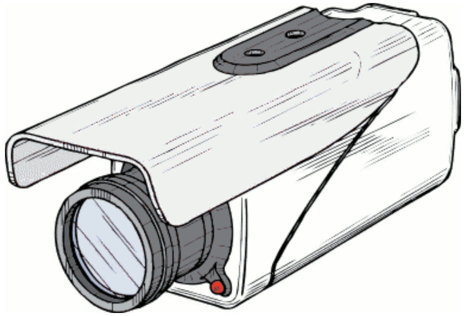
Univerza v Ljubljani

Machine perception
Multiple-view Geometry

Matej Kristan



Laboratorij za Umetne Vizualne Spoznavne Sisteme,
Fakulteta za računalništvo in informatiko,
Univerza v Ljubljani

Single-view geometry

- Structure and depth cannot be inferred from a 2D image

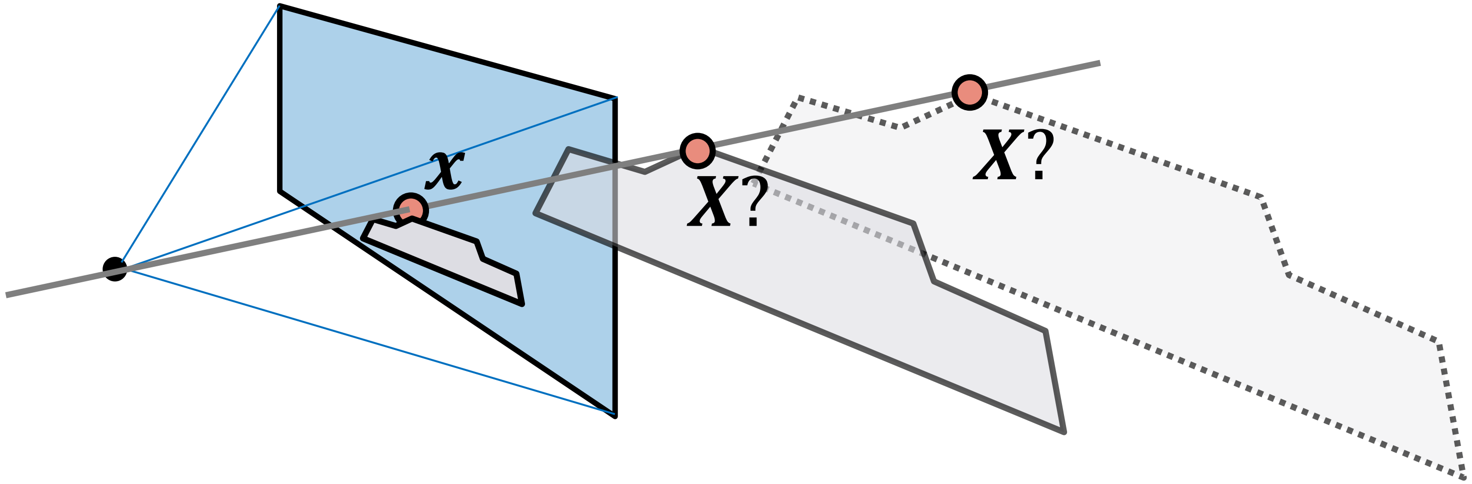
  (without a scene model or other kind of prior information)

The reason behind depth ambiguity

- All points along a ray that passes through a camera center are projected into the same point in the image plane.

- Impossible to infer 3D point from a single 2D point

  (without prior on the scene structure, that is)



Taking advantage of ambiguity

- Anamorphosis (earlier than 15th century)

Author: Julian Beever

Take two images = Stereo!

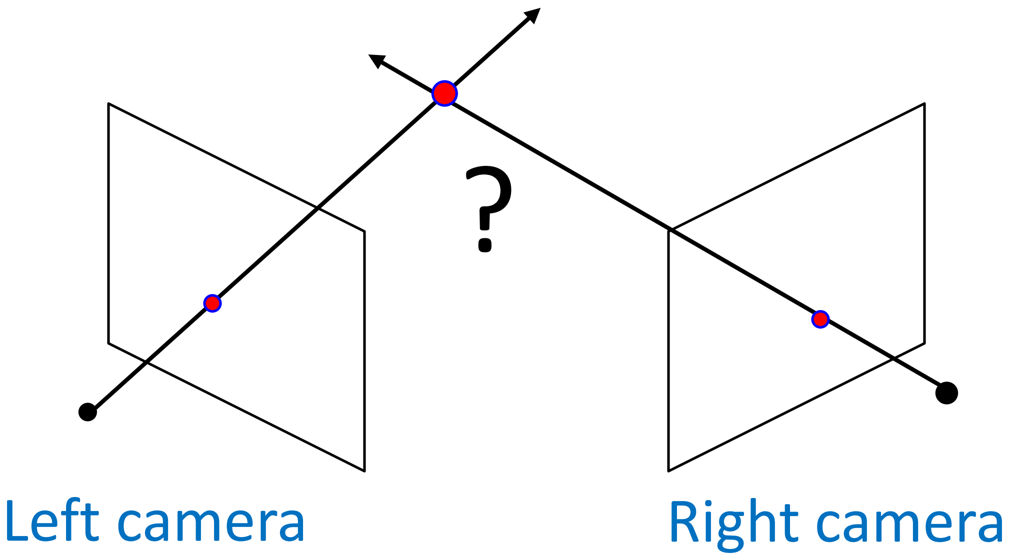
- Much easier using a pair of views...

Photos by: Jim Gasperini

Machine perception

STEREO GEOMETRY AND SCENE RECONSTRUCTION

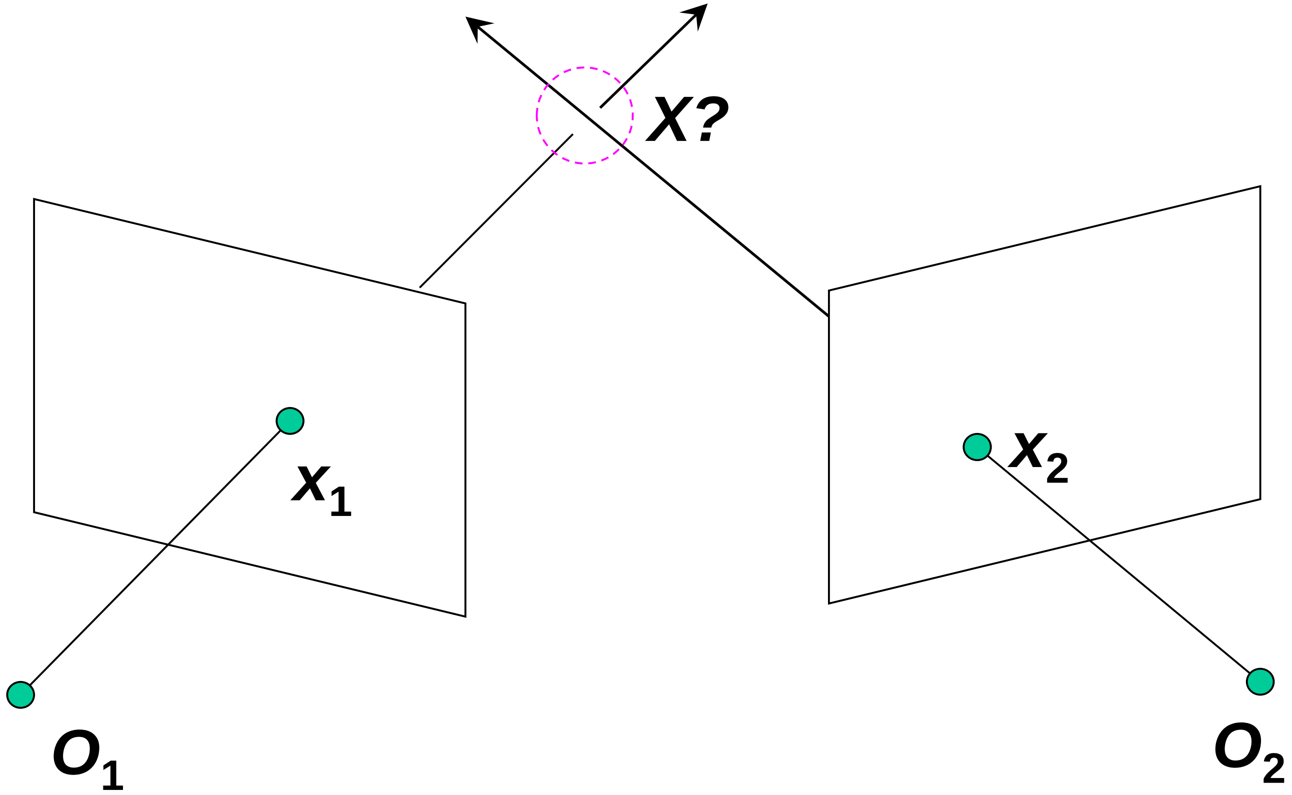
Depth estimation by triangulation



- The basic principle is triangulation
  - Reconstruction calculated by intersection of two rays
  - Assume:
    - Known camera position in 3D (calibration)
    - Correspondence between points is known

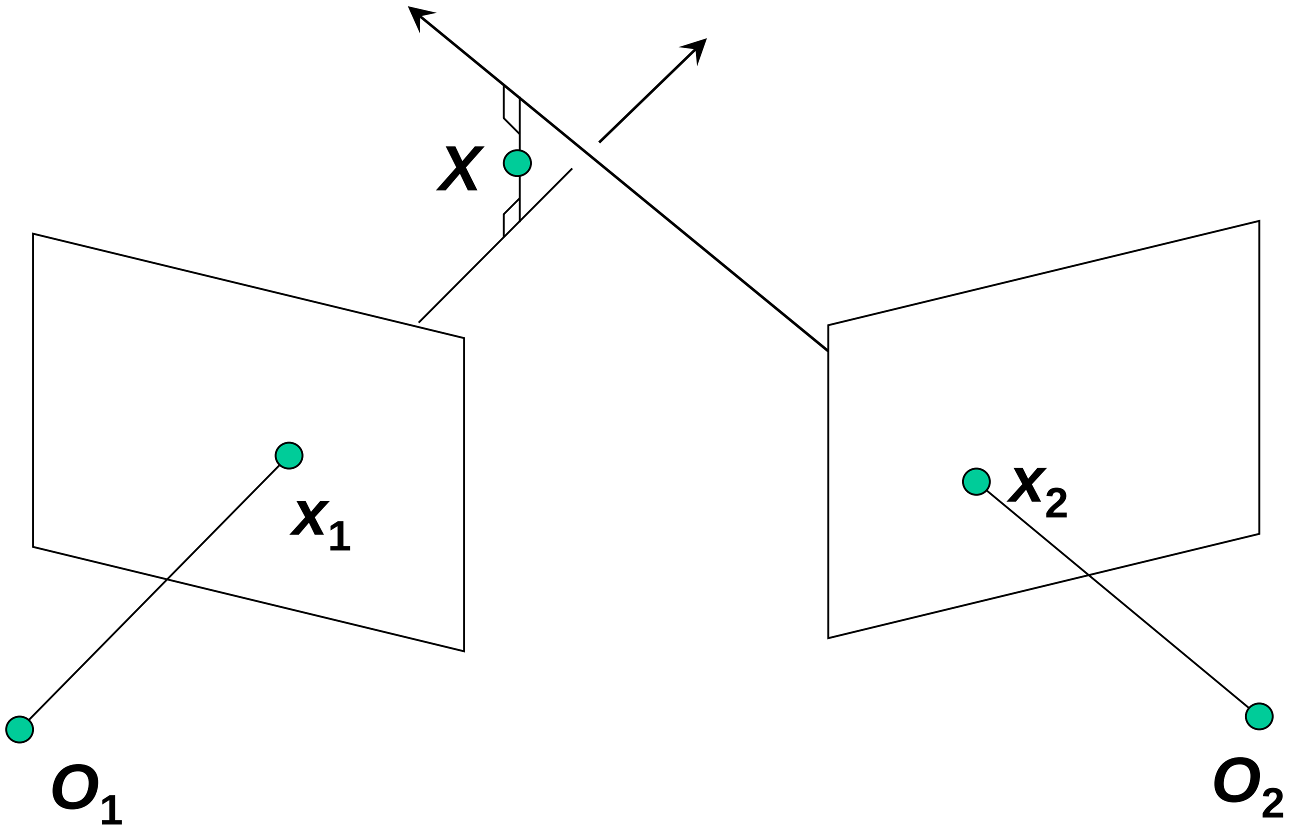
Triangulation by intersection

- Intersect a pair of visual rays, corresponding to x_1 and x_2.
- But because of numerical errors and noise, the rays will not intersect in practice!



Triangulation: Geometric approach

- Find the shortest segment connecting the two rays and take the value X in the middle.
- Not very principled…



Triangulation: A linear algebraic approach

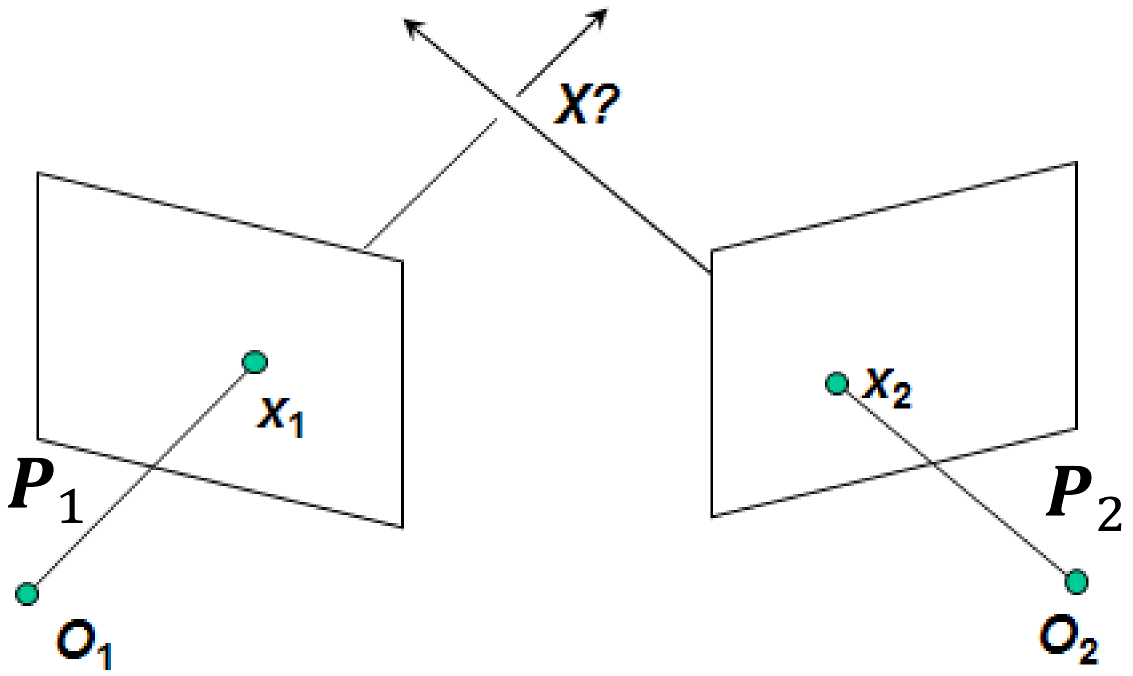
$$\lambda_1 \mathrm{x}_1 = \mathrm{P}_1\mathrm{X}$$
$$\lambda_2 \mathrm{x}_2 = \mathrm{P}_2\mathrm{X}$$

$$\mathrm{x}_1 \times \mathrm{P}_1\mathrm{X} = 0$$
$$\mathrm{x}_2 \times \mathrm{P}_2\mathrm{X} = 0$$

$$[\mathrm{x}_{1\times}]\mathrm{P}_1\mathrm{X} = 0$$
$$[\mathrm{x}_{2\times}]\mathrm{P}_2\mathrm{X} = 0$$

Recall: Vector product written in matrix form:

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



Triangulation: Linear algebraic approach

$$\lambda_1 \mathrm{x}_1 = \mathrm{P}_1 \mathrm{X}$$
$$\lambda_2 \mathrm{x}_2 = \mathrm{P}_2 \mathrm{X}$$

$$\mathrm{x}_1 \times \mathrm{P}_1 \mathrm{X} = 0$$
$$\mathrm{x}_2 \times \mathrm{P}_2 \mathrm{X} = 0$$

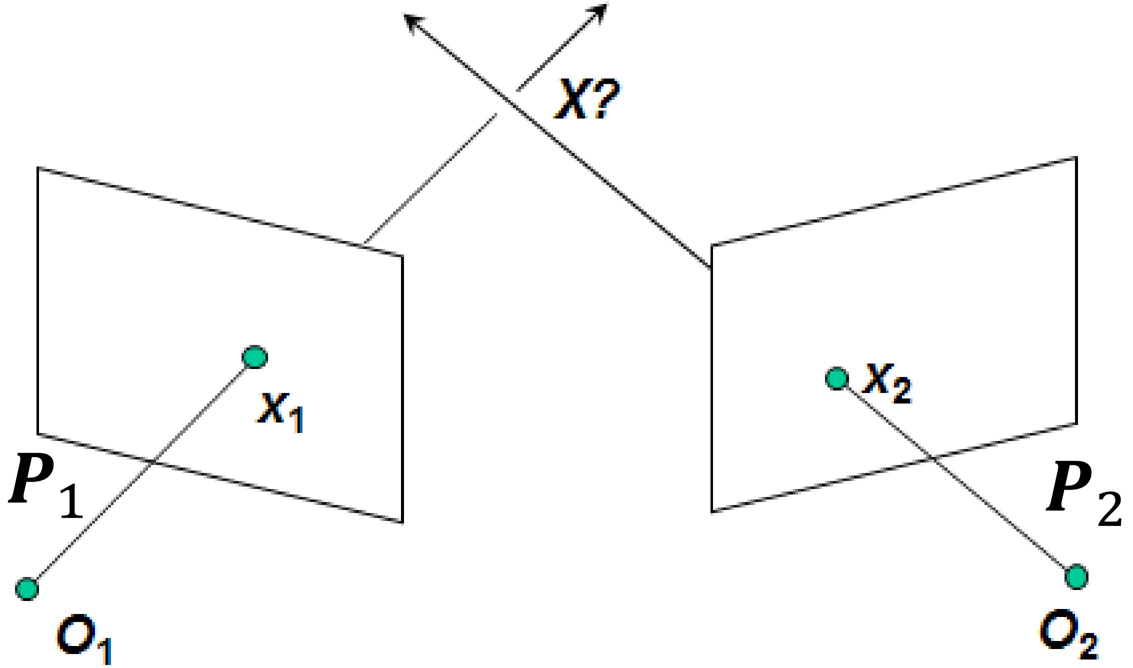
$$\boxed{\begin{array}{l}[\mathrm{x}_{1\times}]\mathrm{P}_1 \mathrm{X} = 0 \\ [\mathrm{x}_{2\times}]\mathrm{P}_2 \mathrm{X} = 0\end{array}}$$

Two independent equations each, 3 unknowns in X.

- Write a homogeneous system.

$$\boldsymbol{AX} = \mathbf{0}$$

- Solve by SVD. Solution for $\boldsymbol{X}$ is the eigenvector corresponding to the smallest eigenvalue.
- Much better than geometric approach, since it easily generalizes to multiple cameras.



Triangulation: Nonlinear refinement

- Find X that minimizes a sum of reprojection errors!

$$d^2(x_1, P_1 X) + d^2(x_2, P_2 X)$$

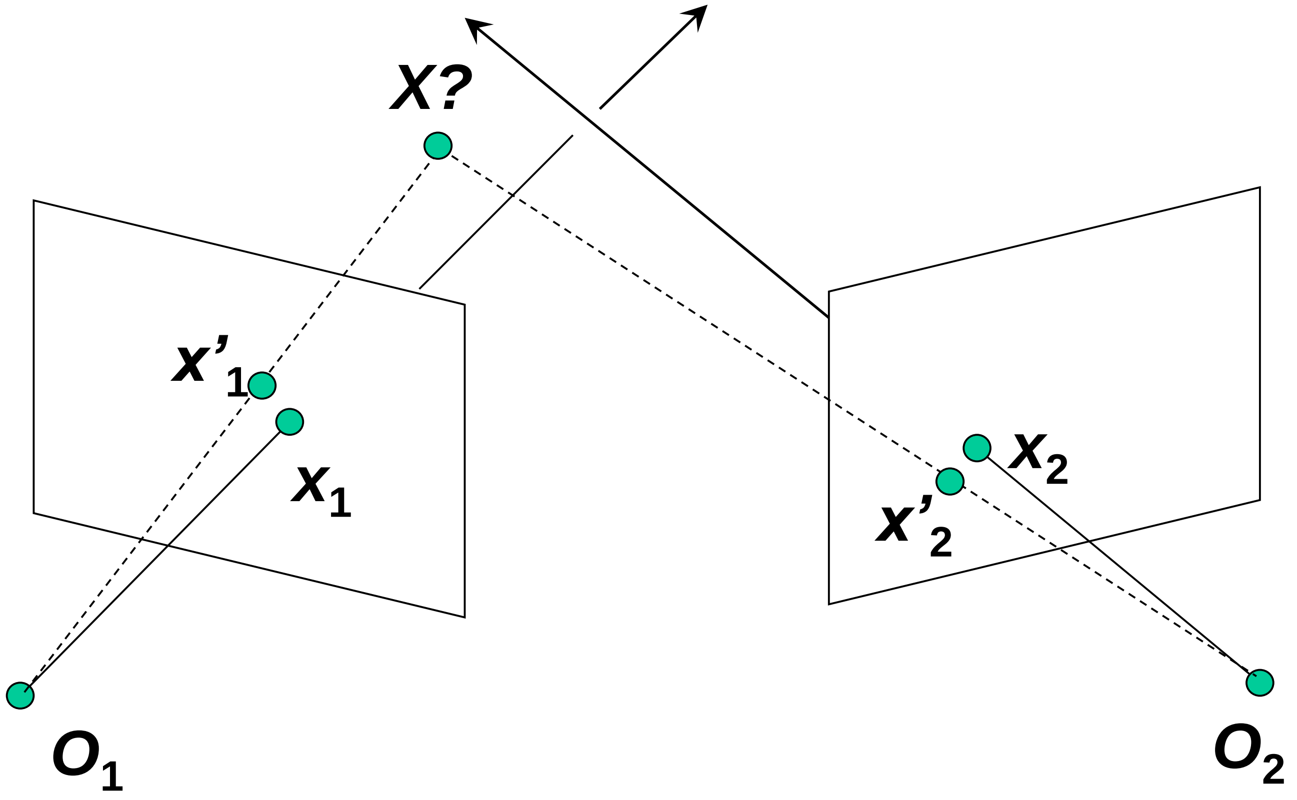


Image credit: Svetlana Lazebnik

Triangulation: Nonlinear refinement

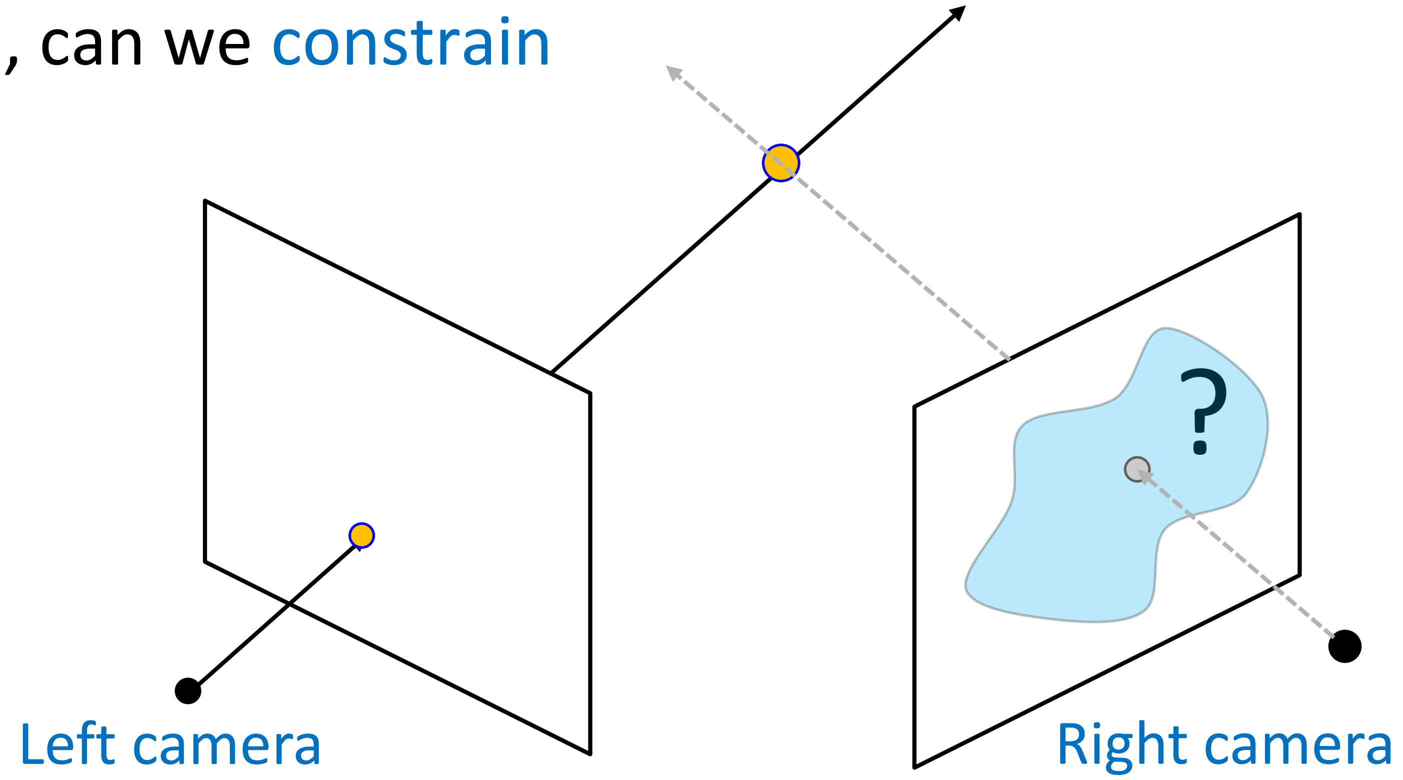
- Find X that minimizes a sum of reprojection errors!

$$d^2(x_1, P_1X) + d^2(x_2, P_2X)$$

- Most accurate, but does not have a closed-form solution.
- Requires iterative algorithm (bundle adjustment)
  - Initialize by DLT.
  - Optimize by Gradient descent or Gauss-Newton or Levenberg-Marquardt (see F&P Chapter. 3.1.2 or H&Z Appendix 6).

But in general correspondences are unknown

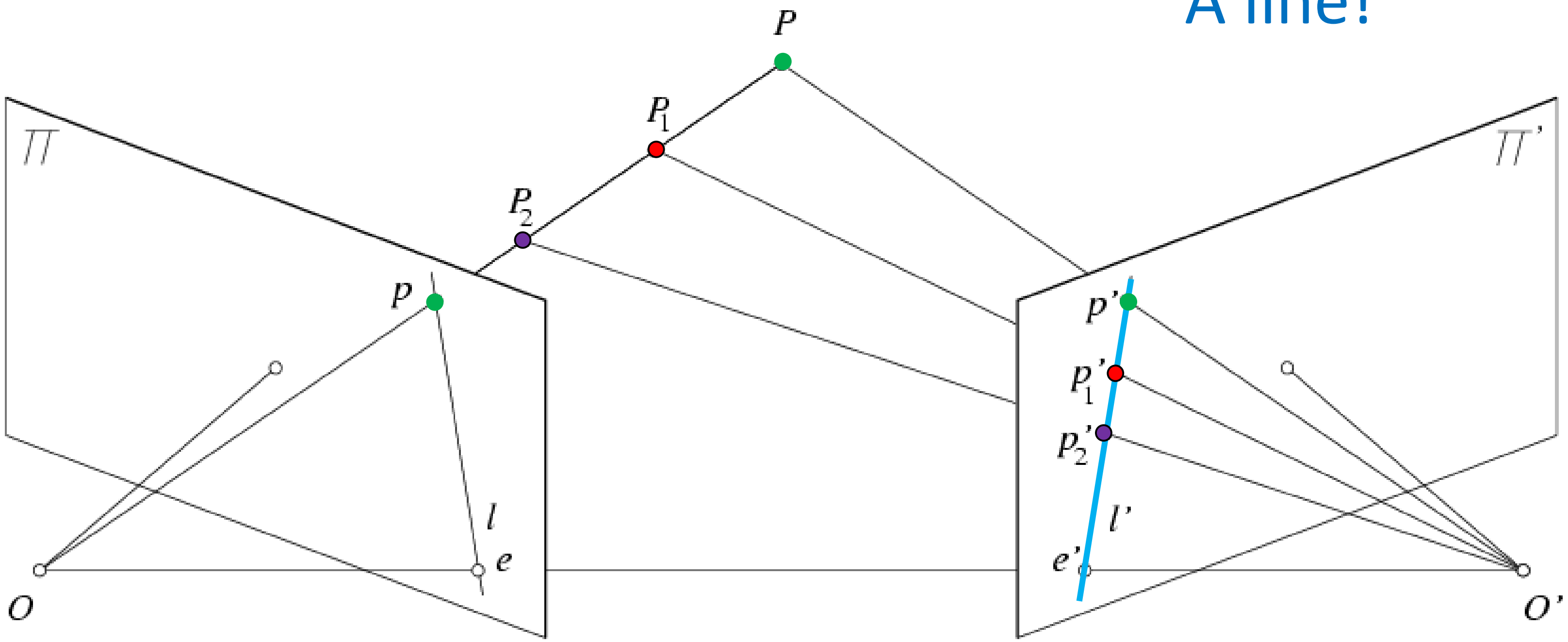
- Correspondences across images are usually not known in advance.
- Assume we know the location of the right camera with respect to the left camera.
- Given a point in the left image, can we constrain the search region of the point in the right image?



Epipolar constraints

- If the point *p* in the left image is known, where to look for its correspondence *p'* in the right image?

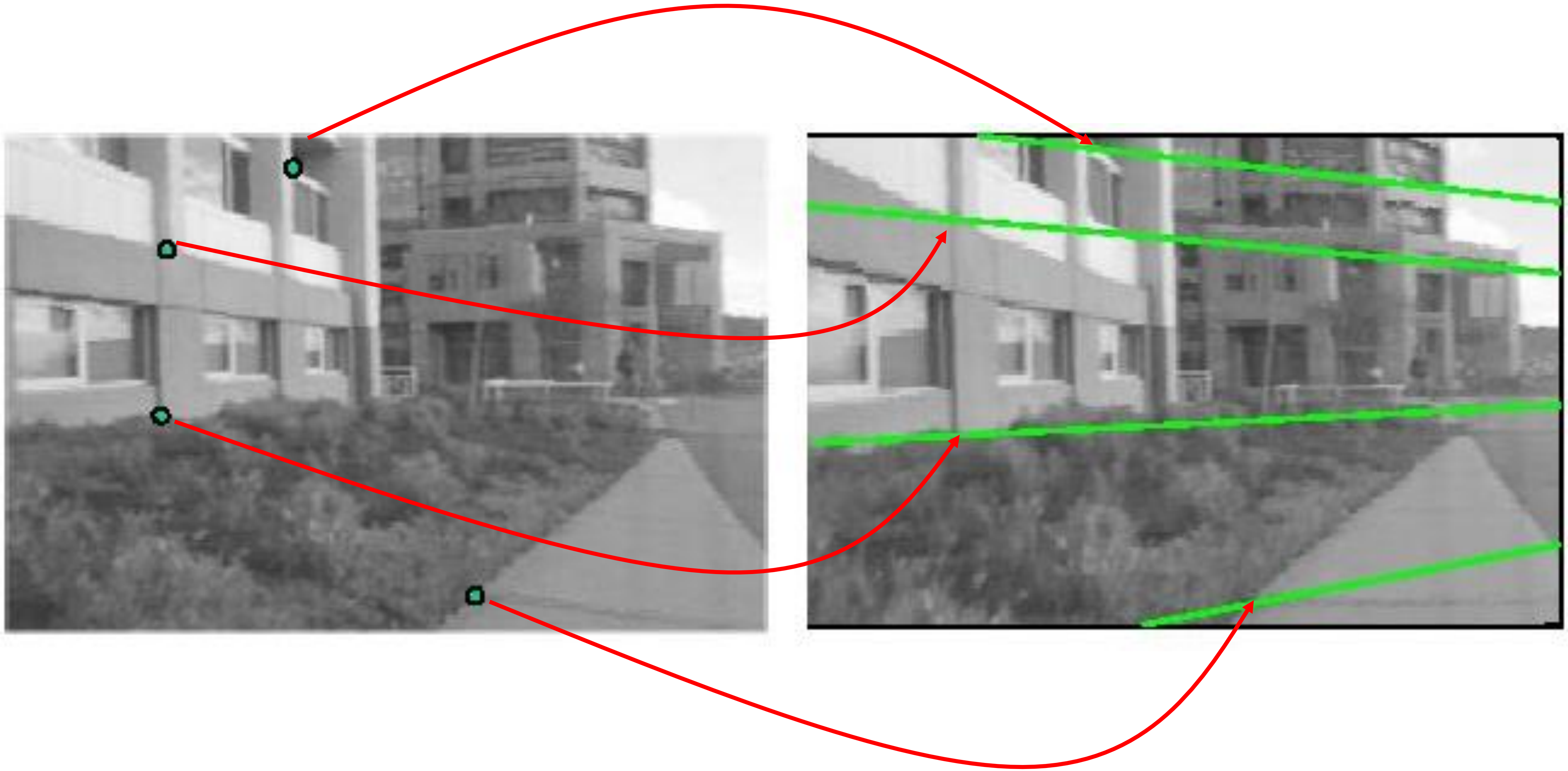
A line!



- Potential matches for *p* necessarily lie on the corresponding ***epipolar line l'***.

http://www.ai.sri.com/~luong/research/Meta3DViewer/EpipolarGeo.html

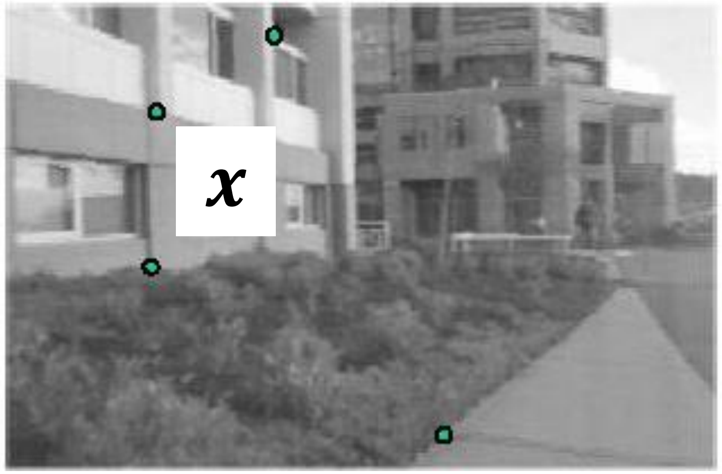
Example



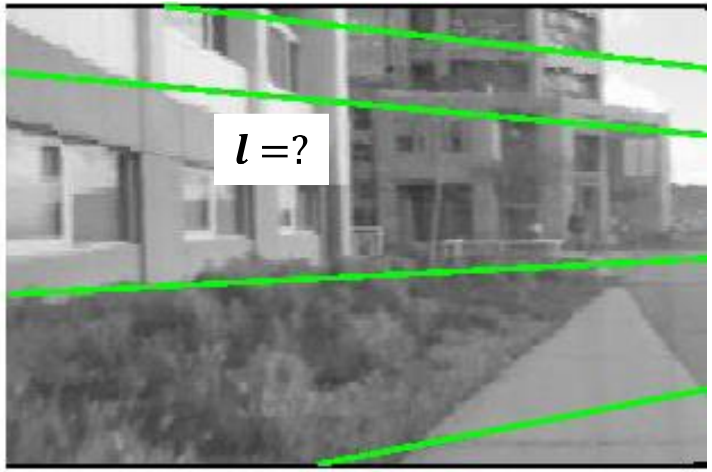
Derivation of the epipolar constraint

- The epipolar constraints, for a general stereo system:
  given a point $\boldsymbol{x}$ in the left image what is the equation of the epipolar line in the right image?

Left image



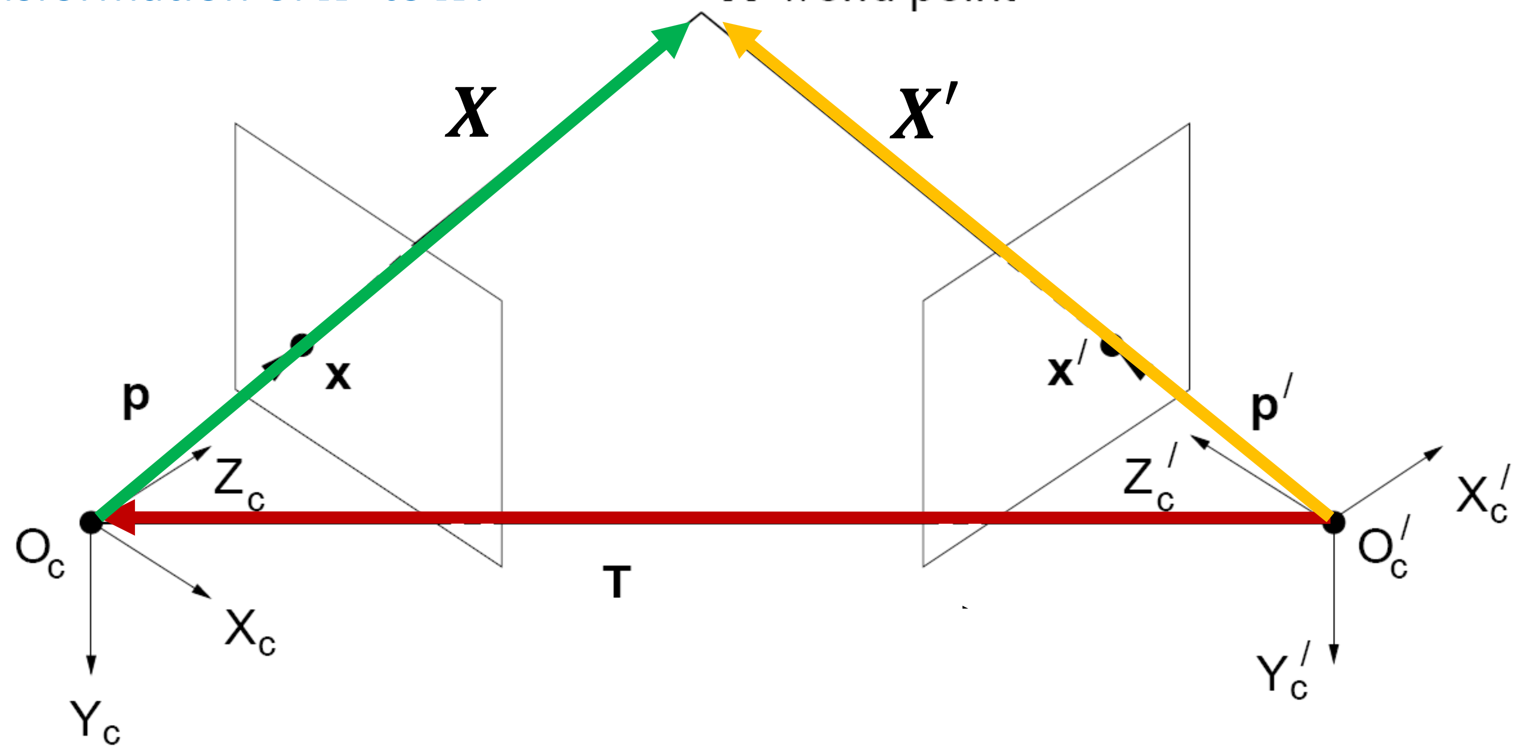
Right image



- Will look at two cases:
  - Calibrated cameras (known calibration matrices K, K')
  - Noncalibrated cameras (unknown calibration matrices K, K')

Epipolar constraint: A calibrated system

Note: X‘ is written in c.s. of O‘, while T and X are written in c.s. of O!
Write transformation of X' to X.



$$\boxed{\mathbf{X}} = \mathbf{R}\boxed{\mathbf{X}'} + \boxed{\mathbf{T}}$$

plane normal

$$\boxed{\mathbf{T} \times \mathbf{X}} = \mathbf{T} \times \mathbf{R}\mathbf{X}' + \mathbf{T} \times \mathbf{T}$$

$$= \mathbf{T} \times \mathbf{R}\mathbf{X}'$$

$$\mathbf{X}^T(\mathbf{T} \times \mathbf{X}) = 0$$

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

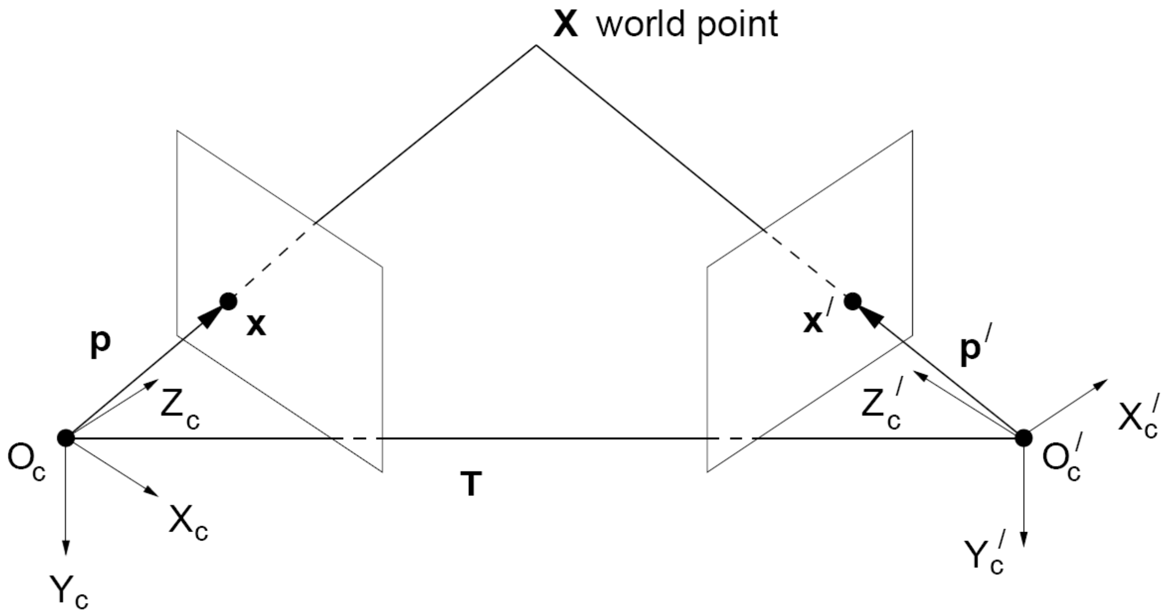
Epipolar constraint: A calibrated system

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

$$\mathbf{X}^T(\underline{[\mathbf{T}_\times]\mathbf{R}}\mathbf{X}') = 0$$

Let $\mathbf{E} = [\mathbf{T}_\times]\mathbf{R}$, then $\mathbf{X}^T\mathbf{E}\mathbf{X}' = 0$

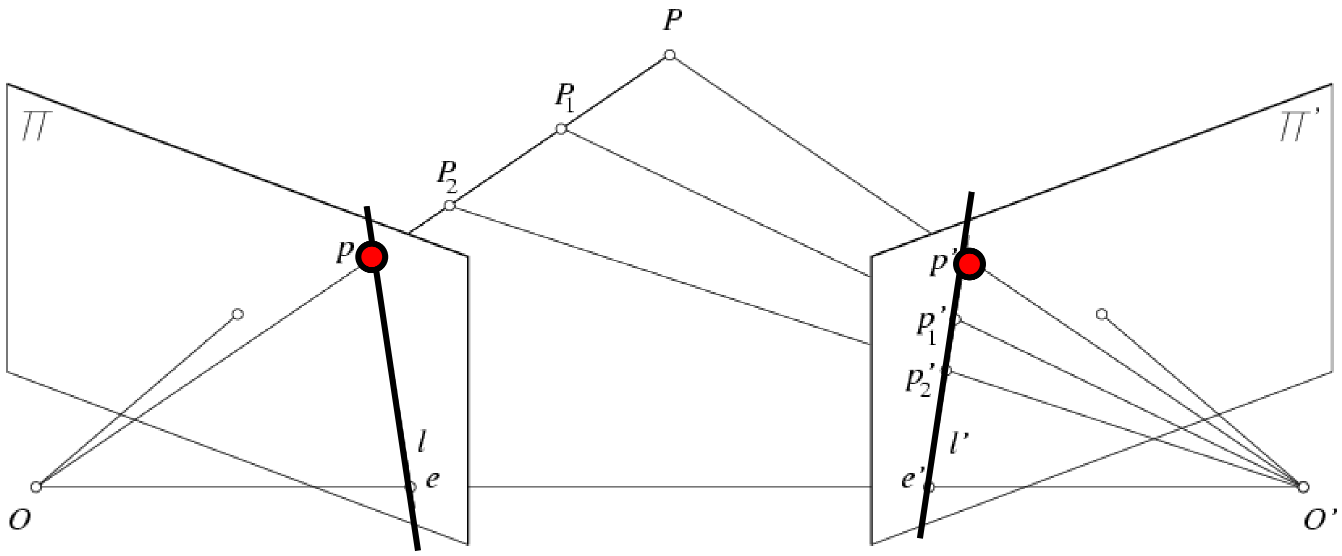
A 3D point written in the left and the right c.s., respectively.



- Points on image plane defined as $p = \lambda_1 X$ and $p' = \lambda_2 X'$, where $\lambda_1$and λ_2 are scalars.
- Then this holds: $\boldsymbol{p}^T\boldsymbol{E}\boldsymbol{p}' = \mathbf{0}$
- Matrix $\boldsymbol{E}$ is called an essential matrix, that relates the corresponding image points [Longuet-Higgins 1981]

Epipolar constraint: Essential matrix

- A 3D point is mapped to points $\boldsymbol{p}$ and $\boldsymbol{p}'$ which are related by $\boldsymbol{p}^T \boldsymbol{E} \boldsymbol{p}' = \mathbf{0}.$



https://brilliant.org/wiki/dot-product-distance-between-point-and-a-line/

$\boldsymbol{l}' = (\boldsymbol{p}^T \boldsymbol{E})^T$ the epipolar line vector $\boldsymbol{l}'$, defined in Π', containing $\boldsymbol{p}'$.

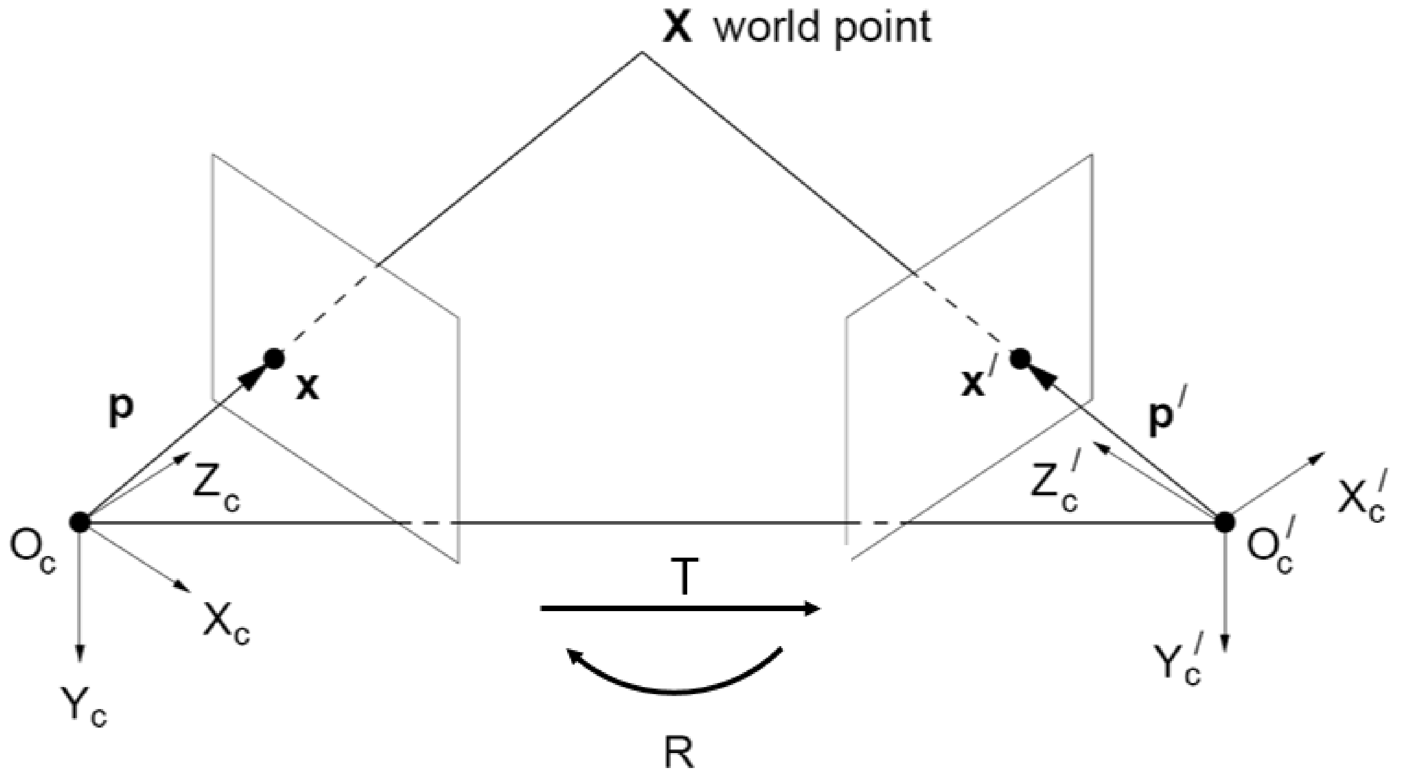
$\boldsymbol{l} = \boldsymbol{E}\boldsymbol{p}'$ the epipolar line vector $\boldsymbol{l}$, defined in Π, containing $\boldsymbol{p}$.

Epipolar constraint: Essential matrix

- Relates images of corresponding points (meters) in both cameras at a given rotation and translation.
- Can be calculated from known extrinsic parameters:

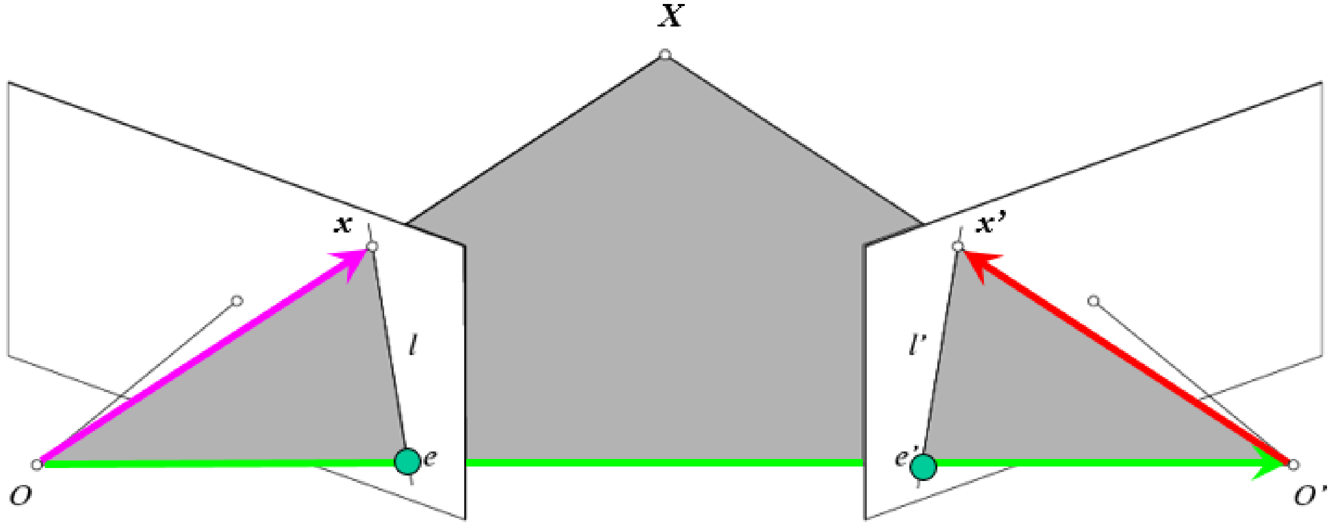
$$\mathbf{E} = [\mathbf{T}_\times]\mathbf{R}$$

Translation and rotation of the second camera with respect (w.r.t.) the first.



Epipolar constraint: A noncalibrated system

- Now consider image points in pixels!
  - x' & x … image plane coordinates (meters)
  - $\hat{x}'$ & $\hat{x}$… image sensor coordinates (pixels)



- Epipolar constraint for a calibrated system:

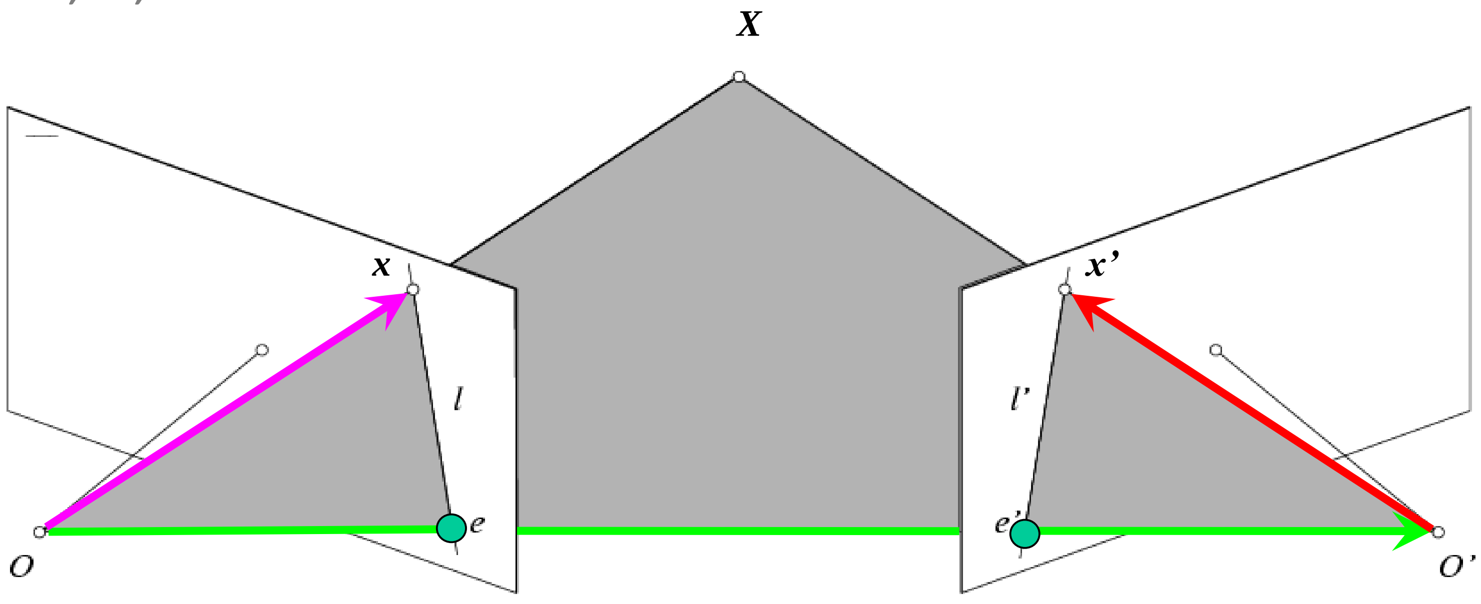
$$x^T E x' = 0$$

- Coordinates related by camera calibration matrix $\boldsymbol{K}$: $\hat{x} = Kx$ (pixels) (meters) $\hat{x}' = K'x'$ (pixels) (meters)

- Camera calibration matrices ***K*** and ***K'*** unknown → derive the epipolar constraint for the points in pixels

Epipolar constraint: A noncalibrated system

$\hat{x}, \hat{x}'$ in image pixels ; x, x' in meters



Epipolar constraint: $x^T E x' = 0$ $\Rightarrow$ $\hat{x}^T \underline{K^{-T} E K'^{-1}} \hat{x}' = 0$ $\Rightarrow$ $\hat{x}^T F \hat{x}' = 0$

(meters) (meters)

$\hat{x} = Kx$
$\hat{x}' = K'x'$
$\Rightarrow$
$x = K^{-1}\hat{x}$
$x' = K'^{-1}\hat{x}'$

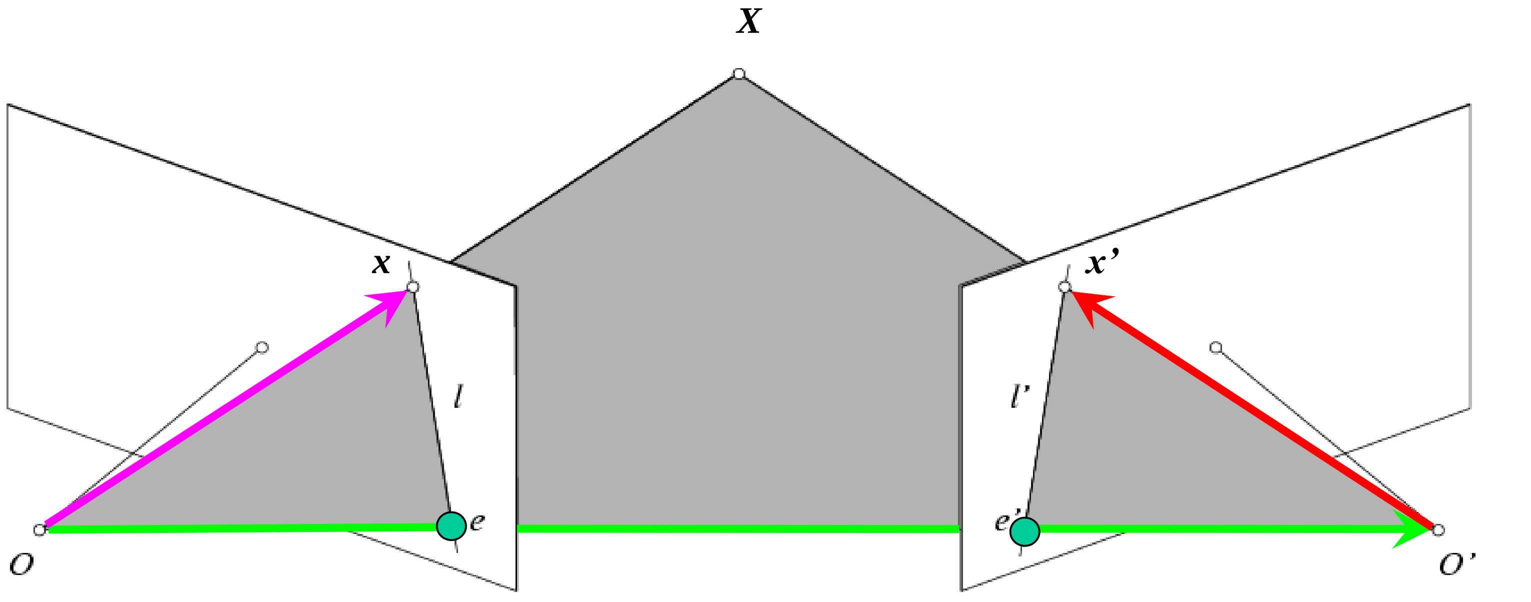
(pixels) (meters) (meters) (pixels)

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

Fundamental matrix

(Faugeras and Luong, 1992)

Epipolar geometry: Fundamental matrix

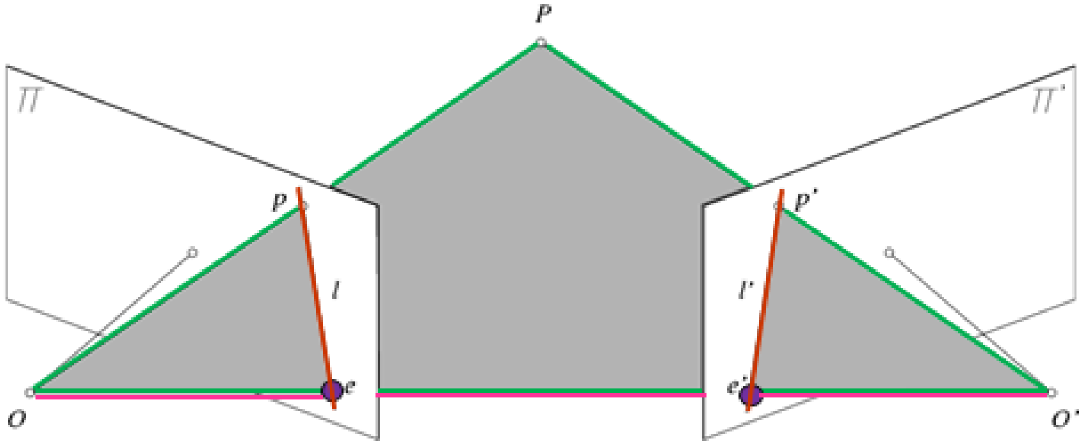


$$\hat{x}^T E \hat{x}' = 0 \quad \Rightarrow \quad x^T F x' = 0 \quad \text{with} \quad F = K^{-T} E K'^{-1}$$

- *Fx'* is epipolar line corresponding to *x'* ($l = F x'$)
- $F^T x$ is epipolar line corresponding to *x* ($l' = F^T x$)
- $Fe' = 0$ in $F^T e = 0$
- *F* is singular (rank=2)
- *F* has seven DoF

Epipolar geometry: Definitions

- *Baseline*: a line connecting the camera centers.
- *Epipole*: point where the baseline punctures the image plane.
- *Epipolar plane*: plane connecting two epipoles and a 3D point.
- *Epipolar line*: intersection of epipolar plane and image plane.



- All epipolar lines of a single image intersect at the camera epipole.

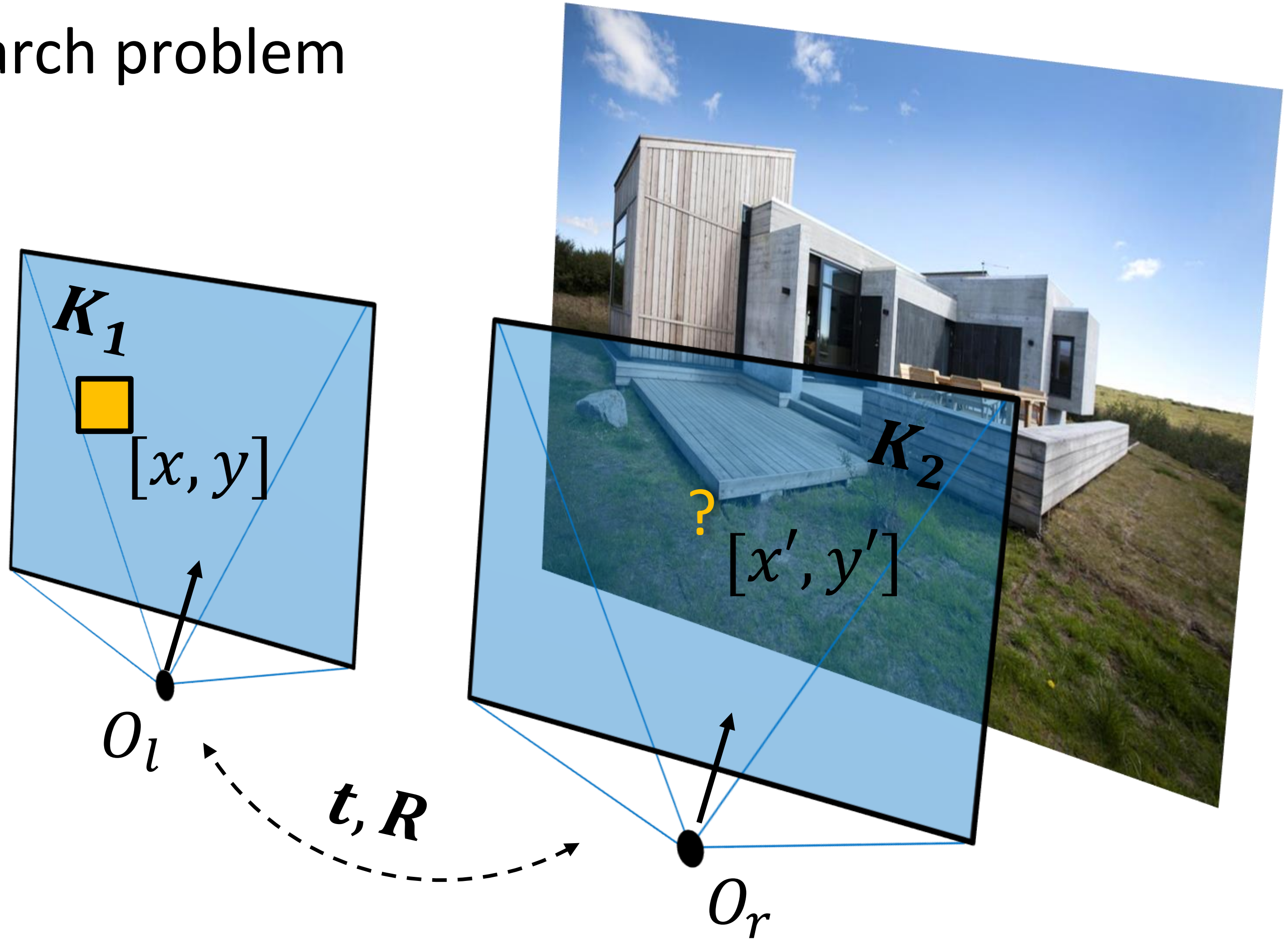
Special case: Geometry of a simple stereo

- Now consider a calibrated stereo system with parallel optical axes.
- This will simplify the search problem significantly…

$\mathbf{E} = \mathbf{T}_{\times}\mathbf{R}$

$\mathbf{p}^{\mathrm{T}}\mathbf{E}\mathbf{p}' = 0$

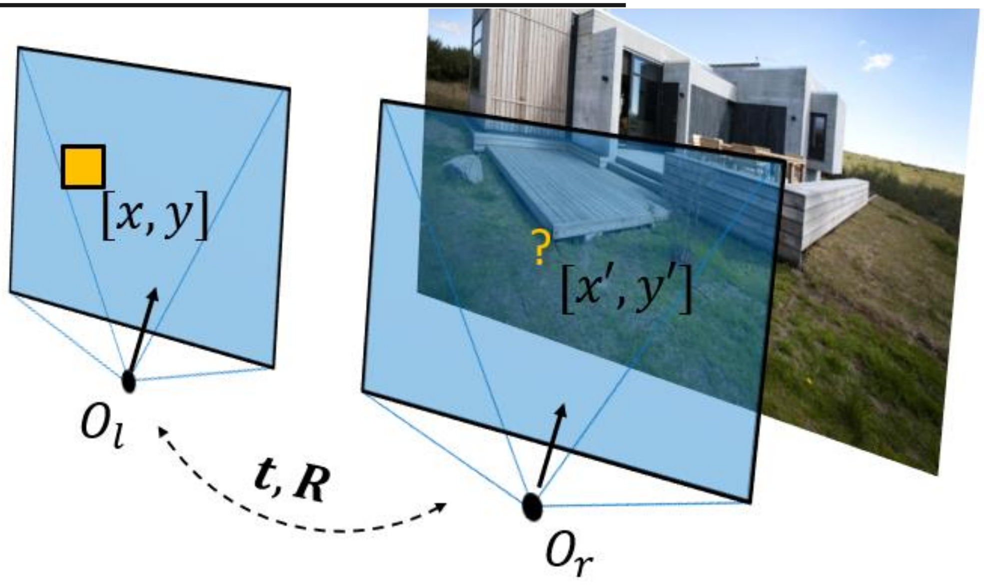
$\boldsymbol{l}' = (\boldsymbol{p}^T\boldsymbol{E})^T$



Given $[x, y]$ in the left image, where will the corresponding $[x', y']$ be in the right image?

Special case: Geometry of a simple stereo

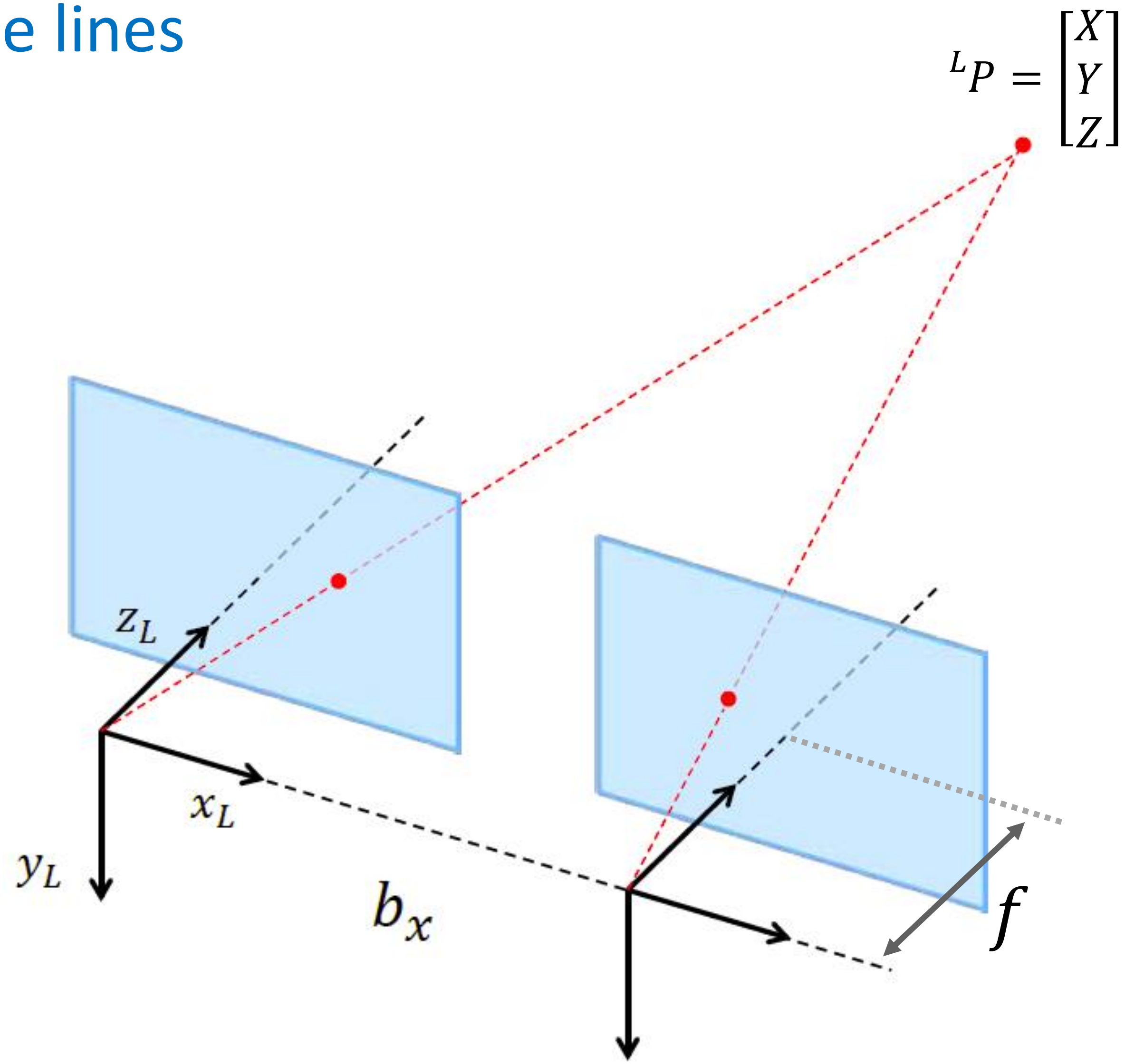
$\mathbf{E} = \mathbf{T}_{\times}\mathbf{R}$ $\mathbf{p}^{\mathrm{T}}\mathbf{E}\mathbf{p}' = 0$ $\boldsymbol{l}' = ?$



Geometry of a simple stereo

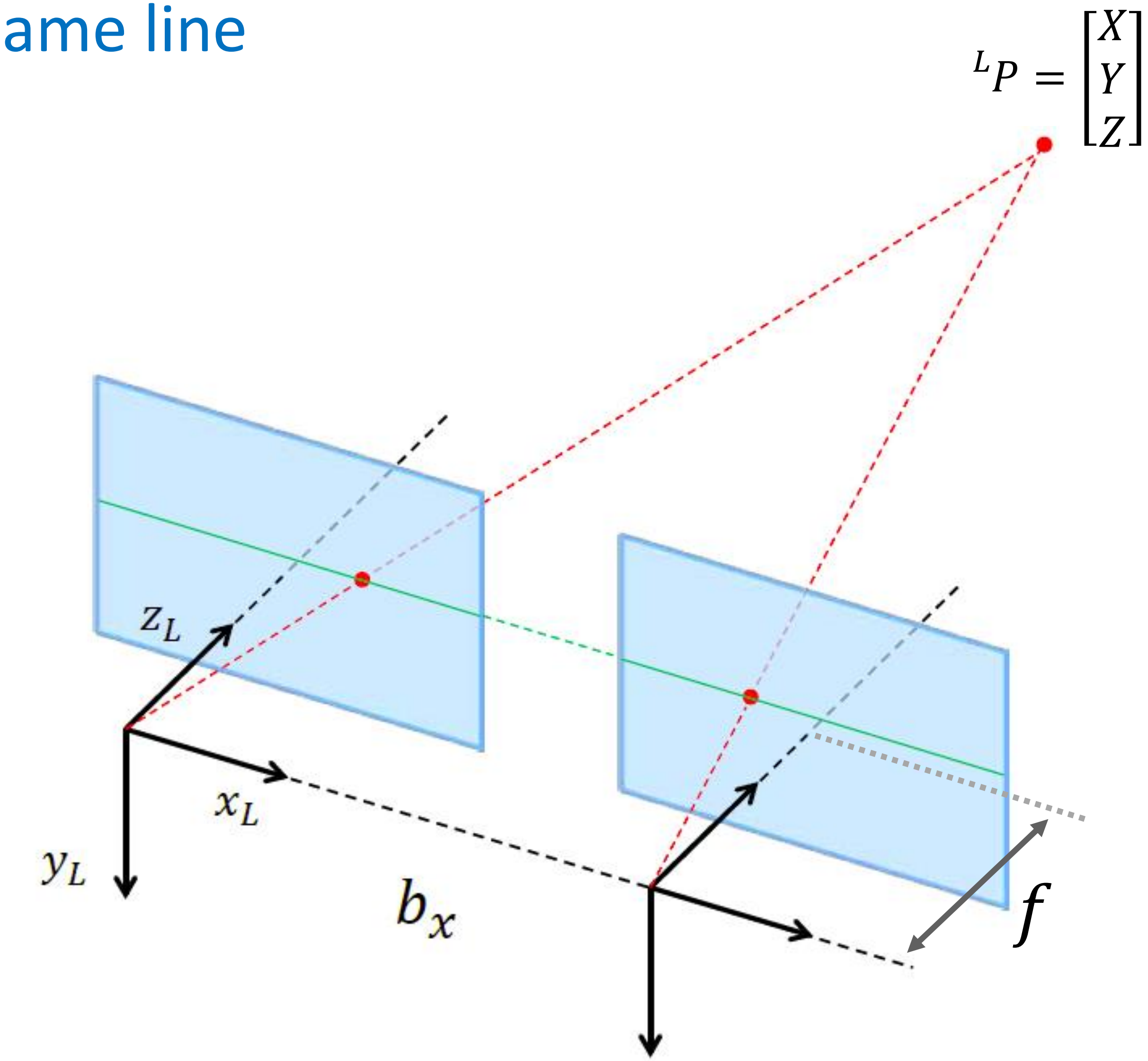
- Parallel optical axes with aligned image lines
- A 3D point written in the coordinate system of the left camera: ${}^{L}P$.
- Baseline b_x: displacement of the right camera along x_L.
- Focal length f: distance of image planes (in both cameras) from their projection centers.

Depth estimation simplifies…



Geometry of a simple stereo

- The corresponding points lie on the same line of pixels (epipolar line).



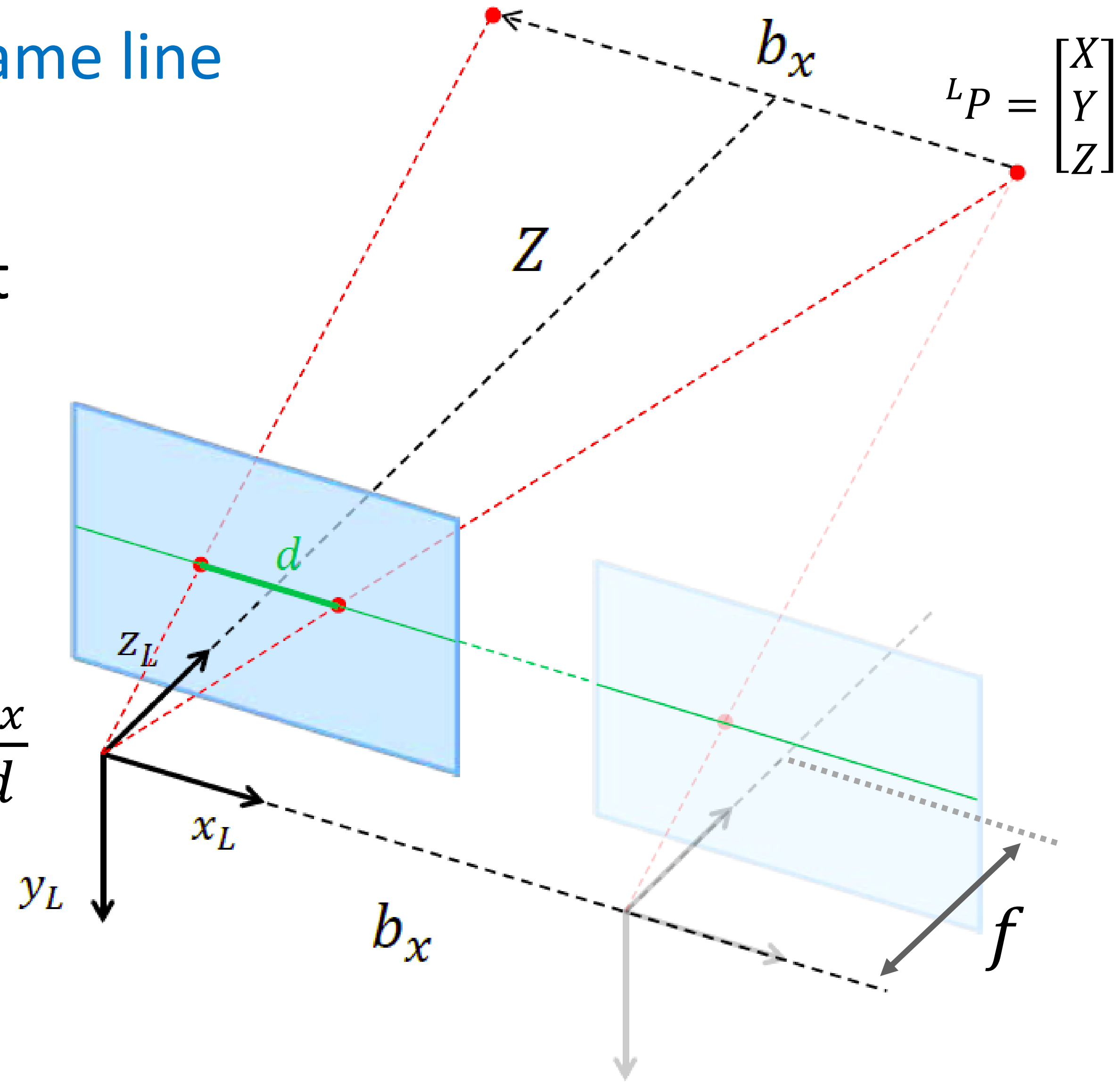
Geometry of a simple stereo

- The corresponding points lie on the same line of pixels (epipolar line)
- Align the right projection onto the left image (displace coordinates of the right projection by “$-b_x$”).
- Depth from disparity:

$$Z = f\,\frac{b_x}{d}$$

- 3D from disparity

$$X = x_L\,\frac{b_x}{d}\quad,\quad Y = y_L\,\frac{b_x}{d}\quad,\quad Z = f\,\frac{b_x}{d}$$

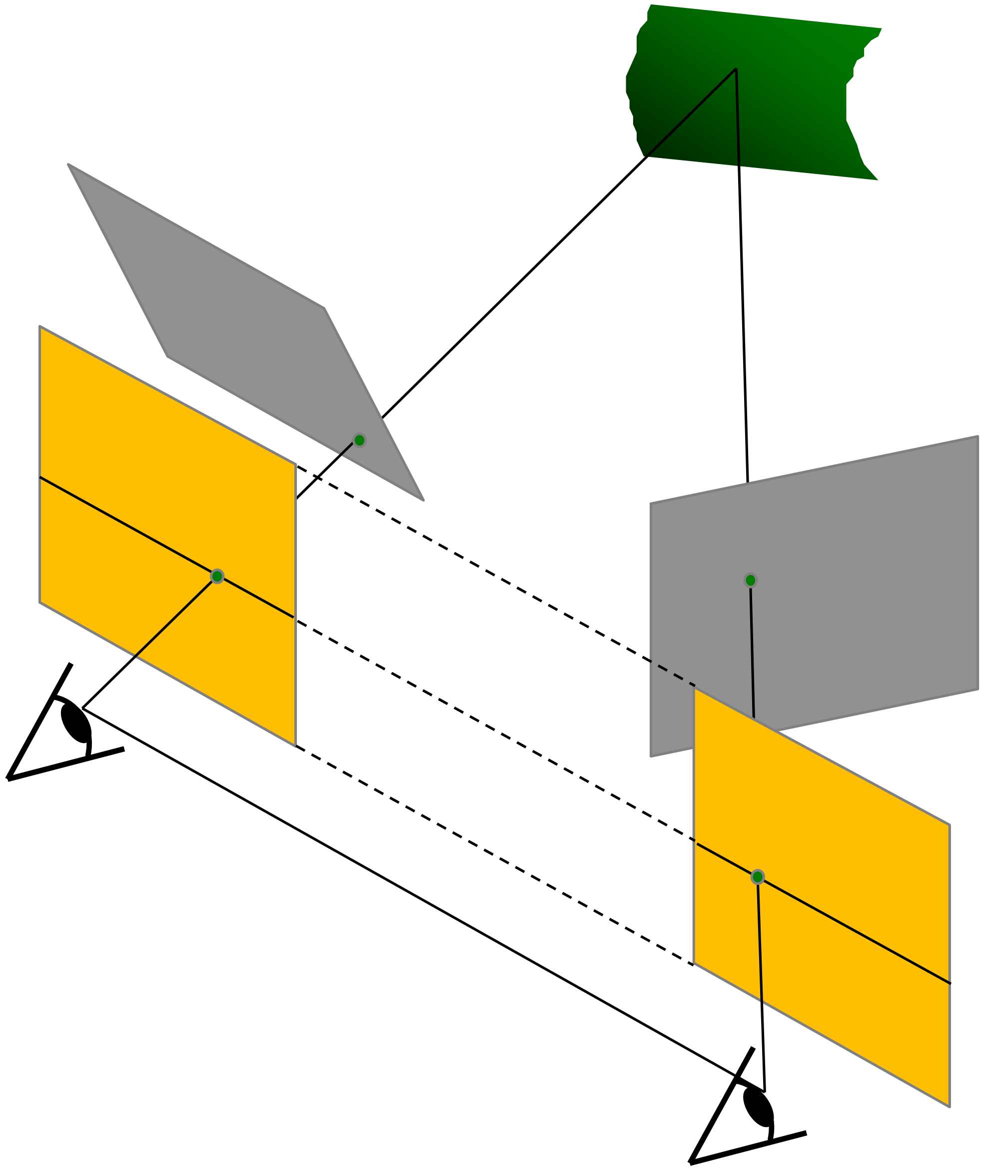


Stereo image rectification

- Convenient if the lines for searching the matches correspond to the epipolar lines – as simple as in parallel cameras system

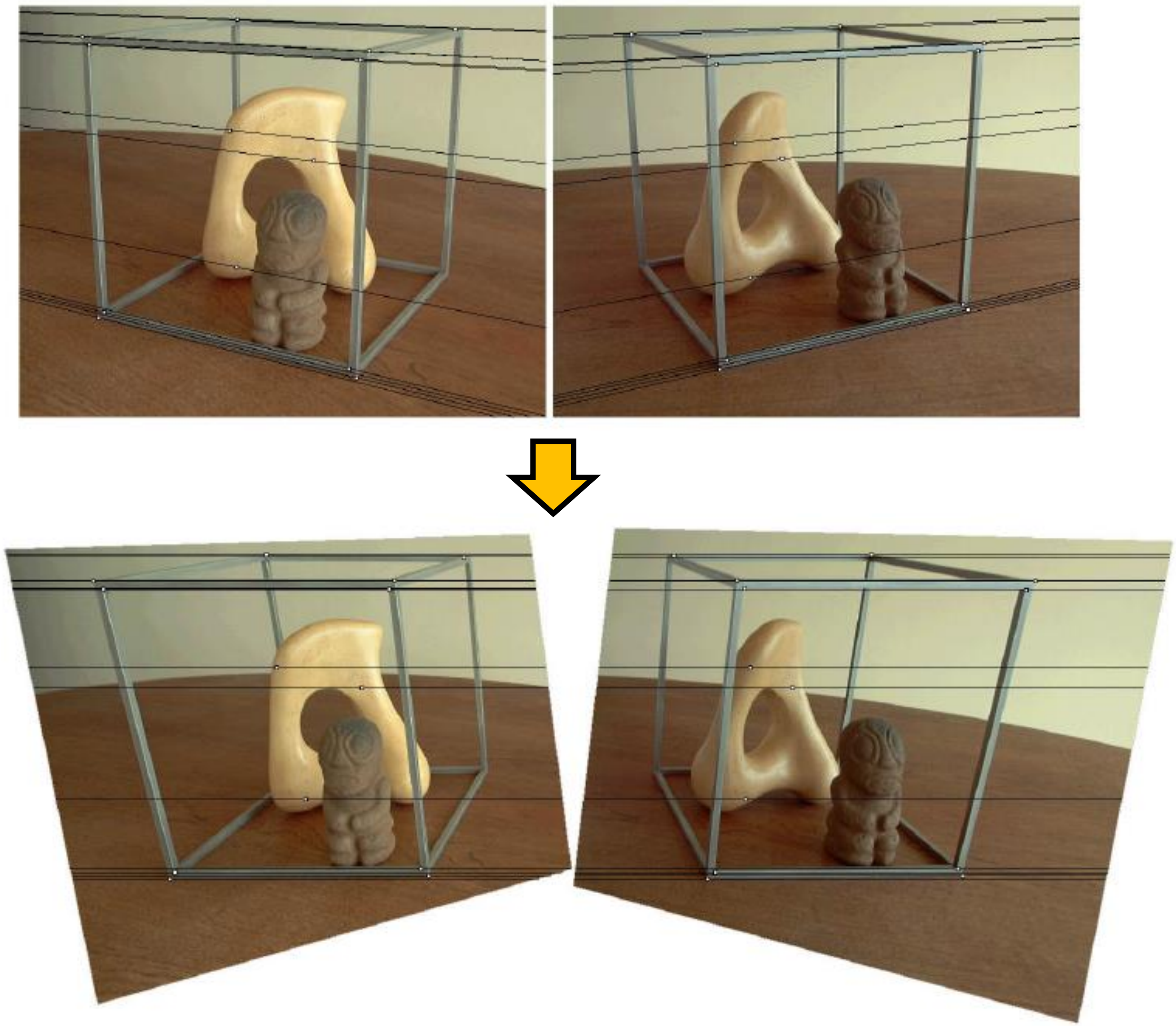
- Reproject image planes into a common plane, parallel to the baseline.

- Two homographies (3x3) – matrix transformation for reprojection of left and right image planes.



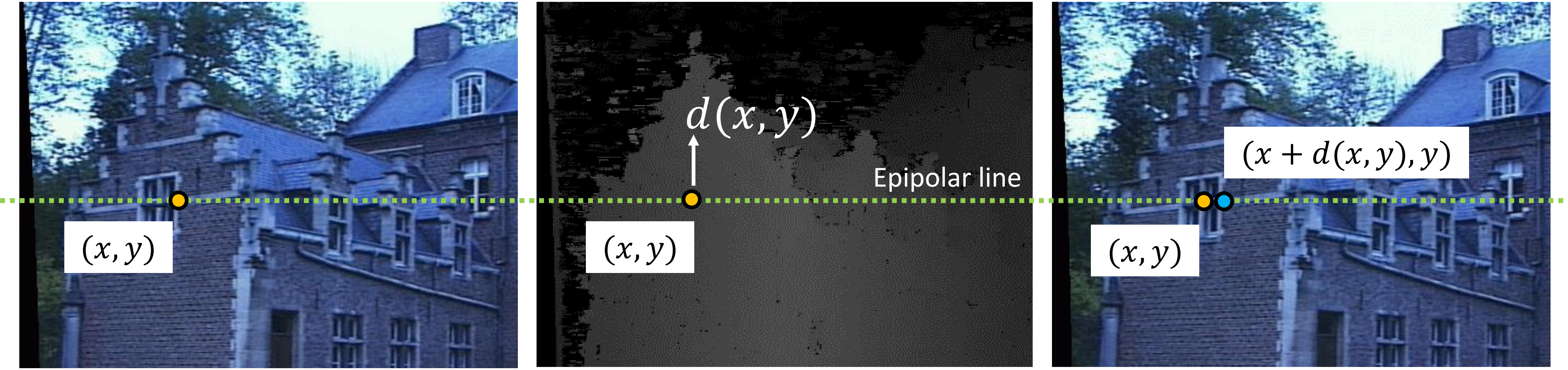
C. Loop & Z. Zhang, [Computing Rectifying Homographies for Stereo Vision](). CVPR'99

Stereo image rectification



Disparity relates the right image to the left

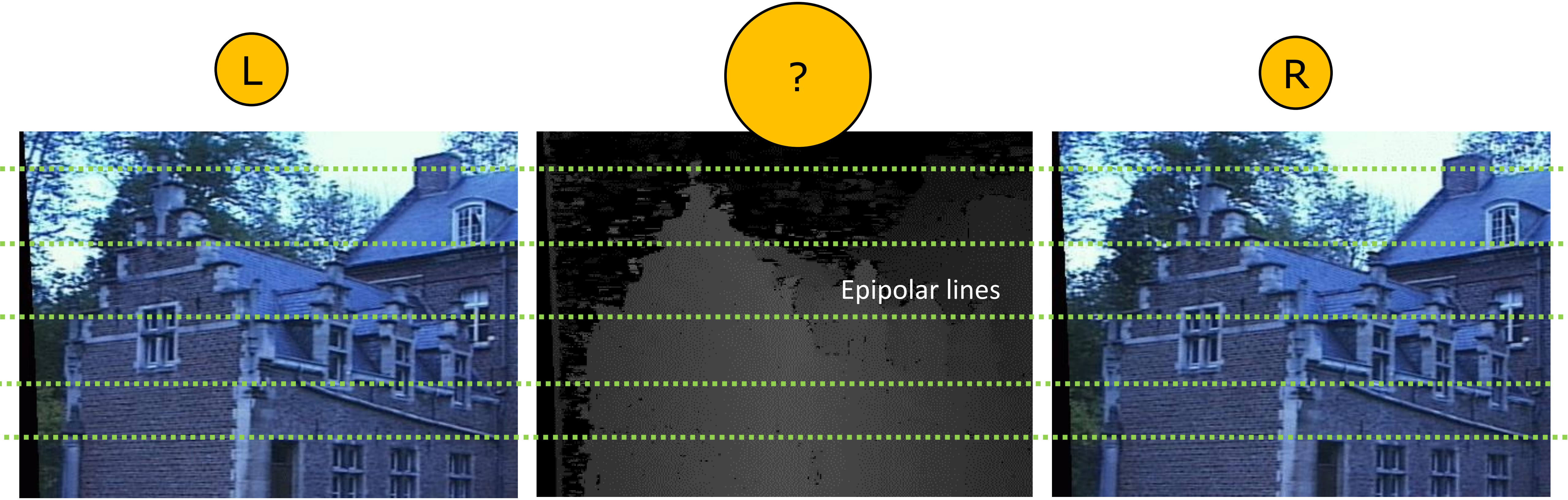
- Assuming perfectly aligned camera axes
- Coordinate of a corresponding point in the right image ==
  x coordinate of the point in the left camera + disparity value at that point



Left image: I(x,y)   Disparity map d(x,y)   Right image: I´(x´,y´)

Disparity estimation

- Disparity estimation problem: for each pixel in the left image, find the corresponding pixel in the right image. Difference in x is disparity.



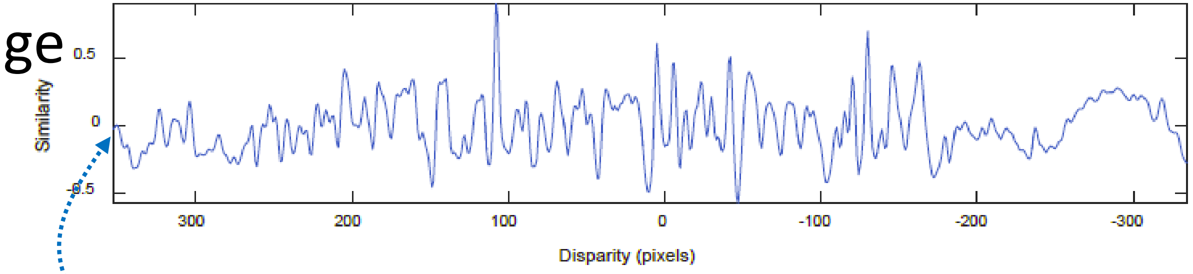
Left image: I(x,y)

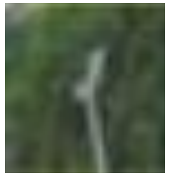
Disparity map d(x,y)

Right image: I´(x´,y´)

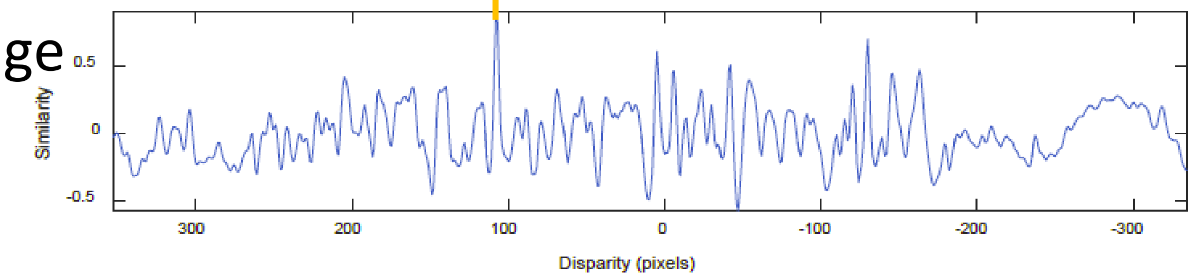
Disparity estimation

- For a patch centered at a pixel in the left image
- Compare to all patches in the right image along the epipolar line (same line)



NCC( , ) $= 0.01$

Disparity estimation



- For a patch centered at a pixel in the left image
- Compare to all patches in the right image along the epipolar line (same line)
- Select the patch with greatest similarity.
- Difference in position of left patch and right patch is the disparity.

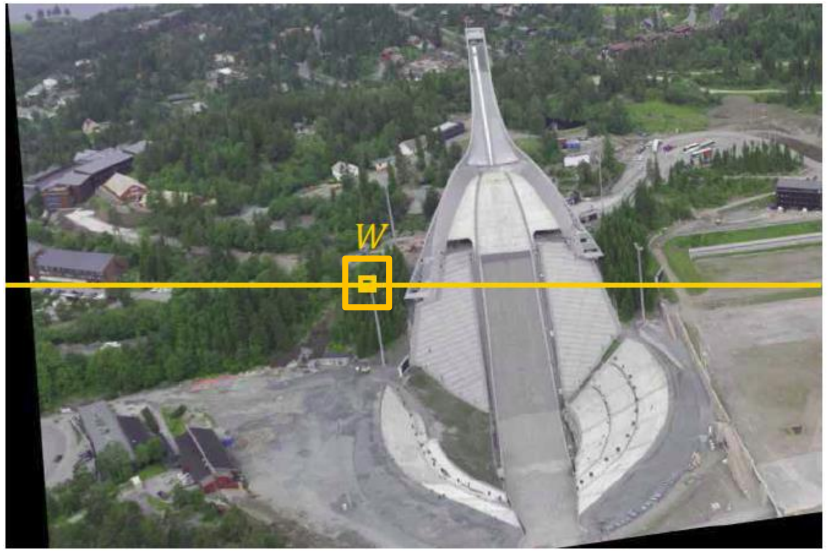
Images credit: Trym Vegard Haavardsholm

Disparity estimation

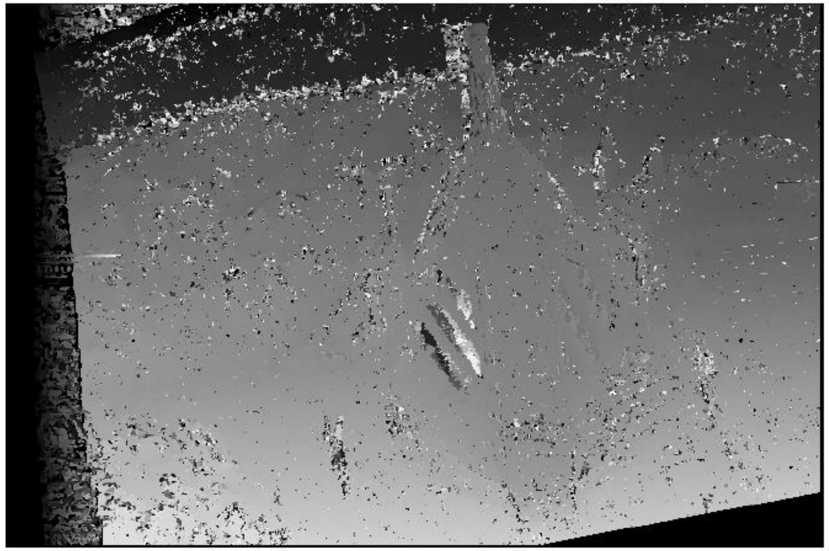
- In practice the disparity values are restricted to a reasonable range of viable disparities.
- E.g.: disparity for an object very far away from the target is 0. d_{max} is specified by the minimum distance of an object from the camera (see geometrical model of a simple stereo system)

Disparity: influence of the window size

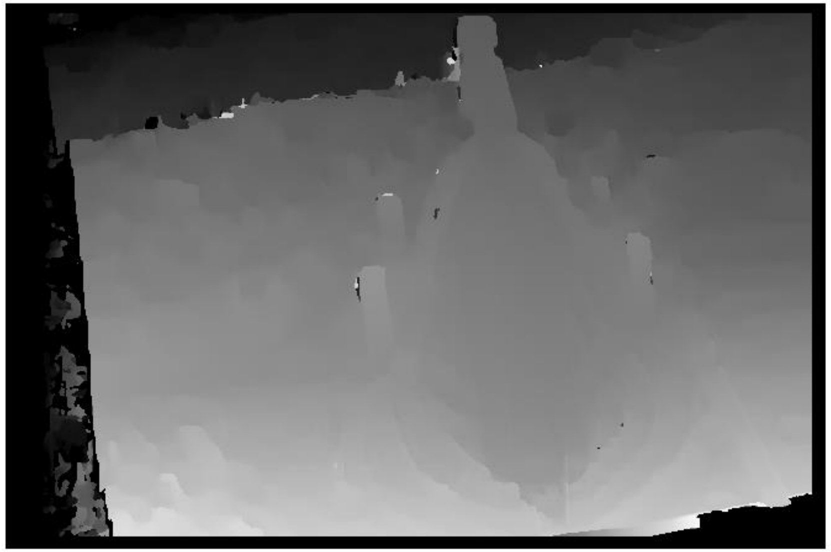
Left image



Small W



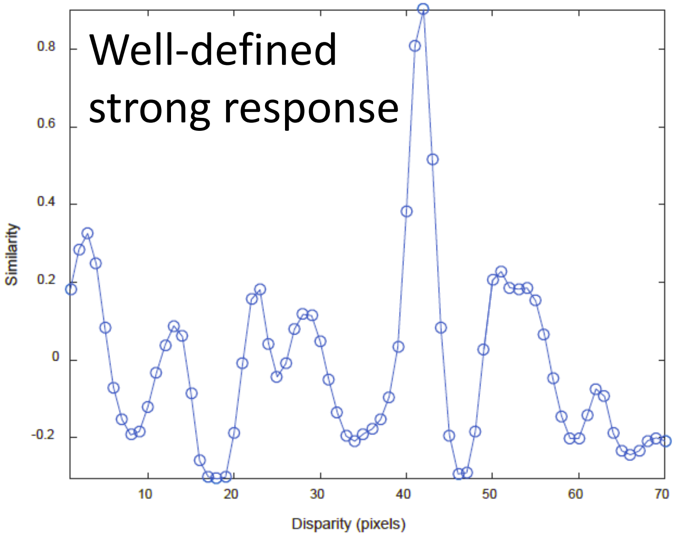
Large W

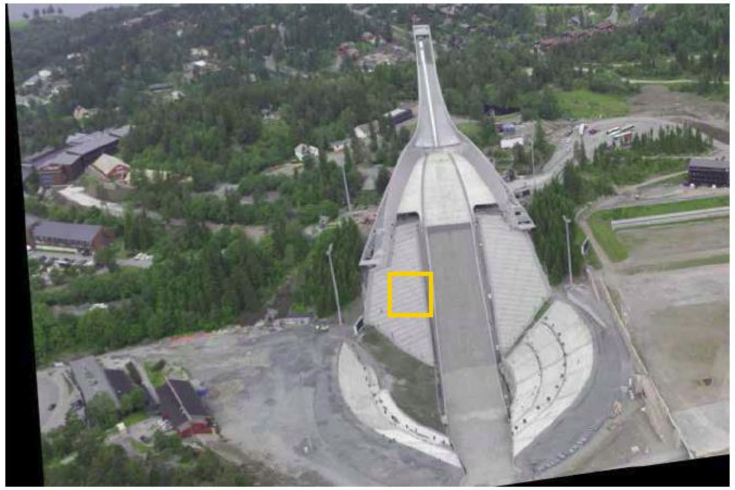


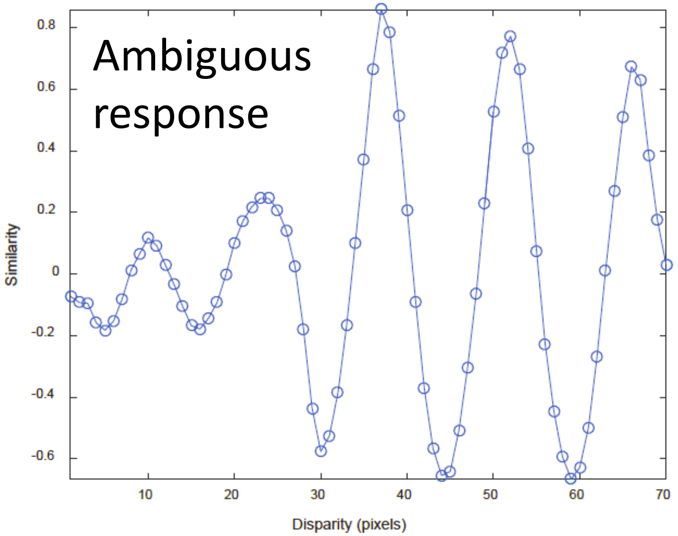
- Small window size W:
  - Details potentially better estimated
  - Noisy disparity
  - Fast(er) computation
- Large window size W:
  - Details potentially lost
  - Smooth disparity
  - Slow(er) computation

Images credit: Trym Vegard Haavardsholm

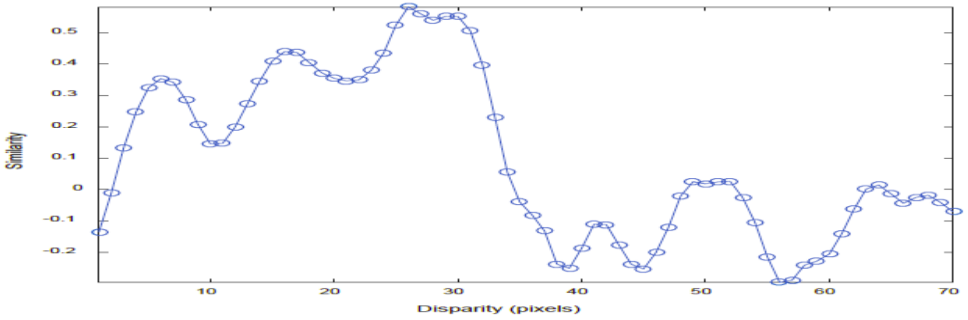
Disparity quality



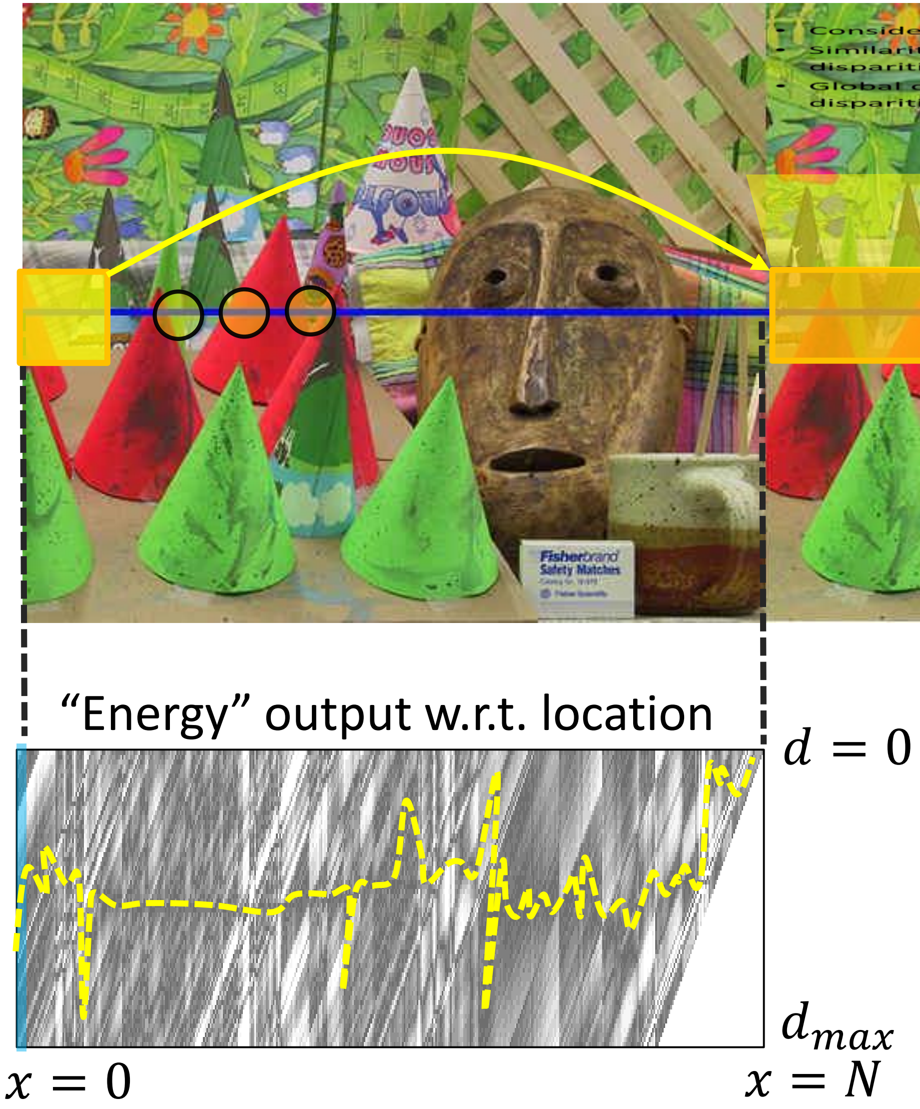




Weak response due to occlusion



Global disparity optimization



- Consider a single line (N pixels)
- Similarity scores* for different disparities for each pixel.
- Global cost of selecting disparities $d = (d_1 : d_N)$:

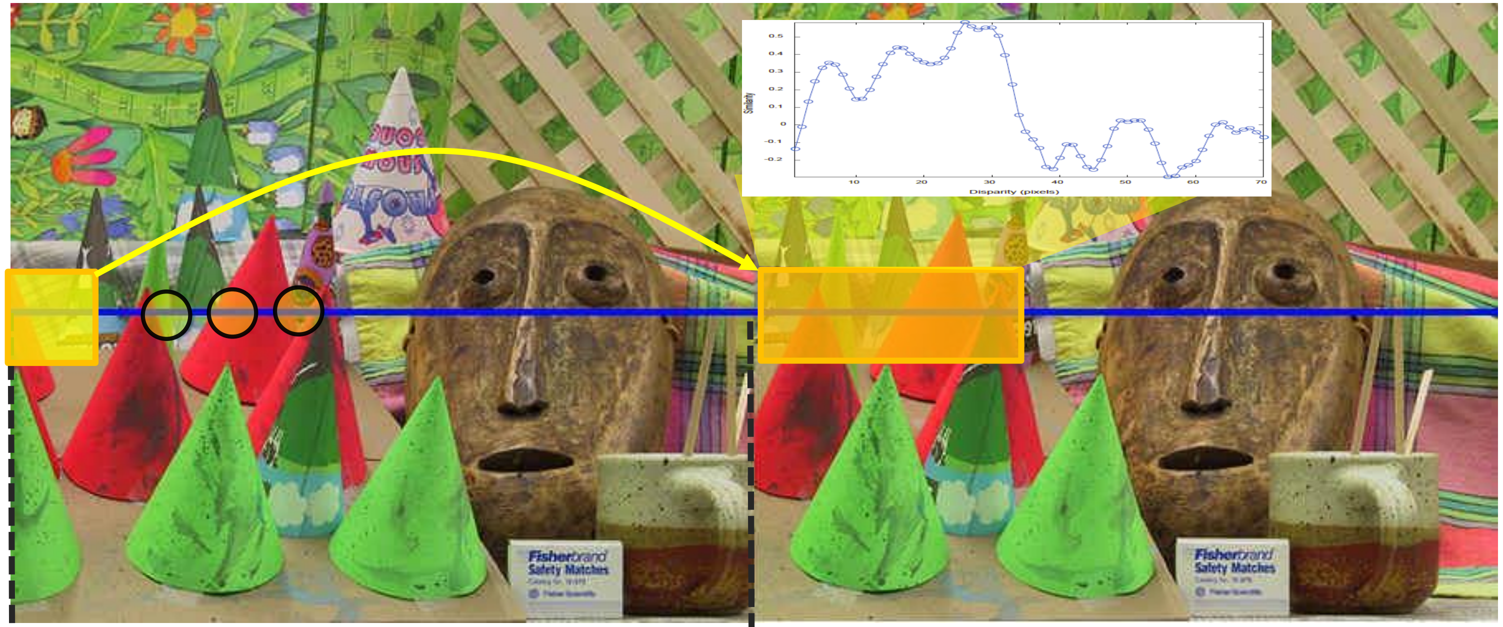
$$E(d_i) = E_{data}(d_i)$$

$$E_{data}(d_i) = e^{-similarity(d_i)}$$

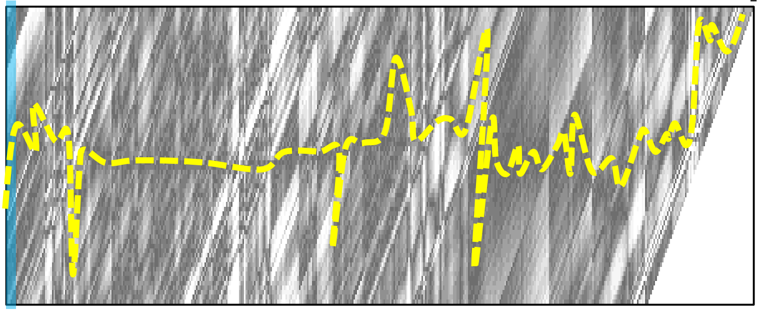
$$E(d) = \sum_i E_{data}(d_i)$$

*Similarity 0 means “no match”, 1 means “perfect match”

Global disparity optimization



Similarity output w.r.t. location



$d = 0$

d_{max}

$x = 0$ $x = N$

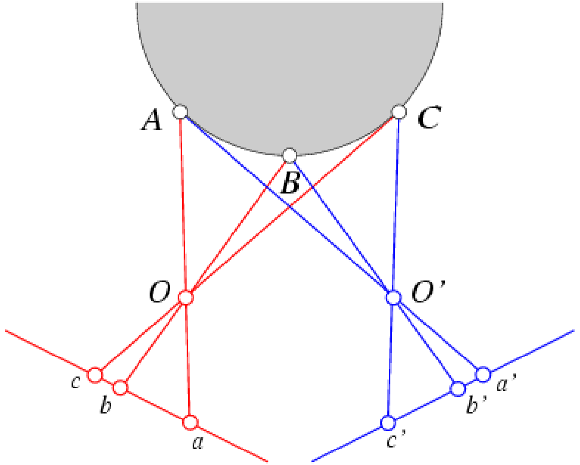
$$E(d_i) = E_{data}(d_i)$$

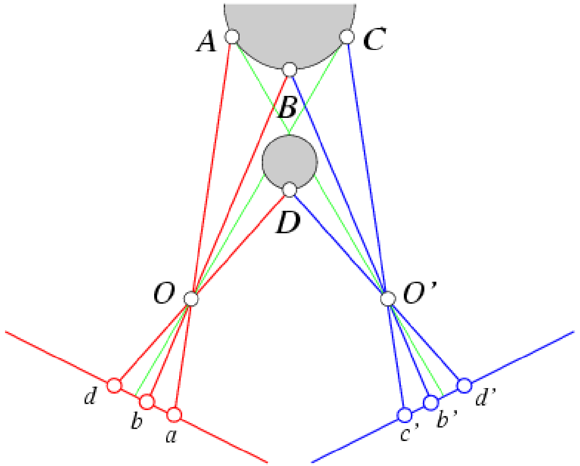
$$E_{data}(d_i) = e^{-similarity(d_i)}$$

- Disparity calculated independently at each pixel.
- Additional constraints can be imposed on the set of viable disparity estimates.

Disparity constraints Constraints:

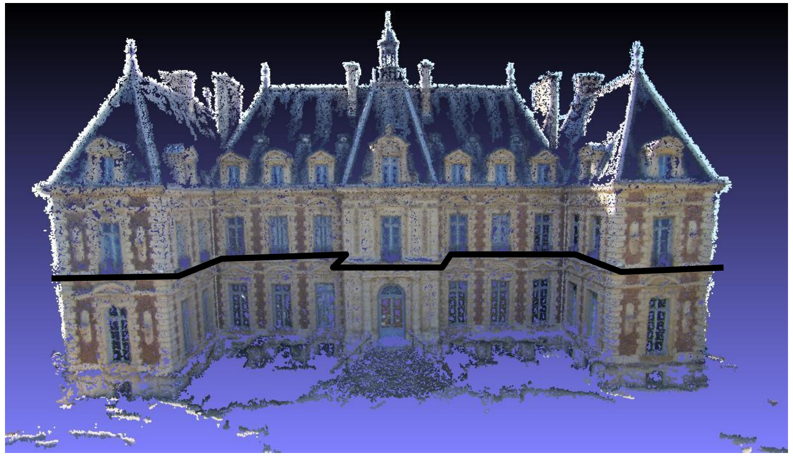
- Order: Points on a single surface appear in the same order in both views.

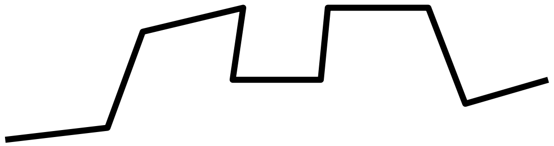




Order of points constraint violated

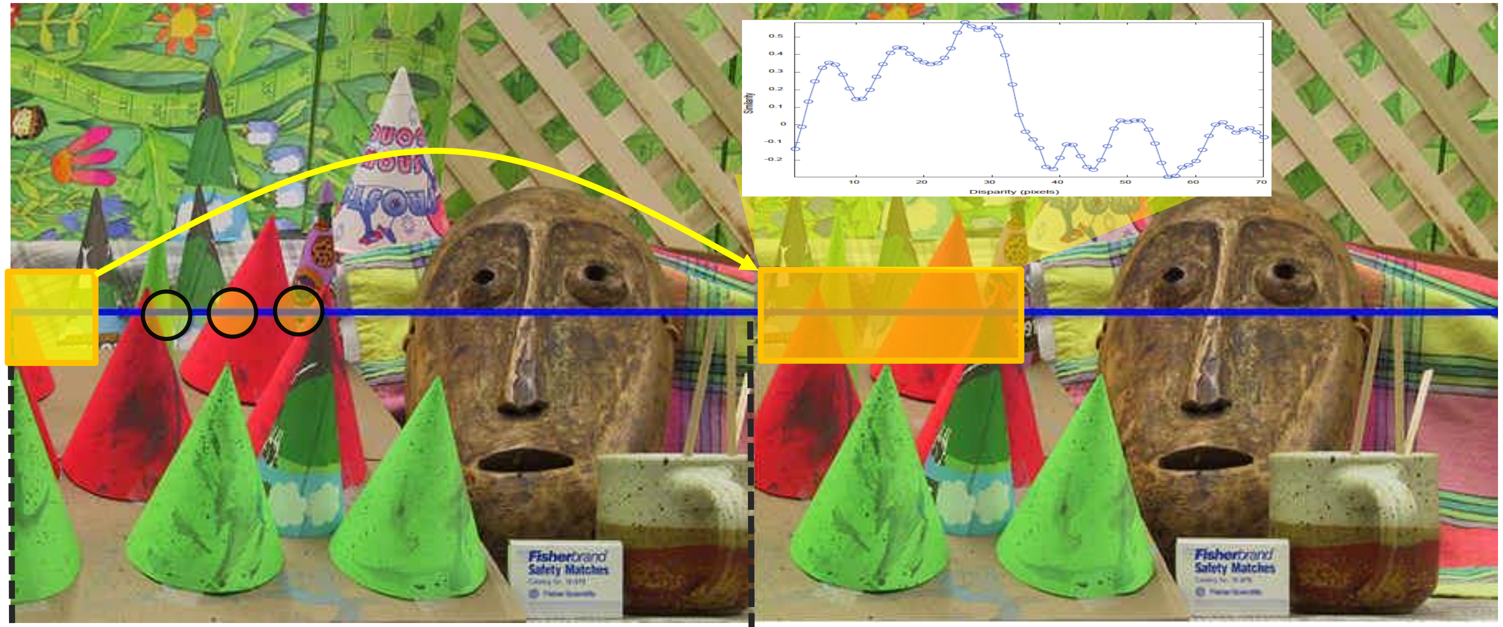
- Slow local depth change: smooth surfaces should result in smooth disparity.





depth

Global disparity optimization



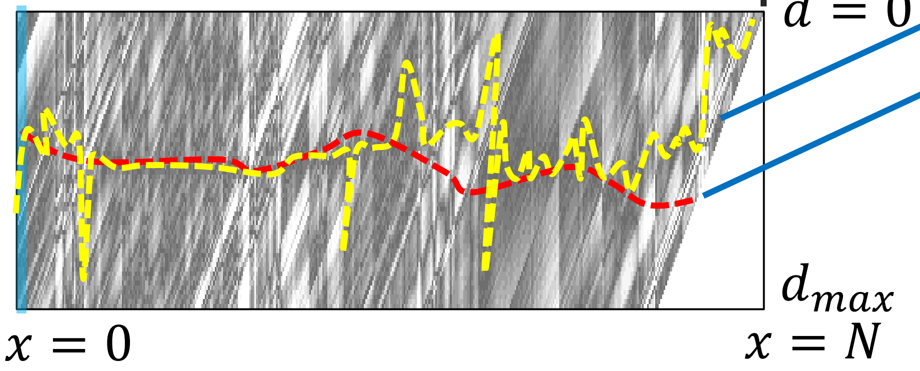
- Consider a single line (N pixels)
- Similarity scores for different disparities for each pixel.
- Global cost of selecting disparities $d = (d_1 : d_N)$:

$$E(d_i) = E_{data}(d_i) + \lambda E_S(d_i)$$

$$E_{data}(d_i) = e^{-similarity(d_i)}$$

Smoothness term that assigns a high cost if disparities change significantly between consecutive pixels

"Energy" output w.r.t. location

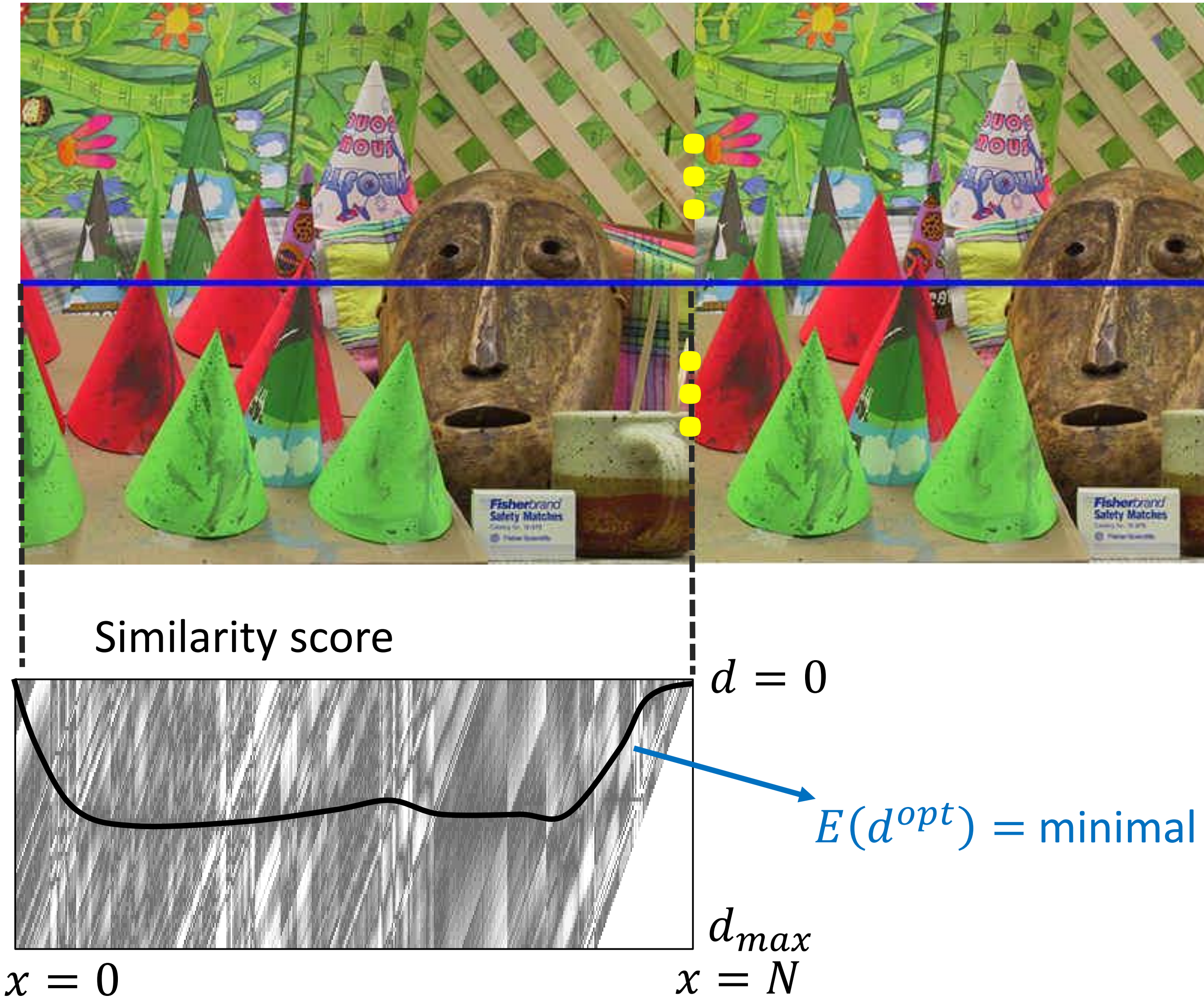


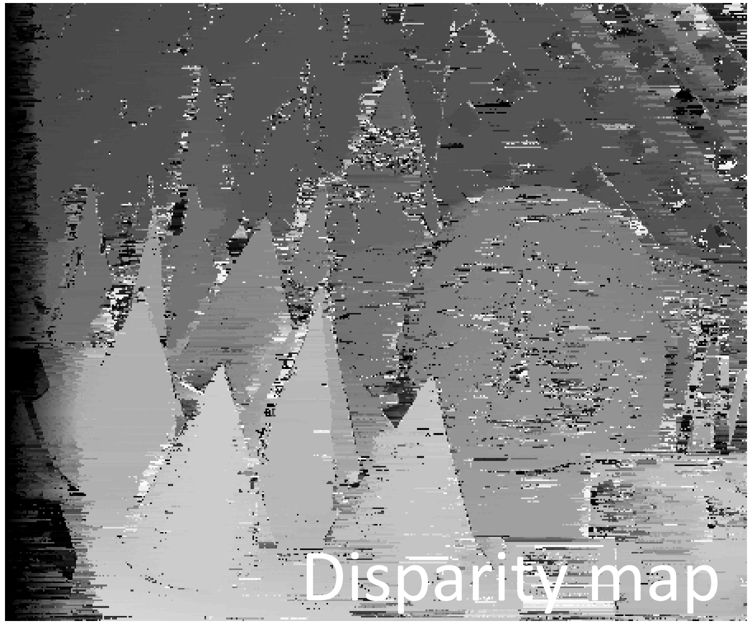
$E(d)$ = large

$E(d)$ = small

How to find d with globally minimal $E(d)$?

Global disparity optimization



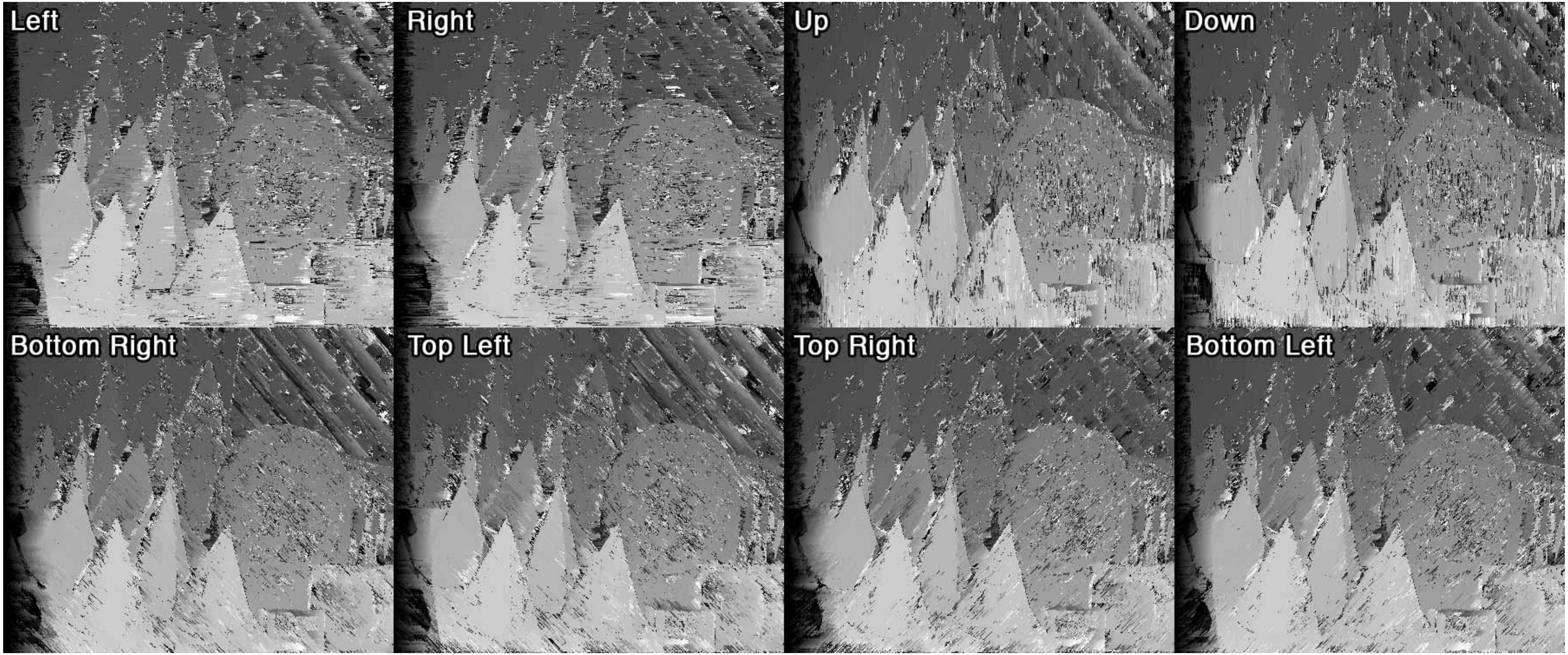


- Optimal sequence of disparities $\mathrm{d}^{\mathrm{opt}} = (d_1 : d_N)$ obtained by Viterbi algorithm (dynamic program).
- Apply independently to each line.

Cox, Hingorani, Rao, Maggs, "A Maximum Likelihood Stereo Algorithm," CVIU, Vol 63(3), 1996.

Semi global block matching (SGBM)

- Apply line-based optimization across several directions in the image …

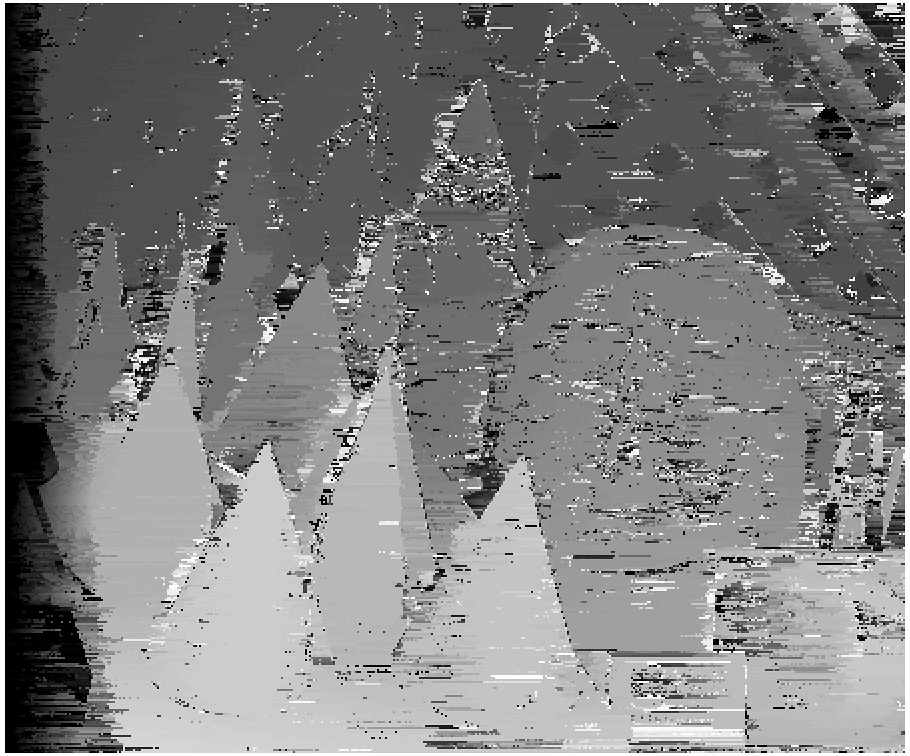


http://lunokhod.org/?p=1356

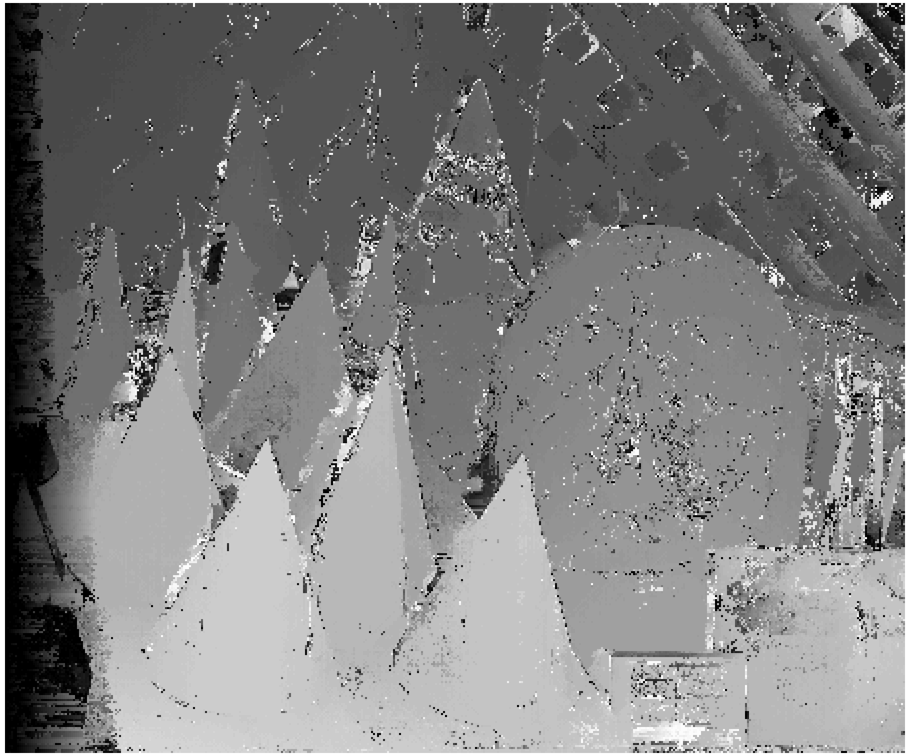
Semi global block matching (SGBM)

- … aggregate disparity energies from all direction-optimal assignments and take the disparity at each pixel that received a minimum energy.

Left-to-right line optimization



After aggregating 8-direction energies



Heiko Hirschmuller, “Stereo processing by semiglobal matching and mutual information”. *TPAMI*, 2008

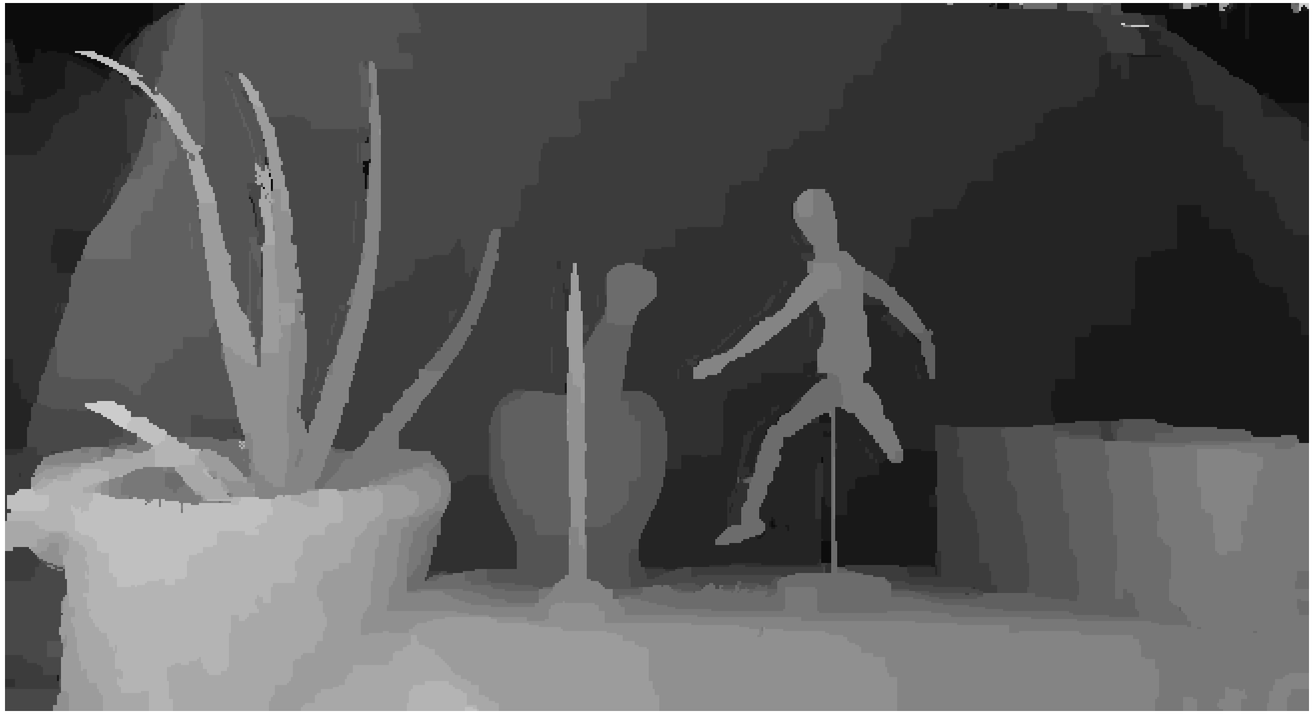
Application: View interpolation

Right image

Application: View interpolation

Left image

Application: View interpolation

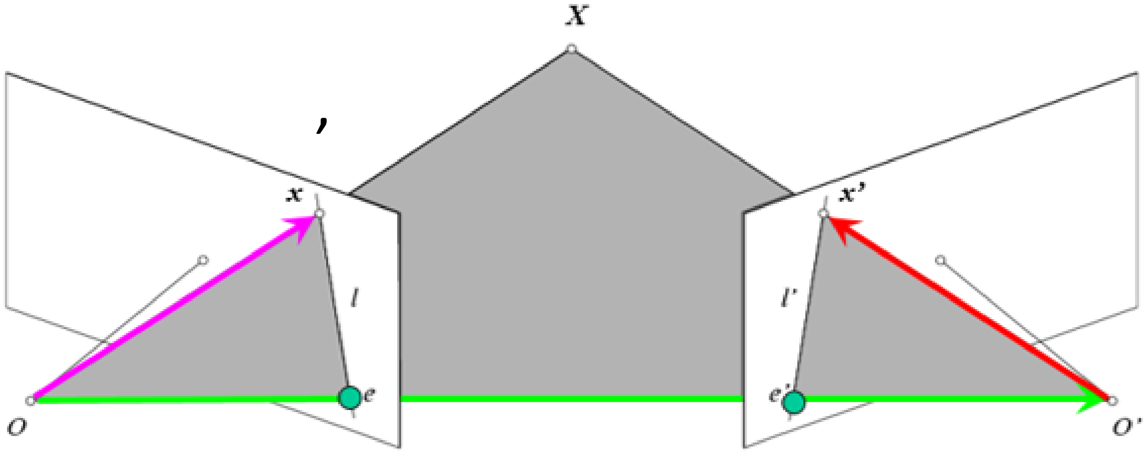


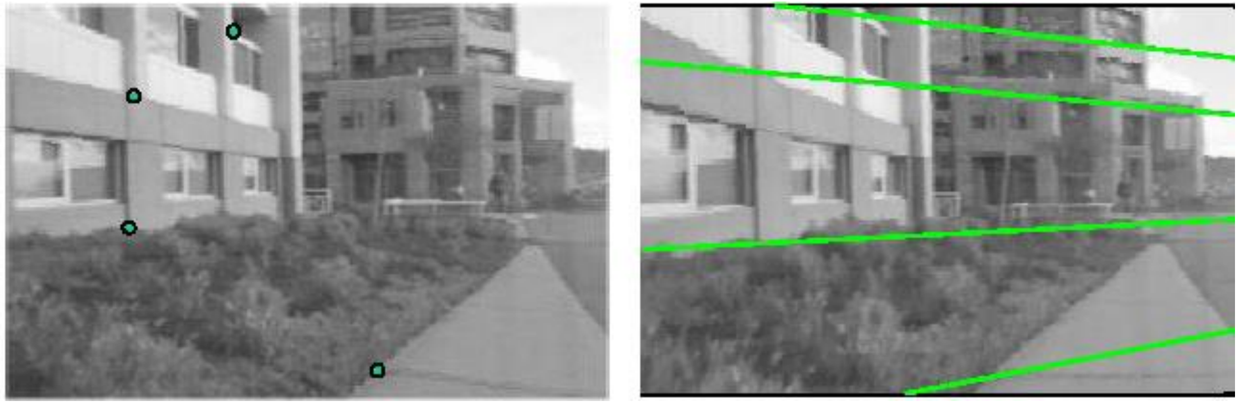
Disparity

Application: View interpolation

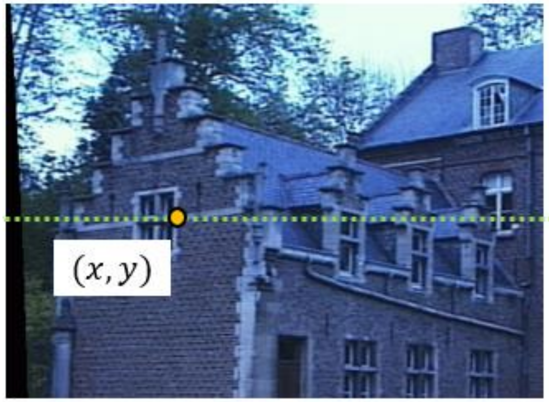
Previously at MP…

- A system of two or more cameras

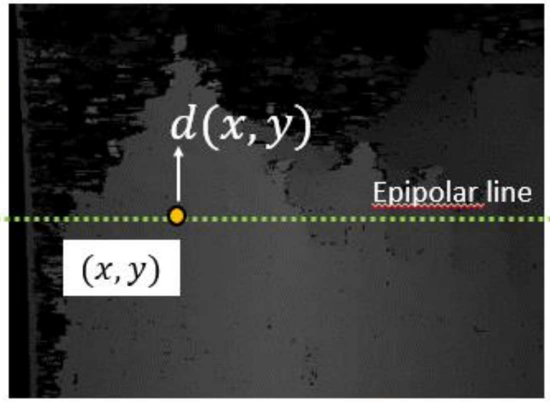




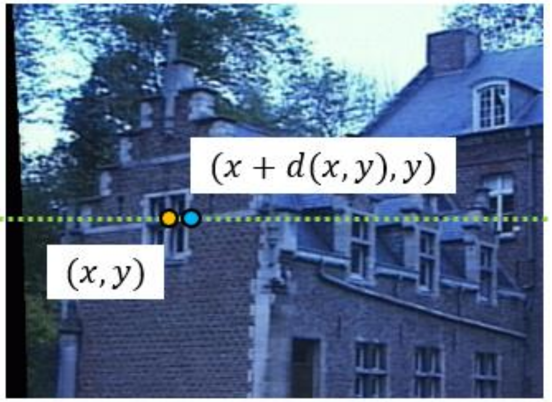
- A system of perfectly aligned cameras



Left image: I(x,y)

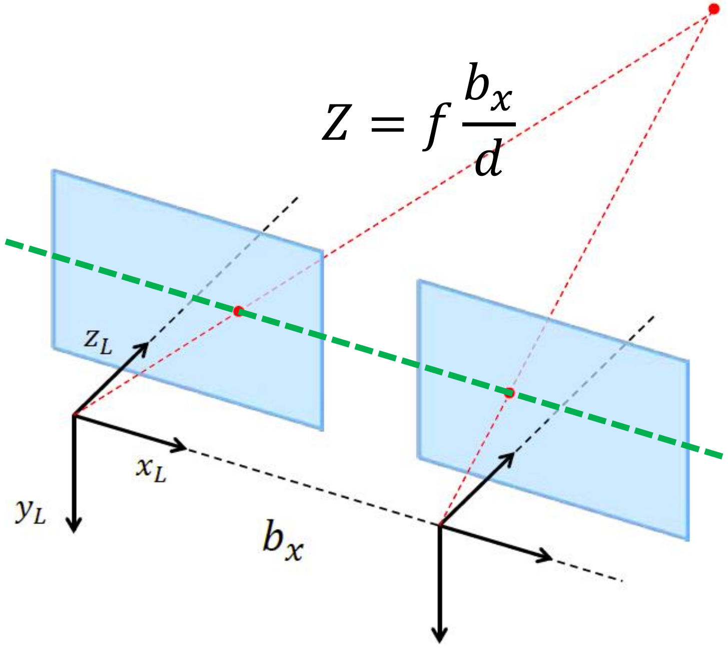


Disparity map d(x,y)

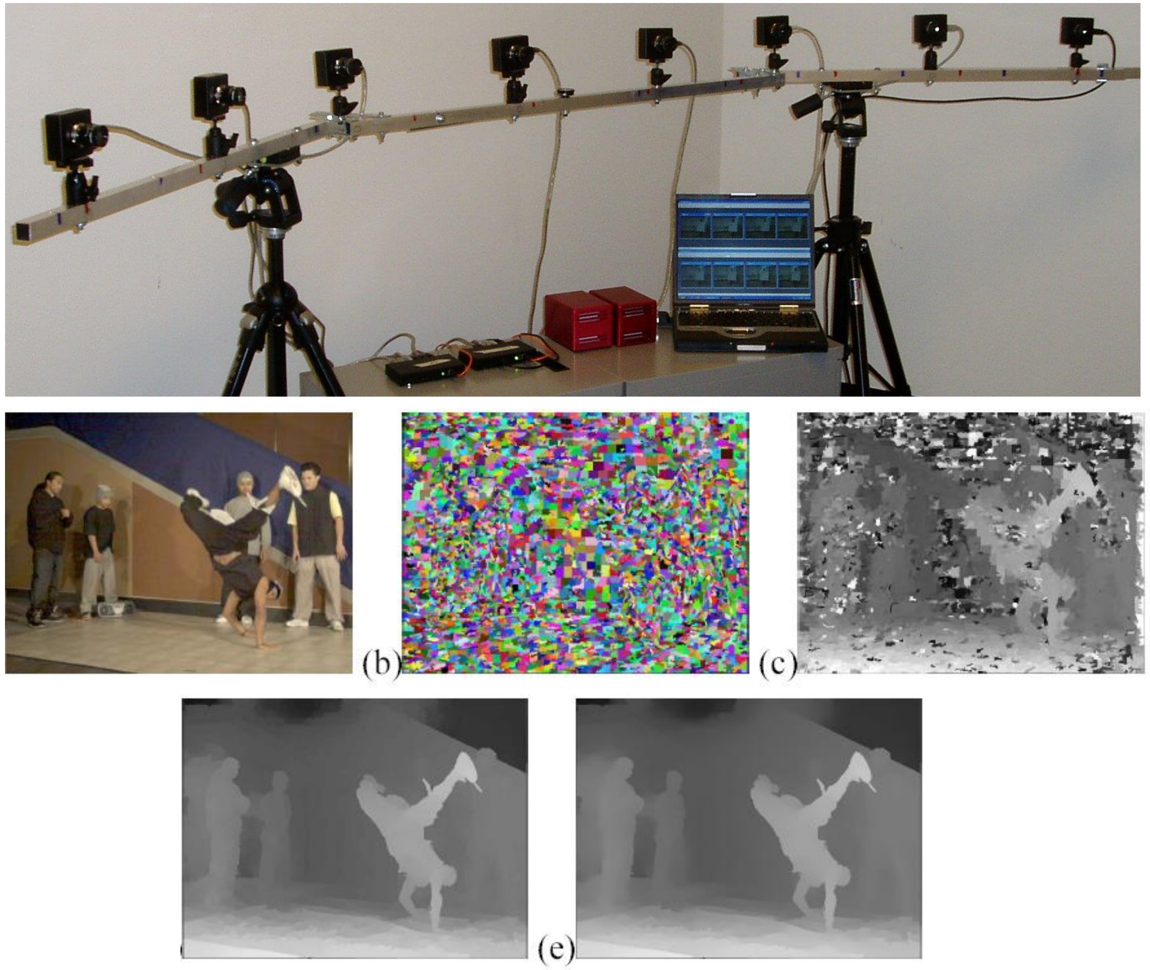


Right image: I´(x´,y´)

$$Z = f\,\frac{b_x}{d}$$



Video view interpolation

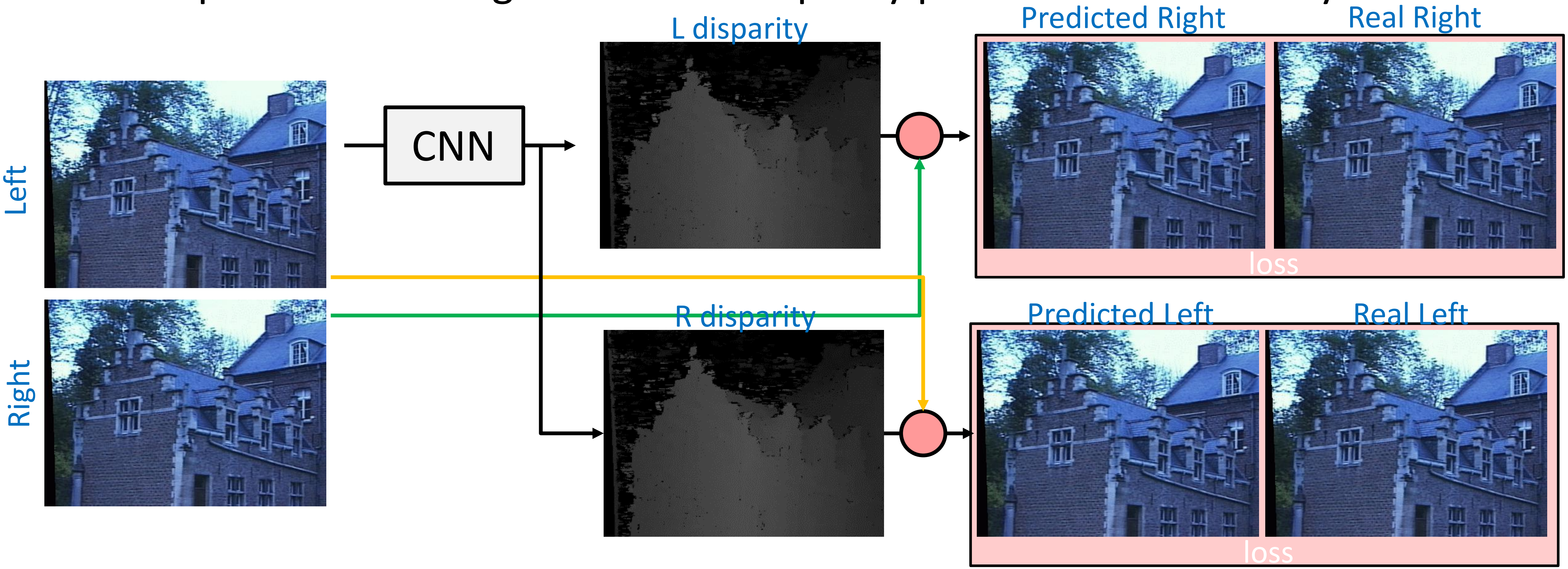


L. Zitnick et al, High-quality video view interpolation using a layered representation, SIGGRAPH 2004

http://research.microsoft.com/IVM/VVV/

Recent works on CNN-based mono-depth

- Train a CNN to predict depth based on a *single* image
- Unsupervised training: use stereo disparity prediction consistency



Godard et al., [Unsupervised Monocular Depth Estimation with Left-Right Consistency](), CVPR2017

Monodepth 2

Godard et al., [Digging Into Self-Supervised Monocular Depth Estimation](), ICCV2019 [GIT]

Reconstruction by a moving camera

- If a camera is moving freely, the "stereo system" cannot be pre-calibrated (except from matrix $\boldsymbol{K}$, that is)
- Actually, we are dealing with multiple "cameras"

Machine perception

STRUCTURE FROM MOTION (SFM)
A BRIEF OVERVIEW

The aim of SFM

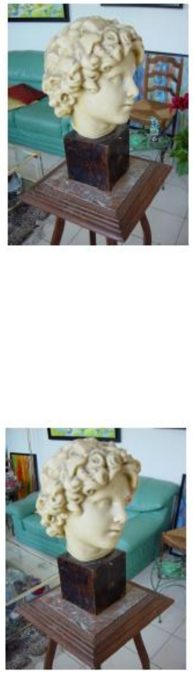
- Given several images of same scene
- Reconstruct the camera positions **and** reconstruct the 3D scene

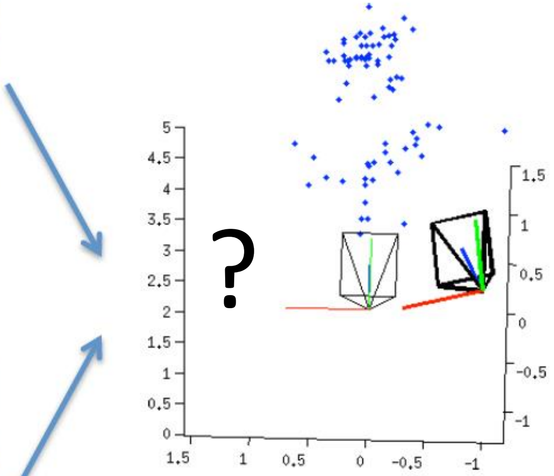
- Assume a partially-calibrated case, in which the camera calibration matrices $\boldsymbol{K}$ are known.

Xiao, J. [Multiview 3D Reconstruction for Dummies]

SFM pipeline (calibrated cameras)

- Triangulation requires:
  - Knowing correspondences.
  - Knowing projection matrices $\{\boldsymbol{P}_j\}_{j=1:M}$ for all M cameras.
- Computing projection matrix $\boldsymbol{P}_j$ requires: $\boldsymbol{K}_j, \boldsymbol{R}_j, \boldsymbol{t}_j$
- Matrices $\boldsymbol{R}_j, \boldsymbol{t}_j$ can be computed from essential matrix $\boldsymbol{E}_j$ between first and j-th view.
- Matrix $\boldsymbol{E}_j$ can be computed from fundamental matrix $\boldsymbol{F}_j$ and calibration matrices $\boldsymbol{K}_1$and $\boldsymbol{K}_j$.





$\mathbf{E} = \mathbf{T}_{\times}\mathbf{R}$

$F = K^{-T}EK'^{-1}$

SFM pipeline

- For simplicity, consider a pair of views with known calibration matrices $\boldsymbol{K_1}, \boldsymbol{K_2}$
- Actually, if the camera is moving, the calibration matrices are equal $\boldsymbol{K} = \boldsymbol{K_1} = \boldsymbol{K_2}$
- The approach generalizes to multiple views

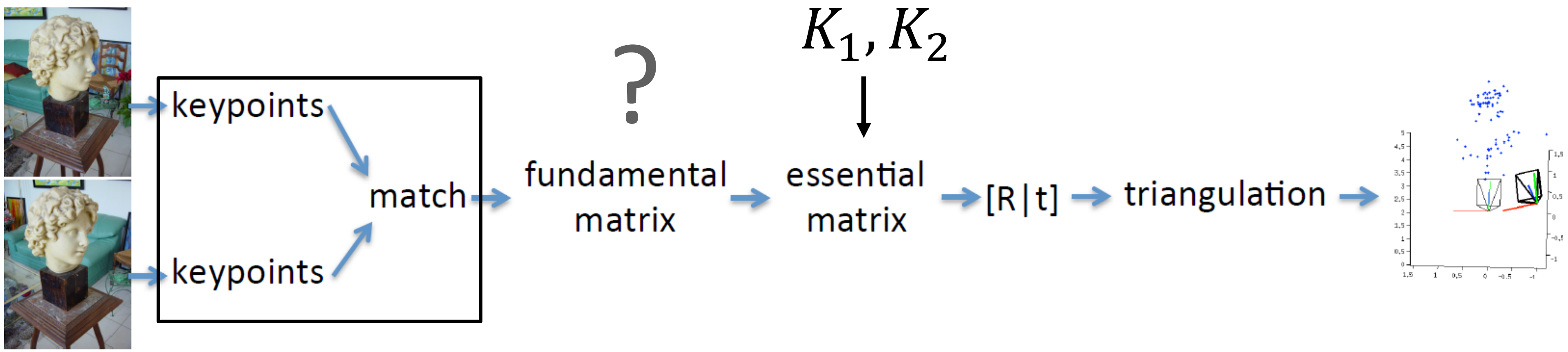
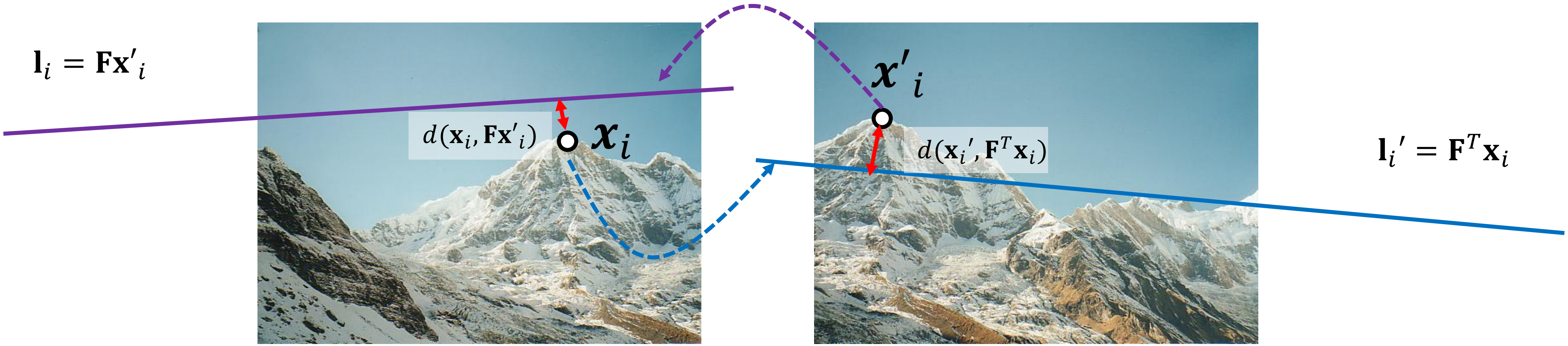


Image from: Xiao, J. [Multiview 3D Reconstruction for Dummies]

Fundamental matrix estimation

Also known as *weak calibration*.

- Assume known correspondences $\{x_i\}_{i=1:N}, \{x'_i\}_{i=1:N}$
- Estimate F that minimizes reprojection errors $\epsilon(\mathbf{F})$



$$\epsilon(\mathbf{F}) = \frac{1}{N}\sum_{i=1}^{N}\left(d^2(\mathbf{x}_i, \mathbf{F}\mathbf{x}'_i) + d^2(\mathbf{x}_i', \mathbf{F}^T\mathbf{x}_i)\right)$$

- Nonlinear optimization (Levenberg-Marquardt), requires good initial estimate.
- Usually initialized by 8-point algorithm (described next).

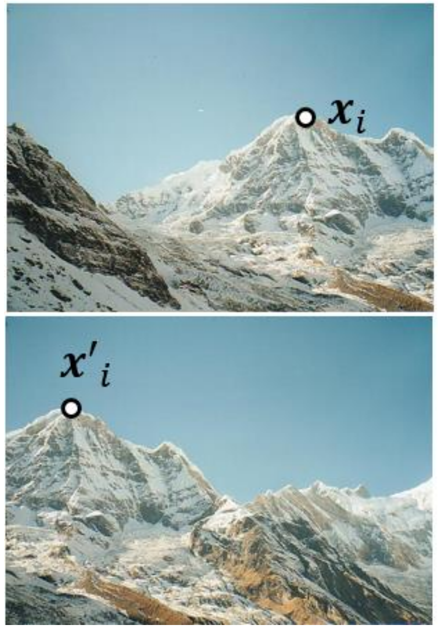
Fundamental matrix estimation: Eight-point algorithm

Coordinates of a pair of corresponding points: $\boldsymbol{x} = (u, v, 1)^T$, $\boldsymbol{x}' = (u', v', 1)^T$

Epipolar constraint: $\boldsymbol{x}^T \boldsymbol{F} \boldsymbol{x}' = \boldsymbol{0}$

$$(u, v, 1)\begin{pmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{pmatrix}\begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix} = 0 \Rightarrow (uu', uv', u, vu', vv', v, u', v', 1)\begin{pmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{pmatrix} = 0$$

$$F_{33} + F_{13}u + F_{31}u' + F_{23}v + F_{32}v' + F_{11}uu' + F_{12}uv' + F_{21}u'v + F_{22}vv' = 0$$



(one equation per correspondence – require 8)

$$\begin{bmatrix} u_1u_1' & u_1v_1' & u_1 & v_1u_1' & v_1v_1' & v_1 & u_1' & v_1' & 1 \\ u_2u_2' & u_2v_2' & u_2 & v_2u_2' & v_2v_2' & v_2 & u_2' & v_2' & 1 \\ u_3u_3' & u_3v_3' & u_3 & v_3u_3' & v_3v_3' & v_3 & u_3' & v_3' & 1 \\ u_4u_4' & u_4v_4' & u_4 & v_4u_4' & v_4v_4' & v_4 & u_4' & v_4' & 1 \\ u_5u_5' & u_5v_5' & u_5 & v_5u_5' & v_5v_5' & v_5 & u_5' & v_5' & 1 \\ u_6u_6' & u_6v_6' & u_6 & v_6u_6' & v_6v_6' & v_6 & u_6' & v_6' & 1 \\ u_7u_7' & u_7v_7' & u_7 & v_7u_7' & v_7v_7' & v_7 & u_7' & v_7' & 1 \\ u_8u_8' & u_8v_8' & u_8 & v_8u_8' & v_8v_8' & v_8 & u_8' & v_8' & 1 \end{bmatrix}\begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Homogeneous system!

$$\mathbf{Af} = \mathbf{0}$$

$\Rightarrow$

Minimize:

$$\sum_{i=1}^{N} (x_i^T F x'_i)^2$$

with constraint

$$||F||^2 = 1$$

$F \leftarrow$ last eigenvector(A)

Normalized 8-point algorithm

1. *Precondition:* Center image points, and scale such that the standard deviation becomes $\sqrt{2}$ pixels.
   - $\widetilde{\boldsymbol{x}} = \boldsymbol{T}\boldsymbol{x},\ \widetilde{\boldsymbol{x}}' = \boldsymbol{T}'\boldsymbol{x}'$
2. Apply 8-point algorithm to calculate $\widetilde{\boldsymbol{F}}$ from the preconditioned points.
3. Enforce rank=2

   (decompose $\widetilde{\boldsymbol{F}}$, by SVD set the smallest eigenvalue to zero and reconstruct $\widetilde{\boldsymbol{F}}$):

$$F \overset{\text{SVD}}{=} \mathbf{U}\mathbf{D}\mathbf{V}^T = \mathbf{U}\begin{bmatrix} d_{11} & & \\ & d_{22} & \\ & & d_{33} \end{bmatrix}\begin{bmatrix} v_{11} & \cdots & v_{13} \\ \vdots & \ddots & \vdots \\ v_{31} & \cdots & v_{33} \end{bmatrix}^T$$

Set d_{33}=0 and reconstruct F:

$$F = \mathbf{U}\begin{bmatrix} d_{11} & & \\ & d_{22} & \\ & & 0 \end{bmatrix}V^T$$

4. Transform the fundamental matrix back to original units:

   Let T and T' be the transformations used to precondition the points in each image separately. Then the fundamental matrix equals $\mathbf{F} = \boldsymbol{T}'^T\widetilde{\boldsymbol{F}}\boldsymbol{T}$.

Fundamental matrix estimation

- In general, the correspondences are unknown
  - Jointly find the fundamental matrix F *AND* the correspondences!
    (pairs across two views (u’,v’) ↔ (u,v)).

- Approach
  1. Find keypoints in each image
  2. Calculate possible matches (potential matches)
  3. Robustly estimate the epipolar geometry by RANSAC

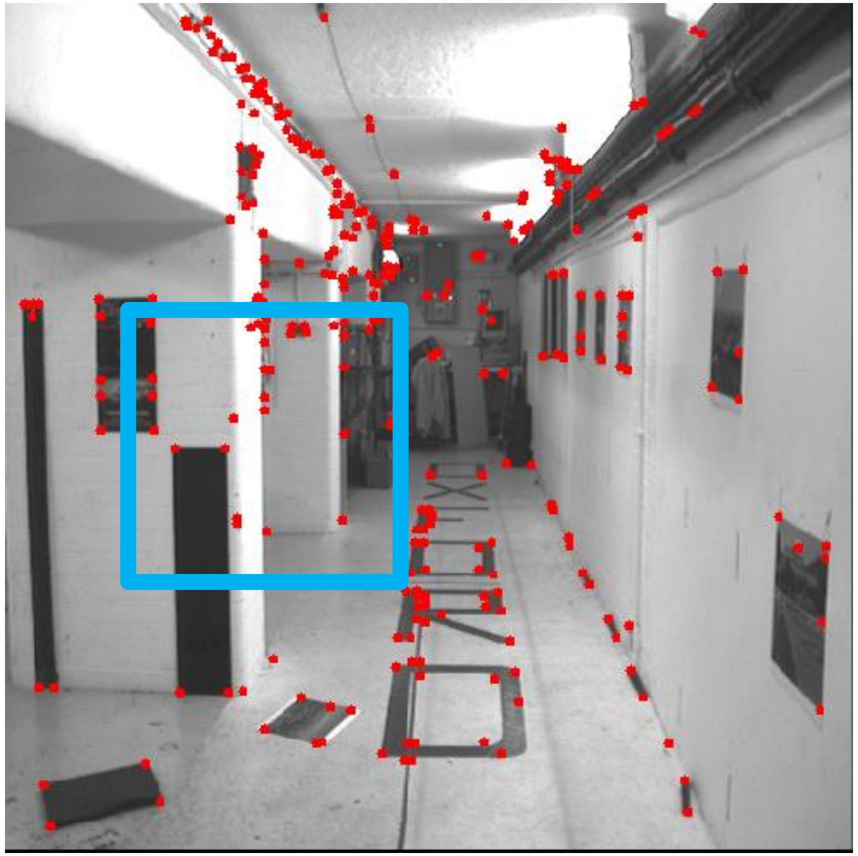
Slide credit: Kristen Grauman

Fundamental matrix estimation

1. Find key-points (eg., Harris corners)

Fundamental matrix estimation

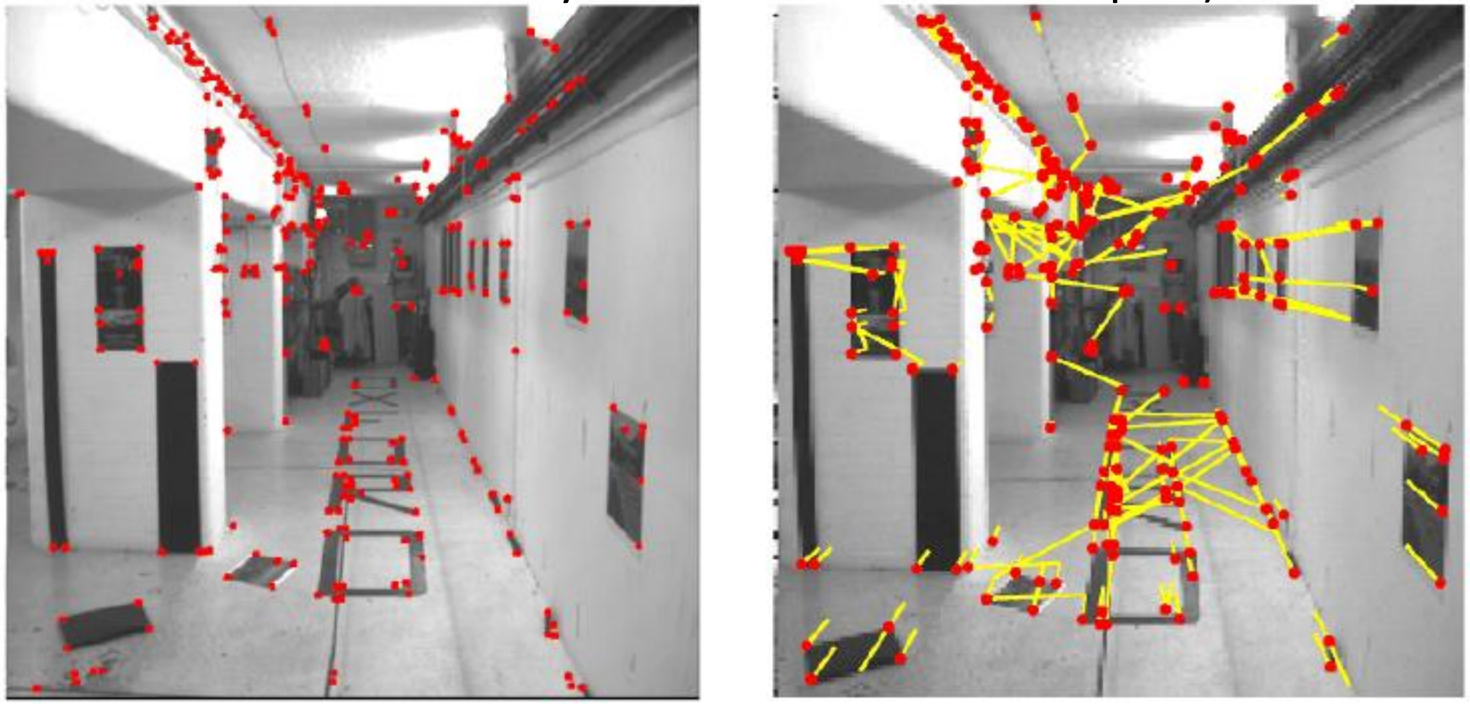
2. Find correspondence using proximity constraints



Fundamental matrix estimation

- Filter the correspondences by visual similarity

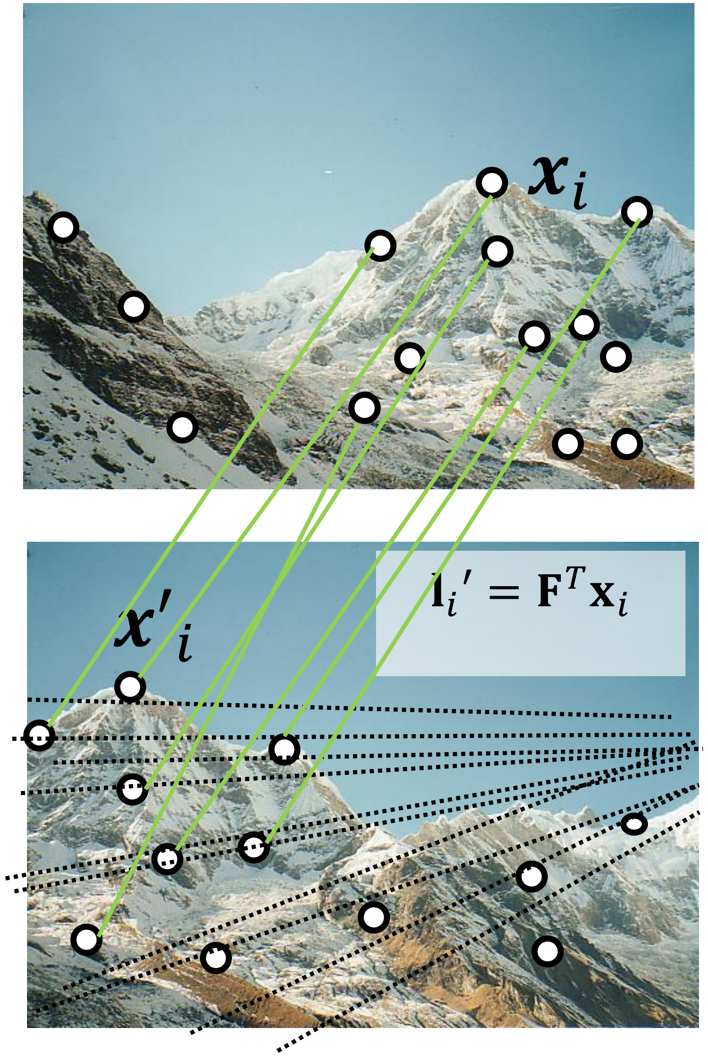
  (e.g.,using normalized cross correlation or by a more advanced descriptor)



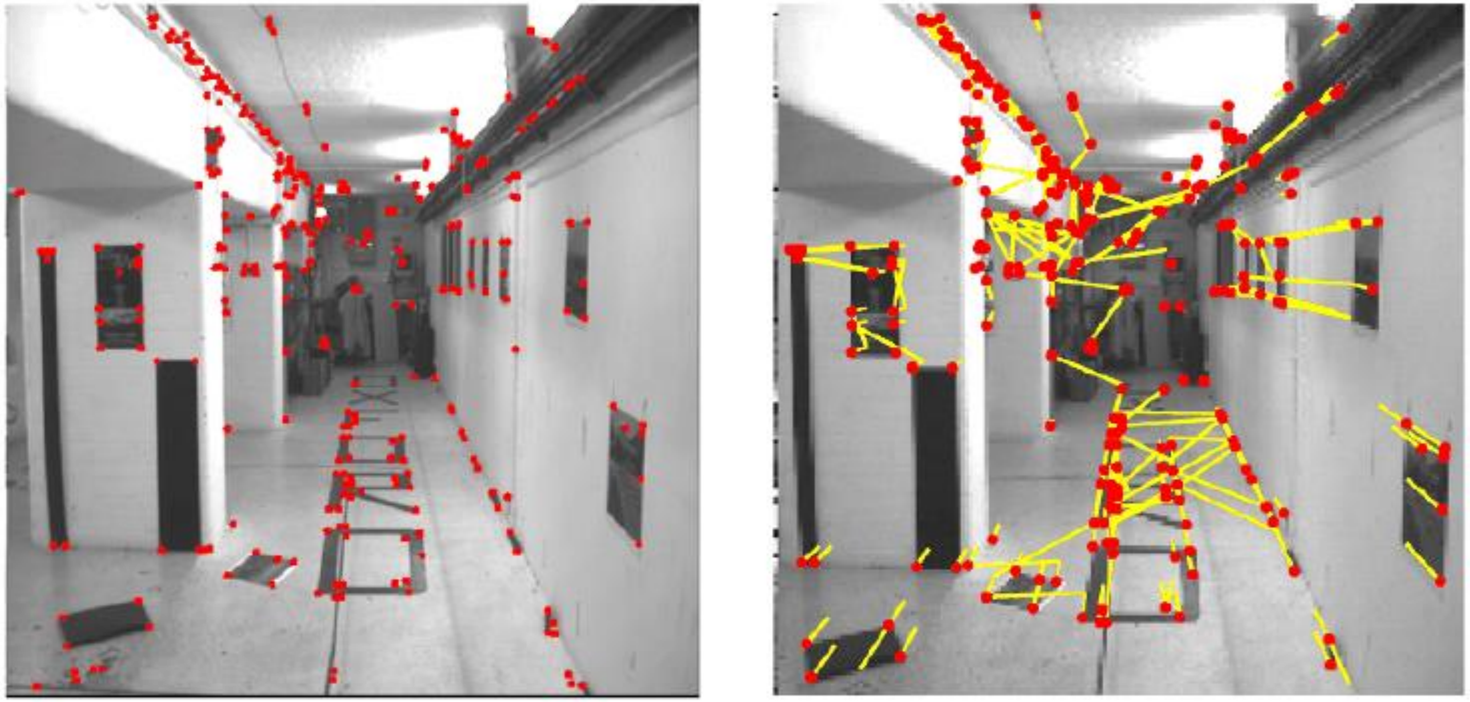
- Many wrong matches (10-50%), but enough to compute F

RANSAC to robustly estimate F

- Randomly select a set of 8 correspondences
- Calculate F using these correspondences
  - This gives the epipolar constraint!
- Estimate how many correspondences support F:
  - Apply the estimated fundamental matrix to all points in image I_1 and compute their epipolar lines in image I_2.
  - Number of inliers: points in I_2 that lie close to their epipolar lines calculated from $\boldsymbol{F}$ and corresponding points from I_1.
- Choose $\boldsymbol{F}$ with maximal support (#inliers)



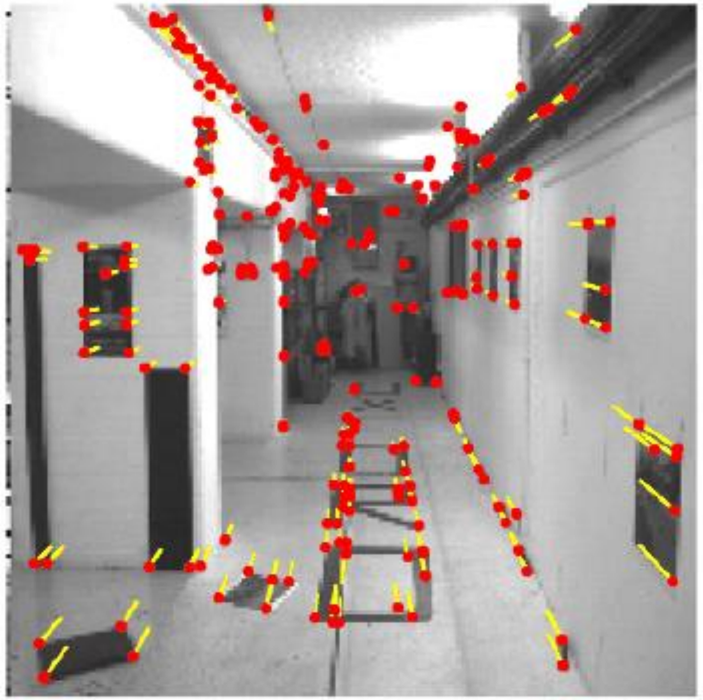
Input correspondences



- Many wrong matches (10-50%), but enough to compute F

Pruned correspondences

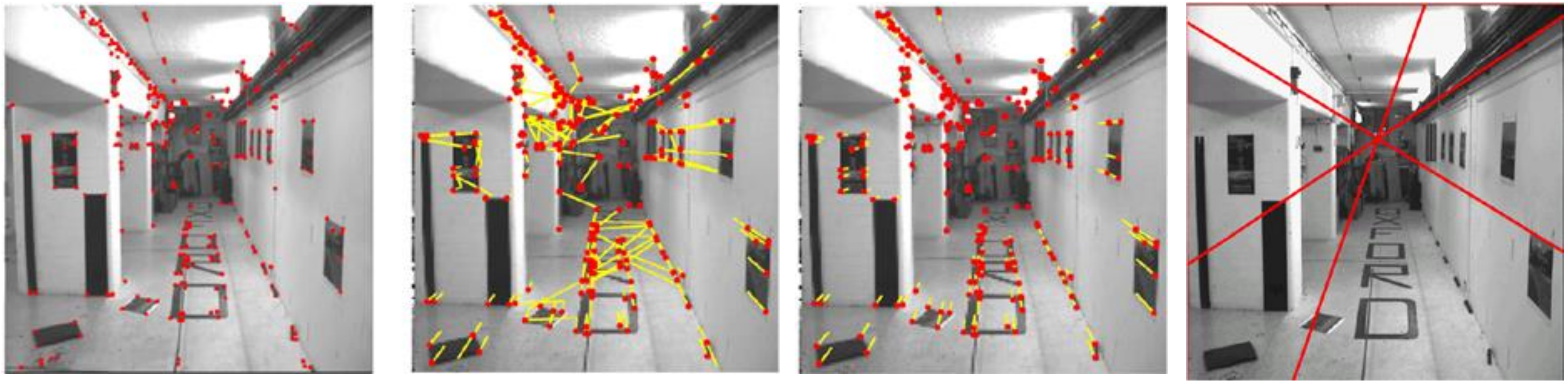
- Correspondences consistent with the epipolar constraint



Epipolar constraint visualized

Summary

- Robust estimation of F



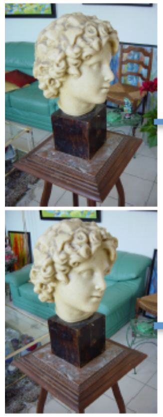
- Improve by a nonlinear optimization of the cost function w.r.t. $\boldsymbol{F}$ using inliers only:

$$\epsilon(\mathbf{F}) = \frac{1}{N}\sum_{i=1}^{N}\left(d^2(\mathbf{x}_i, \mathbf{F}\mathbf{x'}_i) + d^2(\mathbf{x}_i', \mathbf{F}^T\mathbf{x}_i)\right)$$

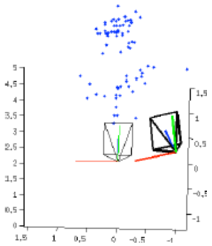
SFM pipeline

- Multiple-view SFM

Nice video (without last part – dense reconstruction)
https://www.youtube.com/watch?v=i7ierVkXYa8



keypoints → match
keypoints → match
match → fundamental matrix → essential matrix → [R|t] → triangulation →



- 8-point algorithm initializes a nonlinear optimization of reprojection errors (bundle adjustment):

$$\{\mathrm{R}_\mathrm{i}^*, \mathrm{t}_\mathrm{i}^*, \mathrm{X}_\mathrm{ij}\} = \mathrm{argmin} \sum_i \sum_j \left(\mathbf{x}_{ij} - \mathbf{K}_i[\mathbf{R}_i|\mathbf{t}_i]\mathbf{X}_{ij}\right)^2$$

- For an excellent overview of SFM see:

Xiao, J. [Multiview 3D Reconstruction for Dummies]

Try SFM at home

- Bundler: Structure from Motion (SfM) for Unordered Image Collections

Software written by Noah Snavely
Download Bundler from the bundler_sfm repository on GitHub

| What is Bundler? | Downloading Bundler | Documentation | References | Links |

What is Bundler?

Bundler is a structure-from-motion (SfM) system for unordered image collections (for instance, images from the Internet) written in C and C++. An earlier version of this SfM system was used in the Photo Tourism project. **For structure-from-motion datasets, please see the BigSFM page.**

Bundler takes a set of images, image features, and image matches as input, and produces a 3D reconstruction of camera and (sparse) scene geometry as output. The system reconstructs the scene incrementally, a few images at a time, using a modified version of the Sparse Bundle Adjustment package of Lourakis and Argyros as the underlying optimization engine. Bundler has been successfully run on many Internet photo collections, as well as more structured collections.

The Bundler source distribution also contains potentially userful implementations of several computer vision algorithms, including:

- F-matrix estimation
- Calibrated 5-point relative pose
- Triangulation of multiple rays

PhotoTurism video on YouTube

https://www.youtube.com/watch?v=5rYyB4pKPRo

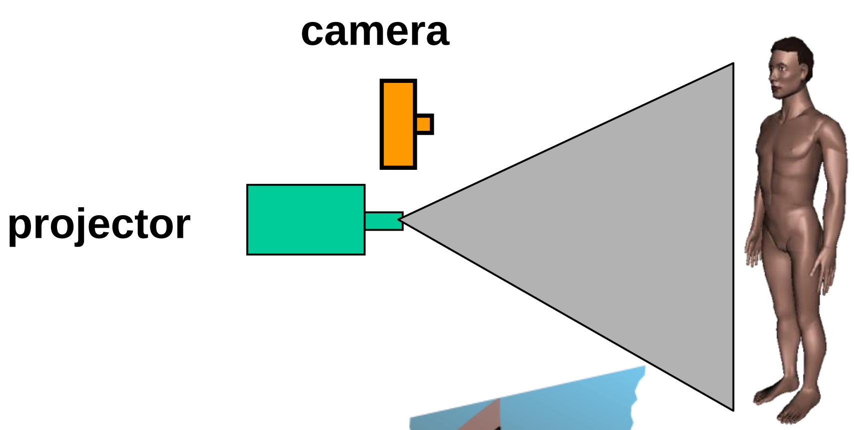
Machine perception

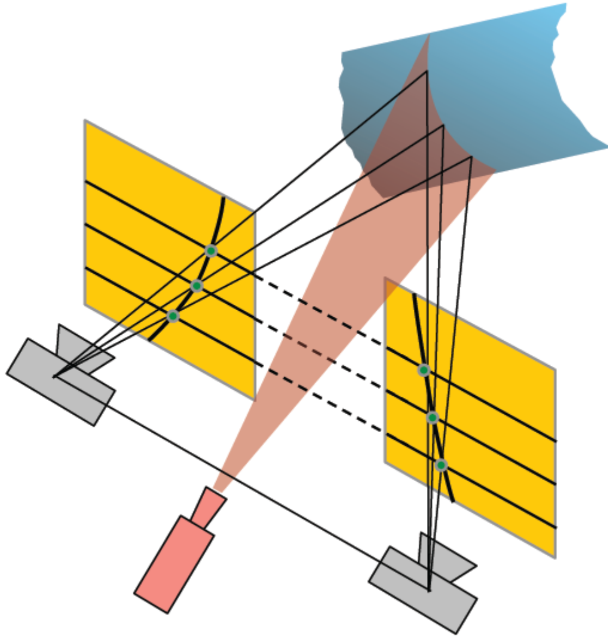
ACTIVE STEREO

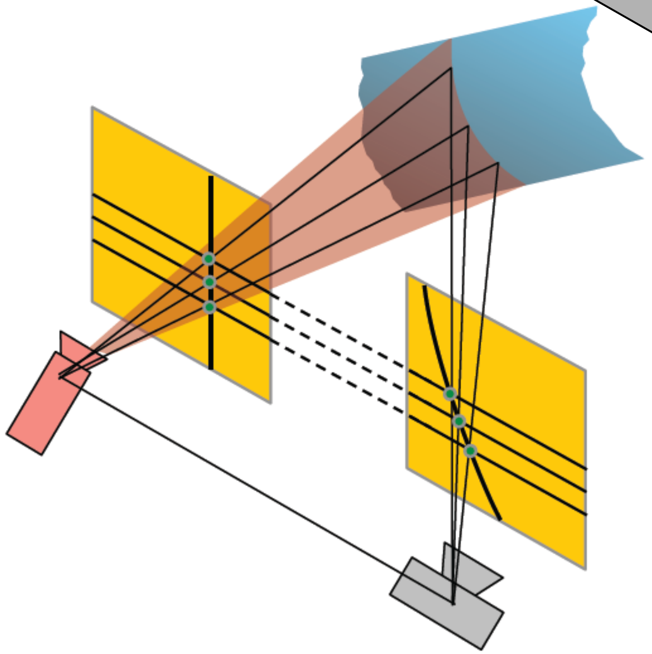
Structured light stereo

- Idea: project „structured“ light patterns over the object
  - Correspondence problem simplifies
  - Can use only a single camera

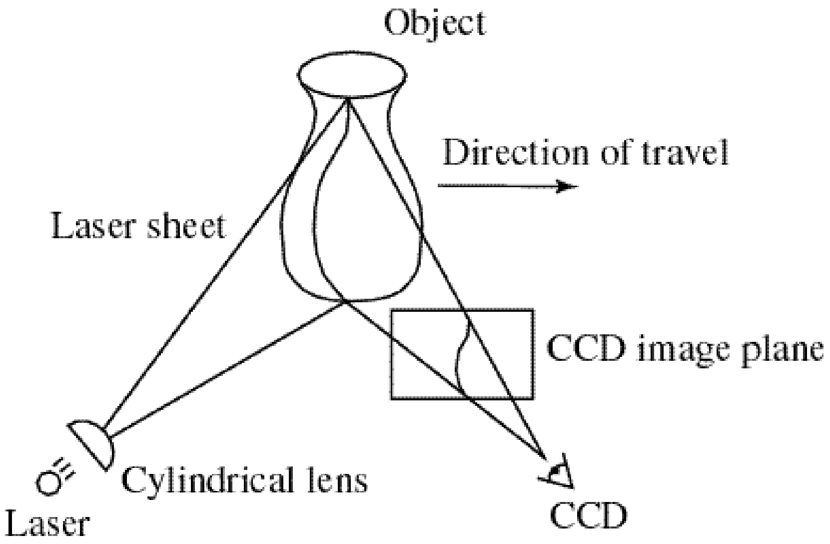
  Epipolar geometry still holds!

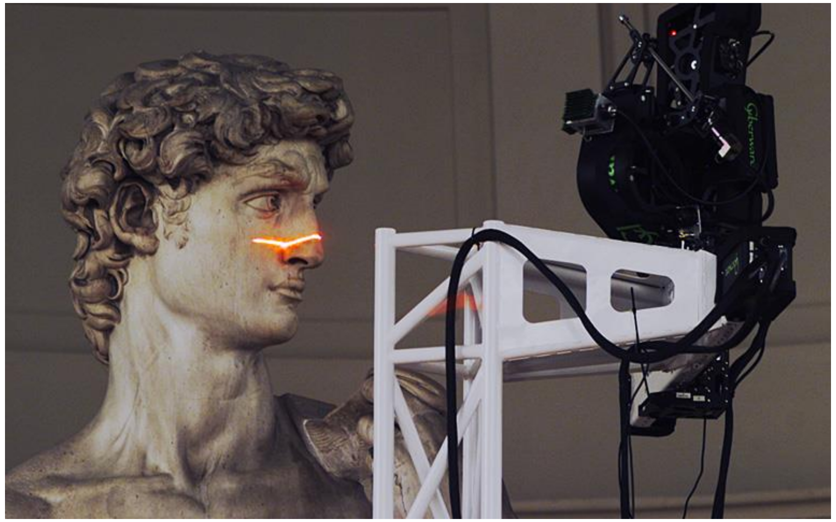






Laser scanning





Digital Michelangelo Project
http://graphics.stanford.edu/projects/mich/

- Optical triangulation
  - Project a laser light plane
  - Move over an object (the motion has to be accurately measured!)
  - Very precise way to scan using structured light.

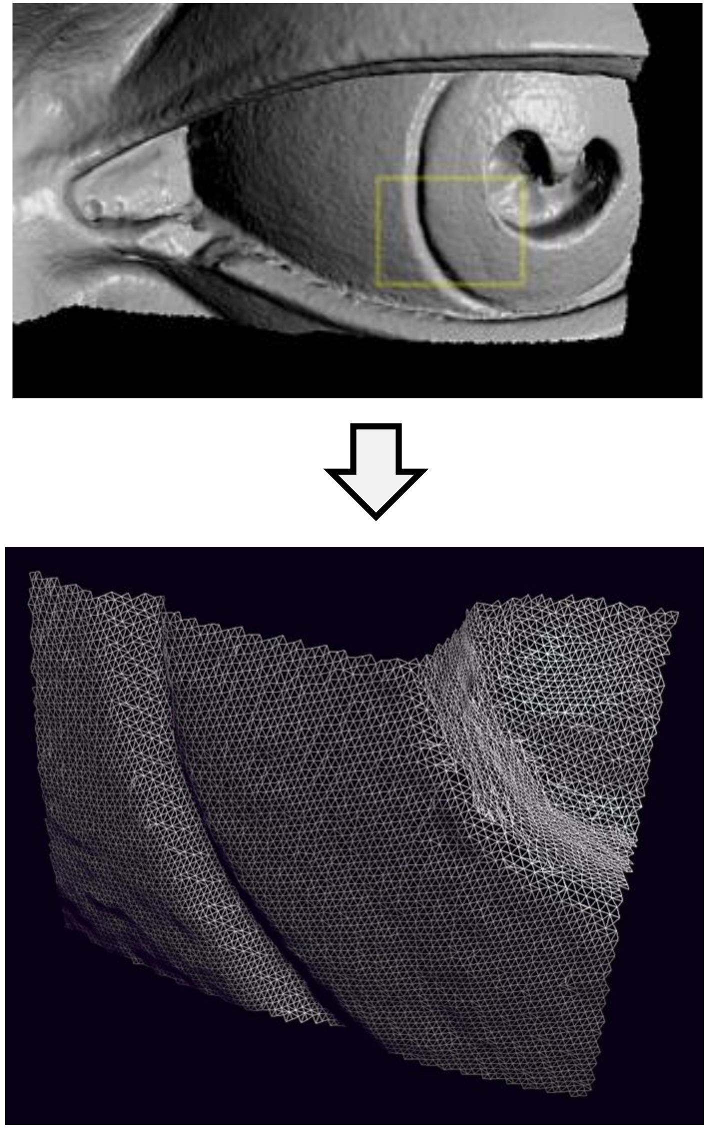
Obtained models

Michelangelo's David (Florence)

The Digital Michelangelo Project, Levoy et al.

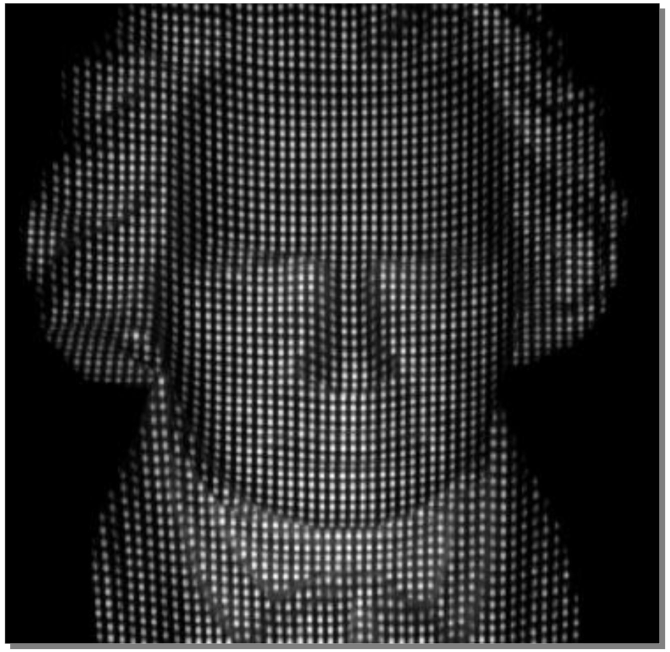
Obtained models

The Digital Michelangelo Project, Levoy et al.



Multi-band triangulation

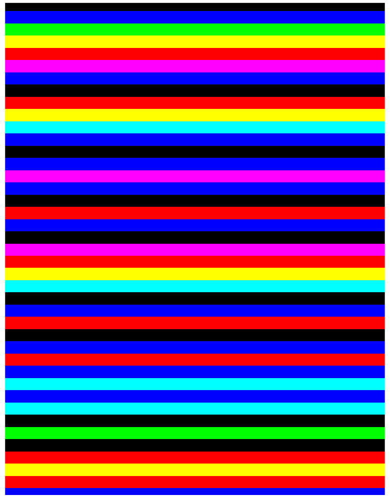
- Project multiple bands to speedup scanning
- But, which pixels belong to which band?
- Answer #1: Assume smooth surface

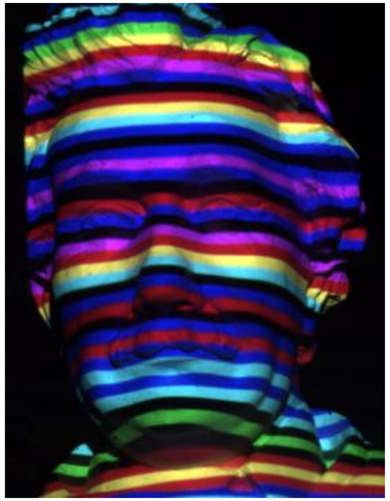


e.g. Eyetronics' ShapeCam

Multi-band triangulation

- Project multiple bands to speedup scanning
- But, which pixels belong to which band?
- Answer #2: Project color bands (or points)

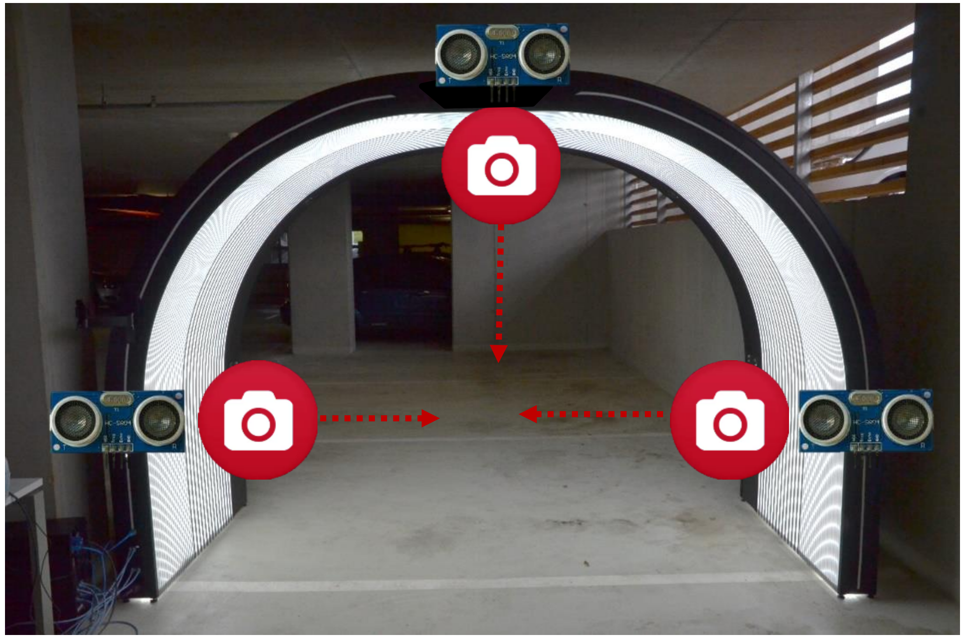


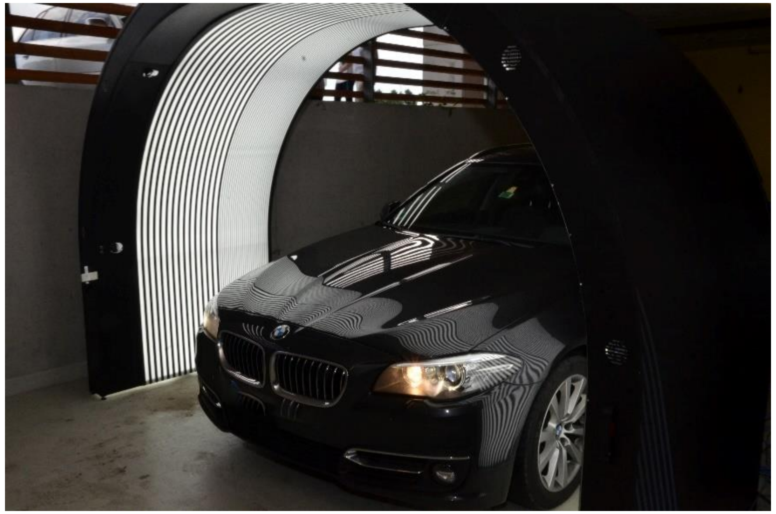


L. Zhang, B. Curless, and S. M. Seitz. Rapid Shape Acquisition Using Color Structured Light and Multi-pass Dynamic Programming. *3DPVT* 2002

Quality control

- Automatic car inspection
- Industrial project (2016)
- Designed sensory system and software

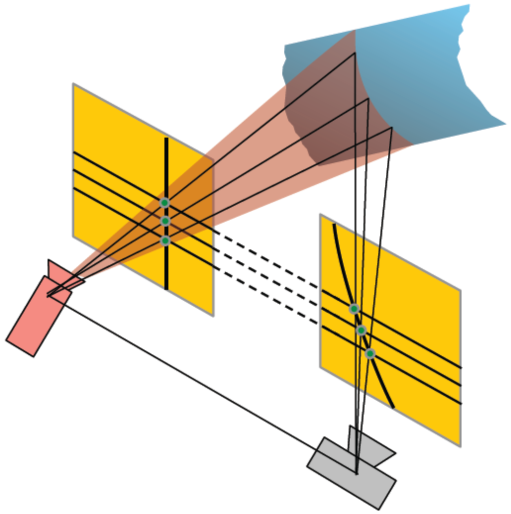


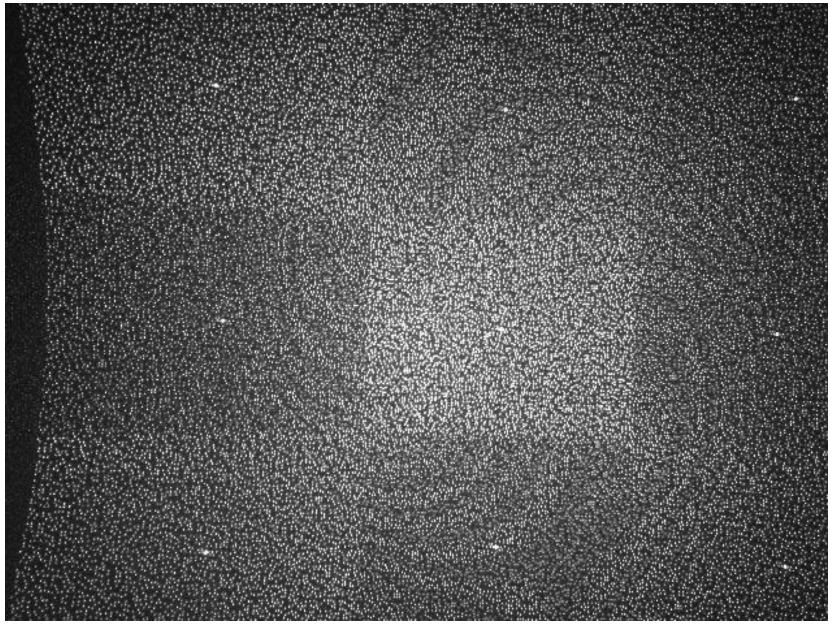


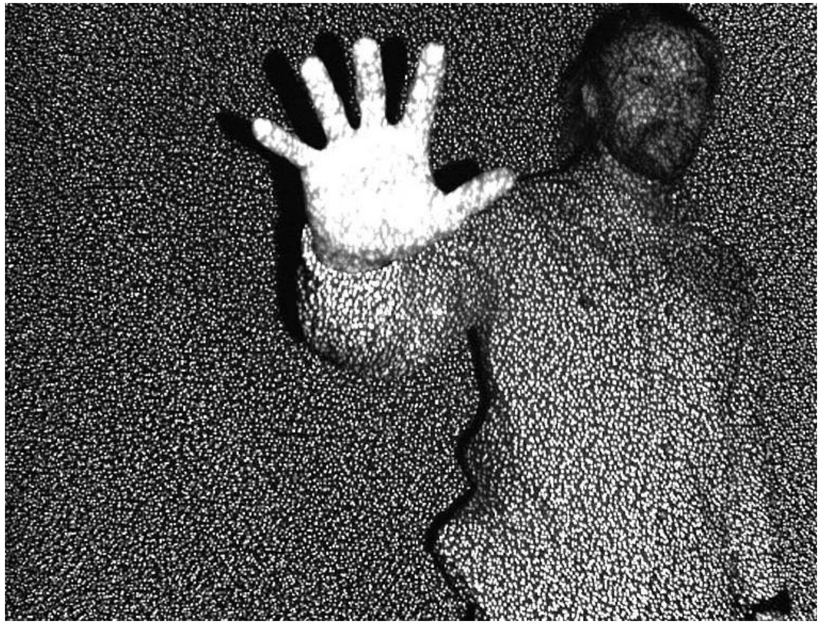
In gaming industry (2010)

[Accuracy and Resolution of Kinect Depth Data for Indoor Mapping Applications](#)
Kourosh Khoshelham and Sander Oude Elberink

- Project a point pattern for ultra fast triangulation!







In phones (2017)

- Project a point pattern for ultra fast triangulation!

https://www.youtube.com/watch?v=OvVKnC6gGtg

References

- David A. Forsyth, Jean Ponce, Computer Vision: A Modern Approach (2nd Edition),
  - Stereo: Chapter 7
  - Structure from motion: Section 8.1
- R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, 2nd Edition, Cambridge University Press, 2004
  - Camera model and calibration (Chapters 6 in 7)
  - Epipolar geometry (Chapter 9)
  - Calculating F (Chapters 11.1-11.6)
- Xiao, J. Multiview 3D Reconstruction for Dummies
- Trym Vegard Haavardsholm: Stereo processing, Lecture 6
- Patent Primesense (Kinect): http://patentscope.wipo.int/search/en/WO2007043036