$$

$$\begin{bmatrix} x_1x_2 & x_1y_2 & x_1 & y_1x_2 & y_1y_2 & y_1 & x_2 & y_2 & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{21} \\ f_{31} \\ f_{12} \\ f_{22} \\ f_{32} \\ f_{13} \\ f_{23} \end{bmatrix} = 0$$

$$\mathbf{A}\mathbf{x} = 0$$

where $\mathbf{x} = [f_{11}, f_{21}, f_{31}, f_{12}, f_{22}, f_{32}, f_{13}, f_{23}, f_{33}]^T$. By solving $\mathbf{x}$ using SVD, the fundamental matrix can be estimated.

```
MATLAB:[~,~,V] = svd(A);
x = V(:,end);
```

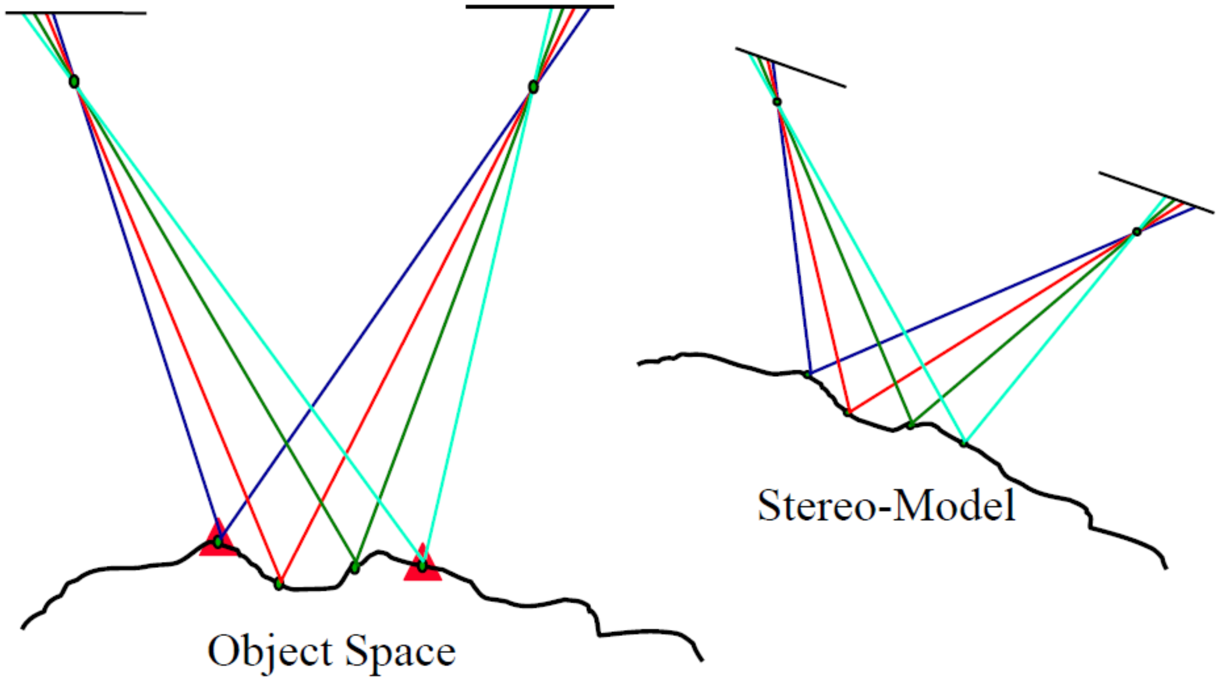
Exterior Orientation of Stereo-pair

- Fully described by 12 parameters
  - $X_l, Y_l, Z_l, \omega_l, \phi_l, \kappa_l$
  - $X_r, Y_r, Z_r, \omega_r, \phi_r, \kappa_r$

- Can be decomposed into
  - Relative Orientation (5 params)
    - 2 translations + 3 rotations
  - Absolute Orientation (7 params)
    - 3 rotations + 3 translations + 1 scale-factor

Absolute Orientation

If we know the camera coordinates relative to the world then we are able to fully reconstruct



The RO model uses the camera-coordinate system of the first camera as reference

In order to relate the model to the physical world, the exact location of this must be determined in object-space (world) coordinates and the scale factor must be resolved!

Recover extrinsics from the Camera-projection matrix

- Recall from the last lecture, that the camera projection matrix,

$$\mathbf{P} = \mathbf{KR}[\mathbf{I} \quad -\tilde{\mathbf{C}}] = \mathbf{K}[\mathbf{R} \quad \mathbf{t}], \qquad \boldsymbol{where}\ \mathbf{t} = -\mathbf{R}\tilde{\mathbf{C}}$$

- If we denote $\mathbf{M} = \mathbf{KR}$, i.e. $\mathbf{M} = \mathbf{P}(1{:}3, 1{:}3)$ we can express

$$\mathbf{P} = \mathbf{M}[\mathbf{I} \quad \mathbf{M}^{-1}\mathbf{p}_4] = \mathbf{KR}[\mathbf{I} \quad -\tilde{\mathbf{C}}]$$

4th column of P

From this, given $\mathbf{P}$, we could easily obtain the camera-centre (in world coordinates) as:

$$\tilde{\mathbf{C}} = -\mathbf{M}^{-1}\mathbf{p}_4$$

See also page 157 i Hartley & Zisserman (chap 6)

Todays Lecture

✓ How can we perceive in 3D with cameras?

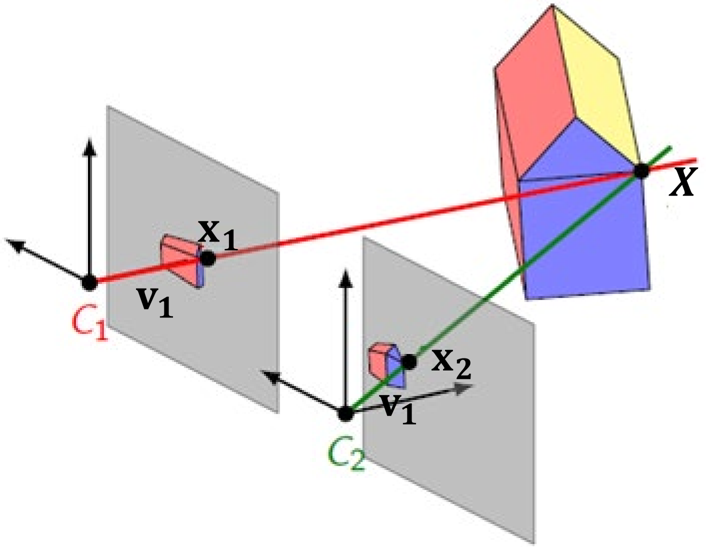
✓ Epipolar Geometry
  ✓ Relative and Absolute Orientation
  ✓ Essential and Fundamental Matrix
  ✓ 8-point algorithm

- Triangulation methods
  – Mid-point method algorithm
  – Linear Triangulation

- Stereo-calibration and rectification
  – Triangulation by Disparity

Triangulation (midpoint algorithm)

- If the intrinsics and extrinsics of both cameras are known, an object point can be found from the intersection point of the rays from each camera!



$$\mathbf{X} = \mathbf{C_1} + (\mathbf{x_1} - \mathbf{C_1})\alpha_1 = \mathbf{C_1} + \mathbf{v_1}\alpha_1$$

$$\mathbf{X} = \mathbf{C_2} + (\mathbf{x_2} - \mathbf{C_2})\alpha_2 = \mathbf{C_2} + \mathbf{v_2}\alpha_2$$

$$\downarrow$$

$$\begin{bmatrix} \mathbf{I_3} & -\mathbf{v_1} & \mathbf{0} \\ \mathbf{I_3} & \mathbf{0} & -\mathbf{v_2} \end{bmatrix} \begin{bmatrix} \mathbf{X} \\ \alpha_1 \\ \alpha_2 \end{bmatrix} = \begin{bmatrix} \mathbf{C_1} \\ \mathbf{C_2} \end{bmatrix} \longrightarrow \begin{bmatrix} \mathbf{X} \\ \alpha_1 \\ \alpha_2 \end{bmatrix} \text{ is solvable from L.S.}$$

Note that all quanties above are expressed as inhomogeneous coordinates in the world-coordinate system

Triangulation (midpoint algorithm)

- How should we find $\mathbf{v_1}$ and $\mathbf{v_2}$, which are necessary for the triangulation algorithm?

- Hint: We know the projection matrices for each of the views and pixel-coordinates for the corresponding image-points.

- Use, the decomposition method described earlier to convert image coordinates to world-coordinate representation

$$x_1^{\mathrm{w}} = \mathbf{M}^{-1}\mathrm{x}_1^{\mathrm{c}\prime} = (\mathbf{KR})^{-1}\mathrm{x}_1^{\mathrm{c}\prime} = \mathbf{R}^{\mathbf{T}}\mathbf{K}^{-\mathbf{1}}\mathrm{x}_1^{\mathrm{c}\prime}$$

Normalized image-coordinates (orientation expressed w.r.t. world-frame)

Pixel-coordinates (in camera-coordinate system)

Triangulation (midpoint algorithm)

Due to a limited resolution in each camera (finite number of pixels), the back-projected rays will generally not intersect.

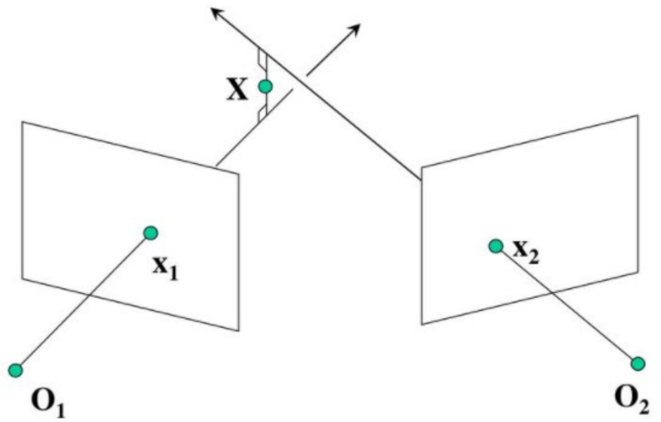
The midpoint-algorithm will however return the midpoint from the shortest possible line between the two rays…

This is equivalent to minimize:

$$\epsilon = d(\mathbf{l_1}, \mathbf{X})^2 + d(\mathbf{l_2}, \mathbf{X})^{\mathbf{2}}$$

Which is the 3D error between the two rays.

↳ It doesn't intersect due to the limited camera resolution



Linear Triangulation (minimizing the algebraic error)

- The mid-point method is considered to be poor for practical problems
- If the 3D point is physically much closer to one of the cameras, finding the midpoint is a poor strategy for the actual location of the point
- Another disadvantage is that this method can not naturally be extended to triangulation from multiple views.
- A better algorithm, which aims to minimize the algebraic error is presented in the following. The starting point of this algorithm is the projective equation from the pinhole camera model,

$$\mathbf{x}_i = \mathbf{P}_i\mathbf{X}$$

We can parametrize $\mathbf{P}$ in terms of the row vectors, $\mathbf{P_i} = \begin{bmatrix} \mathbf{p}_i^1 \\ \mathbf{p}_\mathbf{i}^2 \\ \mathbf{p}_\mathbf{i}^3 \end{bmatrix}$

$P = K[R \ t]$

Linear Triangulation

As we are dealing with homogenous coordinates we have that

$$\mathbf{x}_\mathrm{i} = \lambda_i \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

$$\mathbf{x}_\mathrm{i} = \mathbf{P}_\mathbf{i}\mathrm{X} \Rightarrow \begin{bmatrix} \lambda_i x_i \\ \lambda_i y_i \\ \lambda_i \end{bmatrix} = \begin{bmatrix} \mathbf{p}_i^1 \\ \mathbf{p}_i^2 \\ \mathbf{p}_i^3 \end{bmatrix} \mathbf{X}$$

For each row, we can form the equations:

$\lambda_i x_i = \mathbf{p}_i^1 \mathbf{X}$ (1)

$\lambda_i y_i = \mathbf{p}_i^2 \mathbf{X}$ (2)

$\lambda_i = \mathbf{p}_i^3 \mathbf{X}$ (3)

If we divide (1) and (2) with (3) we can eliminate λ_i, i.e.

$$x_i = \frac{\mathbf{p}_i^1 \mathbf{X}}{\mathbf{p}_i^3 \mathbf{X}} \qquad y_i = \frac{\mathbf{p}_i^2 \mathbf{X}}{\mathbf{p}_i^3 \mathbf{X}}$$

$$(\mathbf{p}_i^3 x_i - \mathbf{p}_i^1)\mathbf{X} = 0$$

$$(\mathbf{p}_i^3 y_i - \mathbf{p}_i^2)\mathbf{X} = 0$$

$$\begin{bmatrix} \mathbf{p}_1^3 x_1 - \mathbf{p}_1^1 \\ \mathbf{P}_1^3 y_1 - \mathbf{P}_1^2 \\ \mathbf{P}_2^3 x_2 - \mathbf{P}_2^1 \\ \mathbf{P}_2^3 y_2 - \mathbf{P}_2^2 \end{bmatrix} \mathbf{X} = 0 \implies \mathbf{AX} = 0$$

X is the right null-space of A (can be solved by SVD)

MATLAB:

```
[U,S,V]=svd(A); X=V(:,end);
```

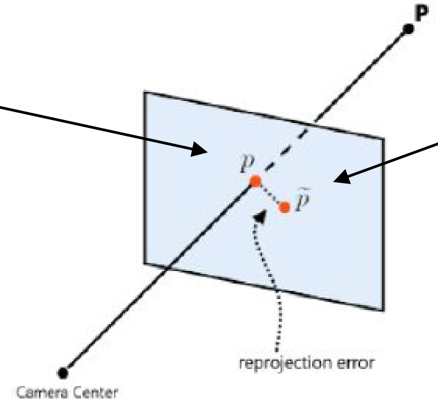
Non-linear Triangulation

- An even better triangulation method would be to minimize a meaningful geometric quantity such as the reprojection error,

$$\epsilon = d(\bar{\mathbf{x}}_1, \mathbf{x}_1)^2 + d(\bar{\mathbf{x}}_2, \mathbf{x}_2)^{\mathbf{2}}$$

  Where $\bar{\mathbf{x}}_1$, $\bar{\mathbf{x}}_2$ denotes the actual image-coordinates and $\mathbf{x}_1$, $\mathbf{x}_2$ the reprojections, i.e. $\mathbf{x}_1 = \mathbf{P}_1\mathbf{X}$ and $\mathbf{x}_2 = \mathbf{P}_2\mathbf{X}$

- This cannot be calculated in closed-form, but can be done iteratively by e.g. employing Gauss-Newton. The starting point from the algorithm would usually be a linear estimate of the triangulated point.

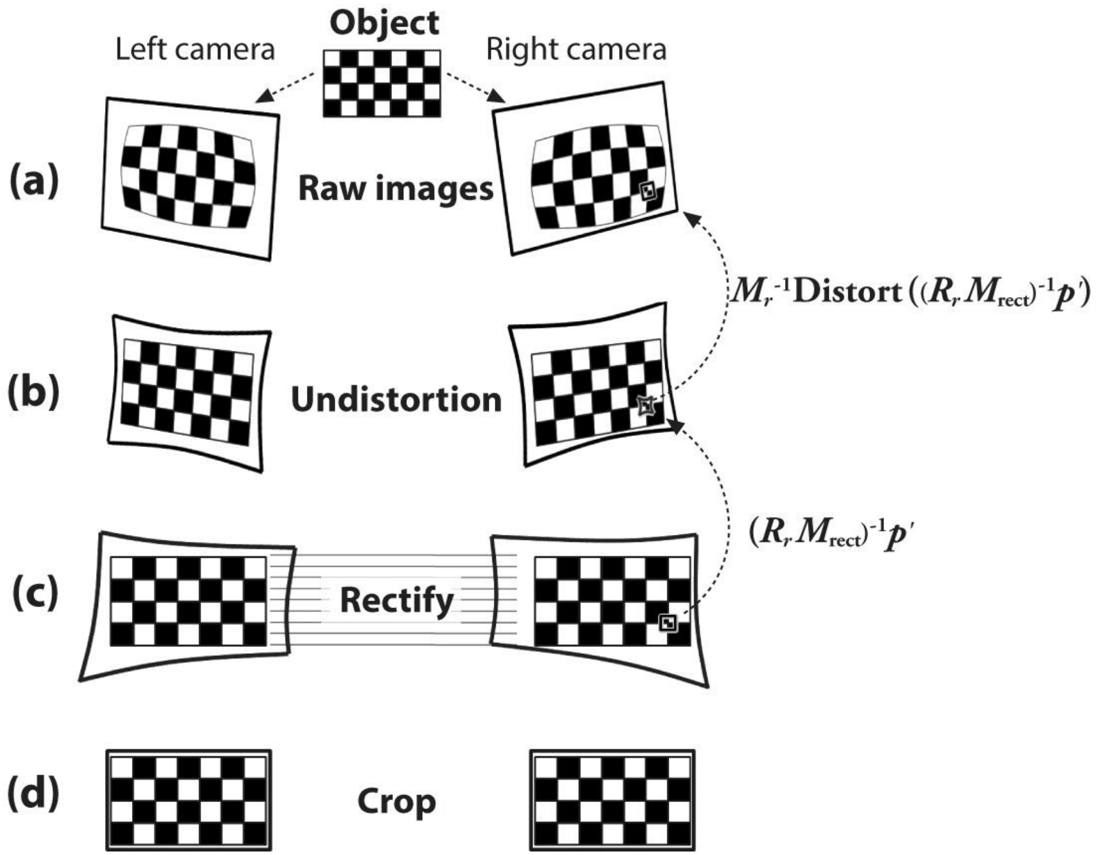


Non parallel cameras

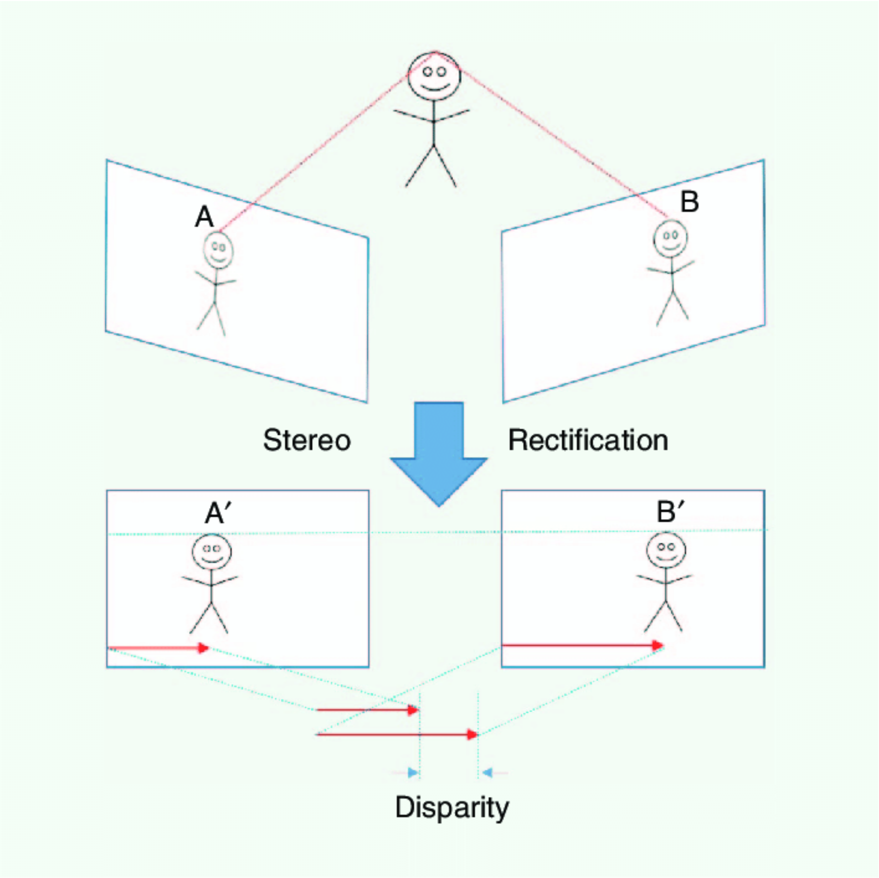
Stereo-rectification and disparity

- An alternative to the previous triangulation methods is to first perform a stereo-rectification of the image pair and then compute a disparity to recover the depth of point correspondes

- After stereo-rectification is the corresponding epipolar line horizontally as the process warps each image into a common plane (homography)



Stereo-rectification

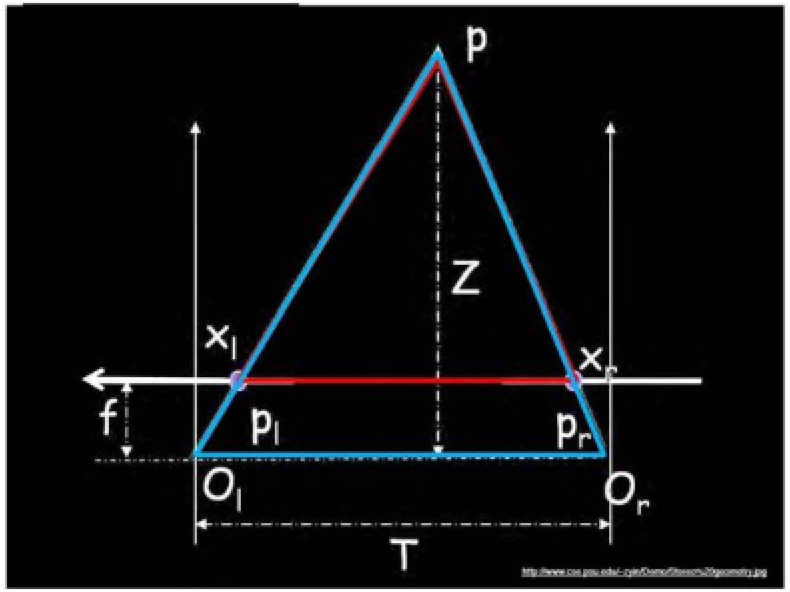


Triangulation by Disparity

Triangulation by Disparity

Quick
Skip

- Assume parallel optical axes, known camera parameters (i.e., calibrated cameras). We can triangulate via:



Similar triangles (p_l, P, p_r) and (O_l, P, O_r):

$$\frac{T + x_l - x_r}{Z - f} = \frac{T}{Z}$$

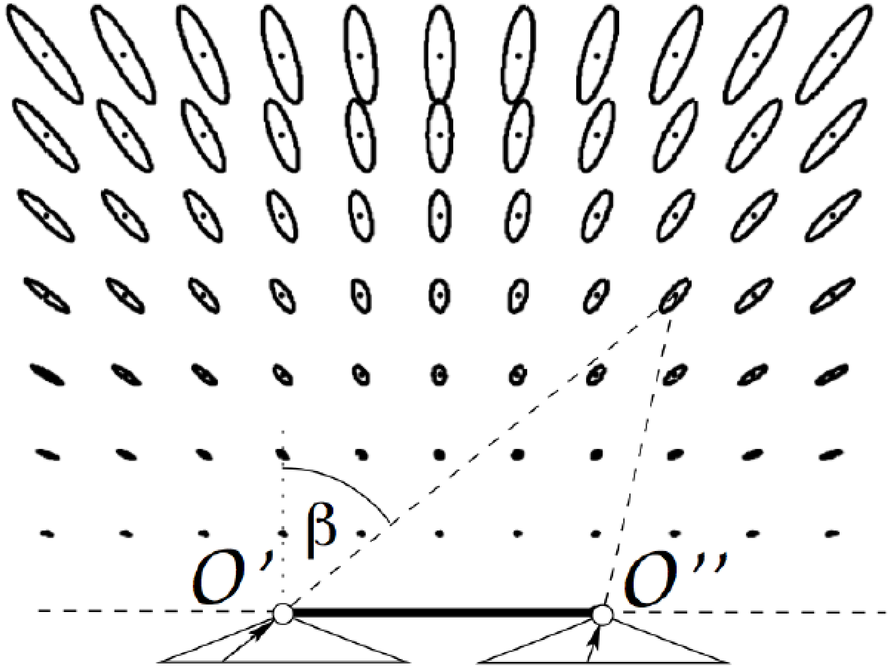
$$Z = f \frac{T}{x_r - x_l}$$

disparity

Slide credit: Kristen Grauman

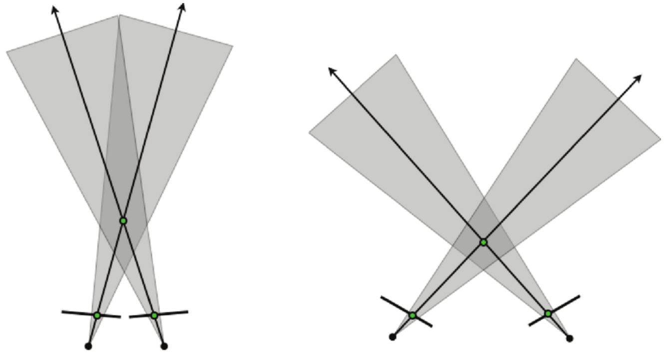
Triangulation Uncertainty

: We have a limited amount of pixels



$$x', x''$$

$$\sigma_{x'}^2 = \sigma_{x''}^2$$



(a) Small baseline (b) Large baseline