Lecture 13
Stereo Reconstruction

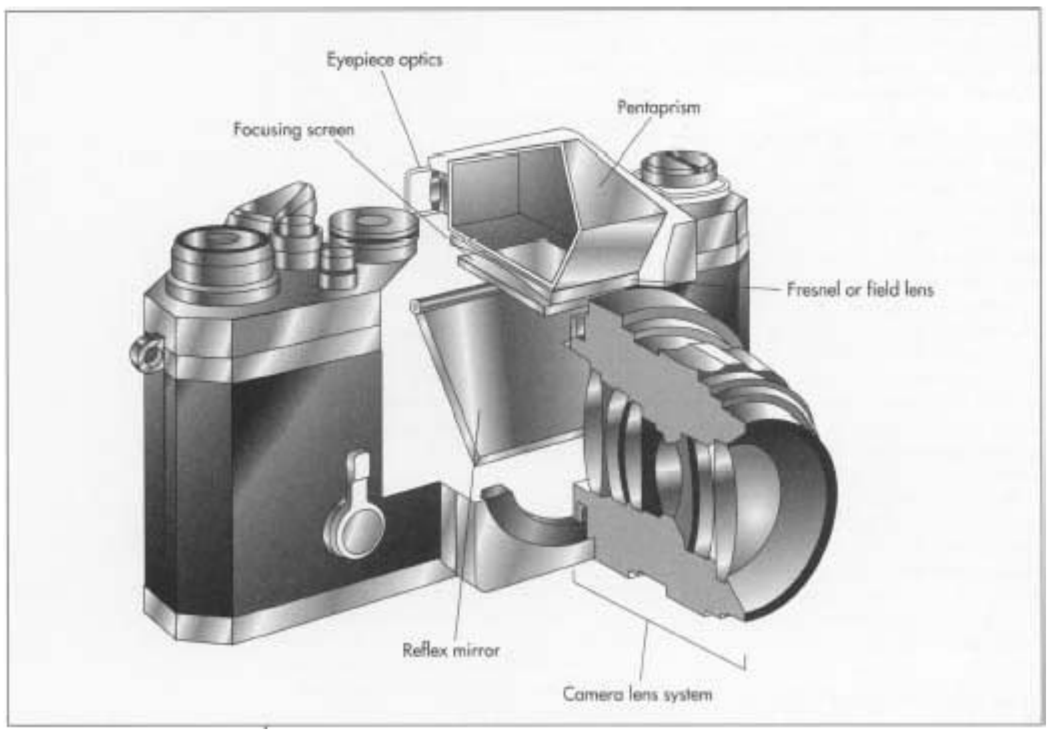
Slides from A. Zisserman & S. Lazebnik

Overview

- Single camera geometry
  - Recap of Homogenous coordinates
  - Perspective projection model
  - Camera calibration

- Stereo Reconstruction
  - Epipolar geometry
  - Stereo correspondence
  - Triangulation

Single camera geometry



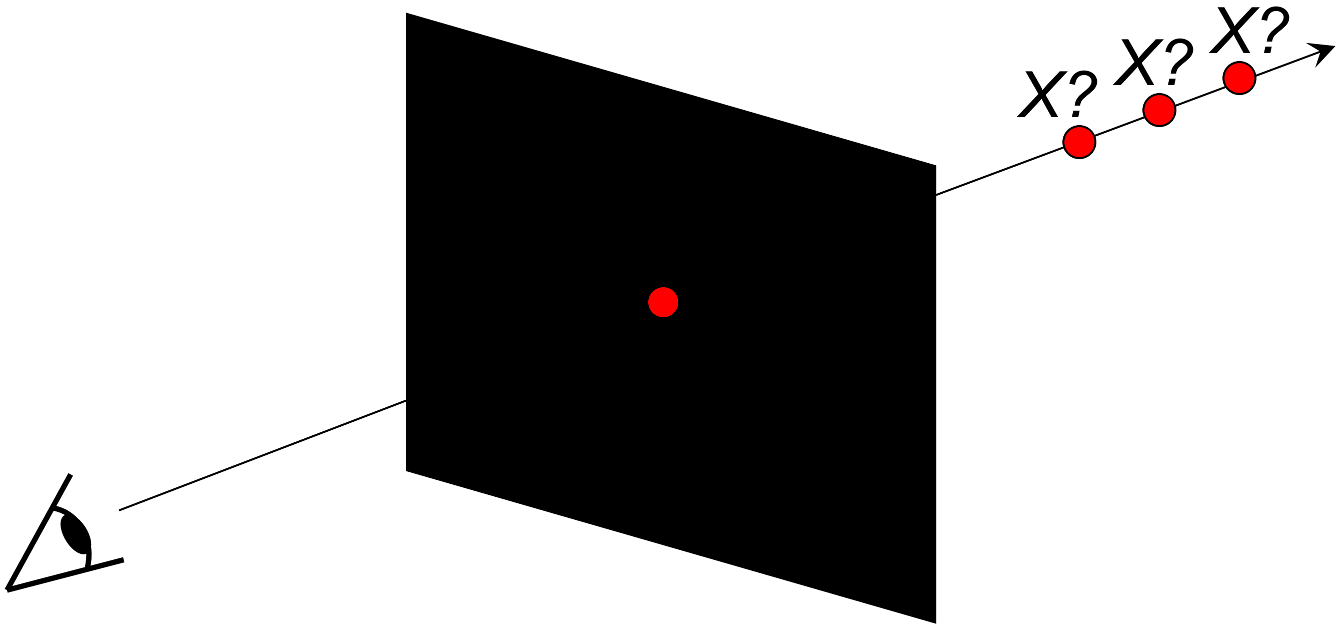
Projection


MAKE POVERTY HISTORY
JULIAN BEEVER
CoolOpticalIllusions.com

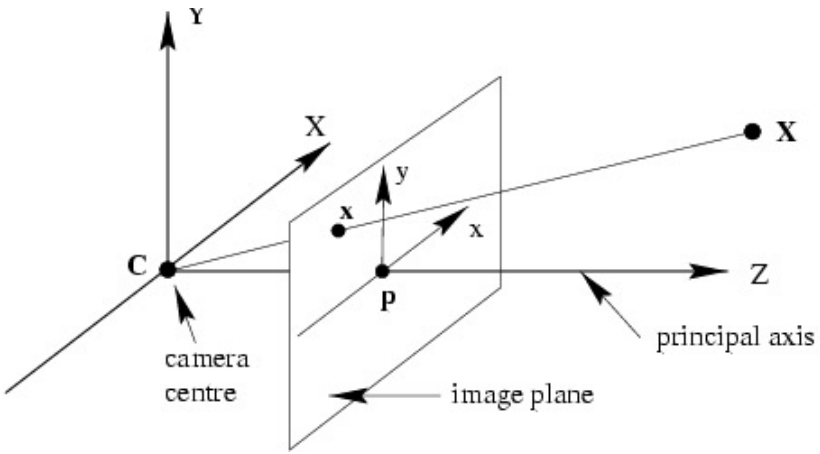
Projection

Projective Geometry

- Recovery of structure from one image is inherently ambiguous
- Today focus on geometry that maps world to camera image

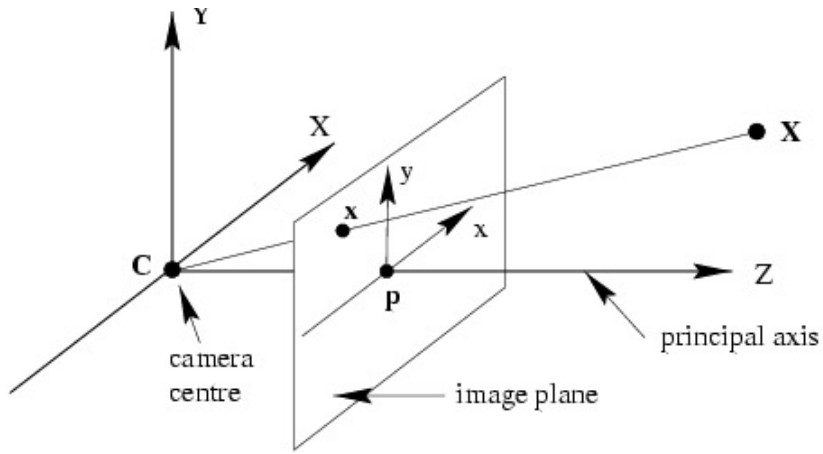


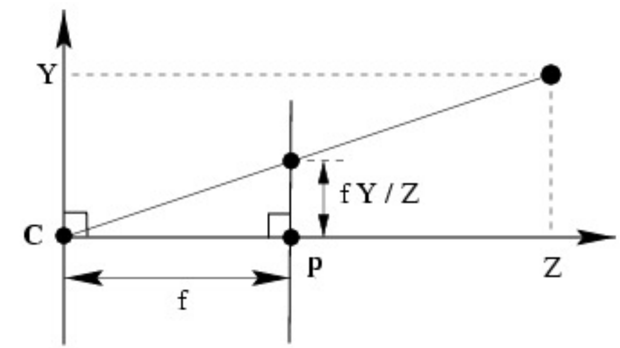
Recall: Pinhole camera model



- **Principal axis:** line from the camera center perpendicular to the image plane
- **Normalized (camera) coordinate system:** camera center is at the origin and the principal axis is the z-axis

Recall: Pinhole camera model





$$(X, Y, Z) \mapsto (f X / Z, f Y / Z)$$

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} f X \\ f Y \\ Z \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \qquad \mathbf{x} = \mathbf{P}\mathbf{X}$$

Recap: Homogeneous coordinates

- Is this a linear transformation? $(x, y, z) \rightarrow (f\frac{x}{z}, f\frac{y}{z})$
    - no—division by z is nonlinear

Trick: add one more coordinate:

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

homogeneous image coordinates

$$(x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

homogeneous scene coordinates

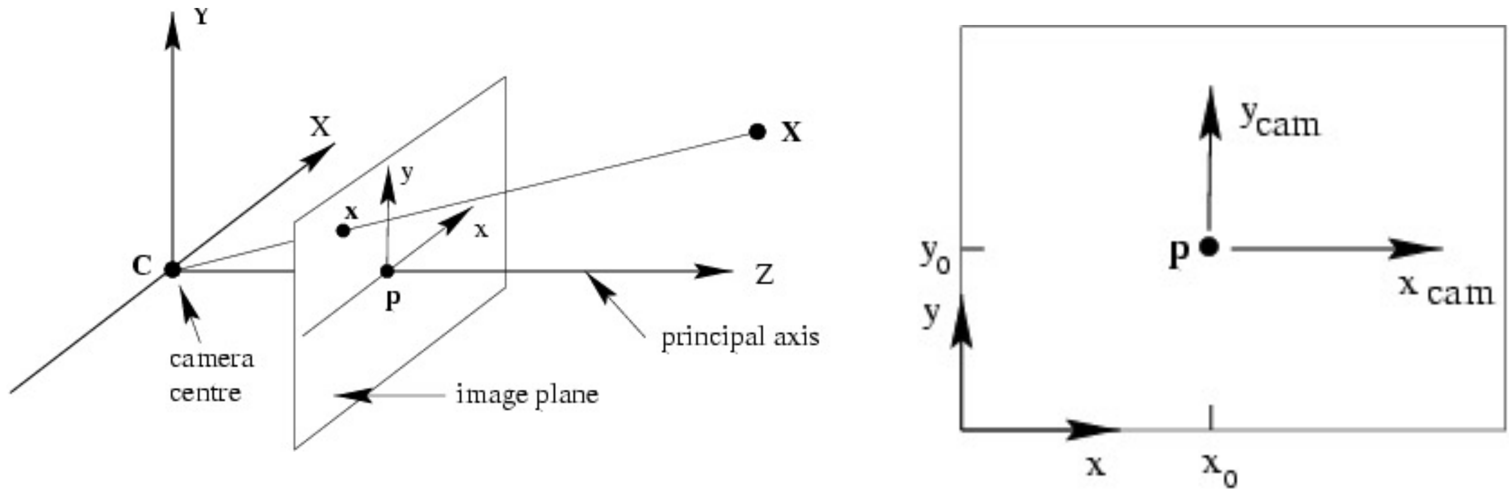
Converting *from* homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

$$\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Rightarrow (x/w, y/w, z/w)$$

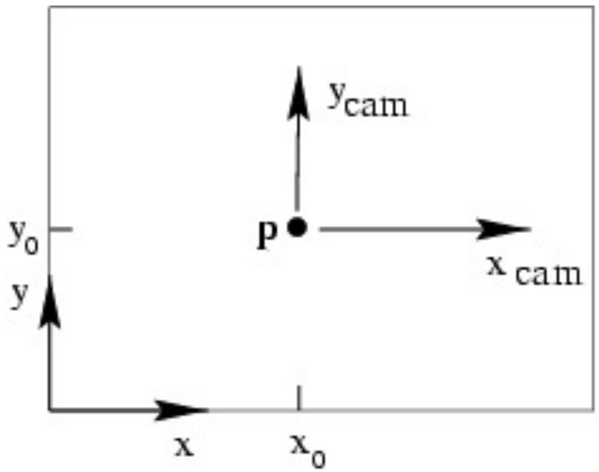
Slide by Steve Seitz

Principal point



- **Principal point (p):** point where principal axis intersects the image plane (origin of normalized coordinate system)
- Normalized coordinate system: origin is at the principal point
- Image coordinate system: origin is in the corner
- How to go from normalized coordinate system to image coordinate system?

Principal point offset

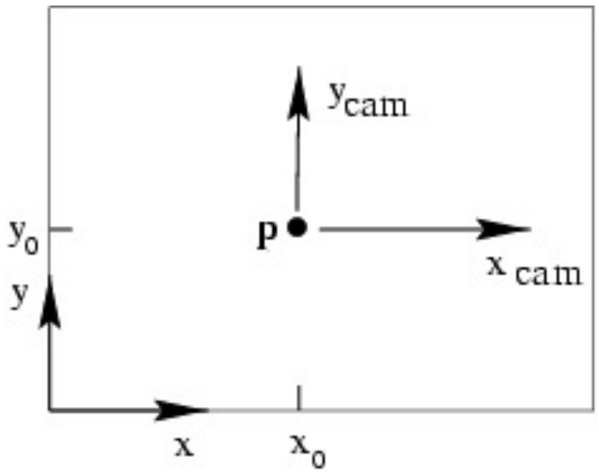


principal point: (p_x, p_y)

$$(X, Y, Z) \mapsto (f X / Z + p_x, f Y / Z + p_y)$$

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} f X + Z p_x \\ f Y + Z p_y \\ Z \end{pmatrix} = \begin{bmatrix} f & & p_x & 0 \\ & f & p_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Principal point offset

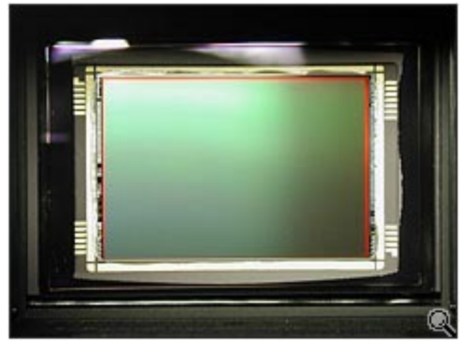


principal point: (p_x, p_y)

$$\begin{pmatrix} fX + Zp_x \\ fY + Zp_y \\ Z \end{pmatrix} = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

$$K = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}$$ calibration matrix $$\mathrm{P} = \mathrm{K}[\mathrm{I} \mid 0]$$

Pixel coordinates

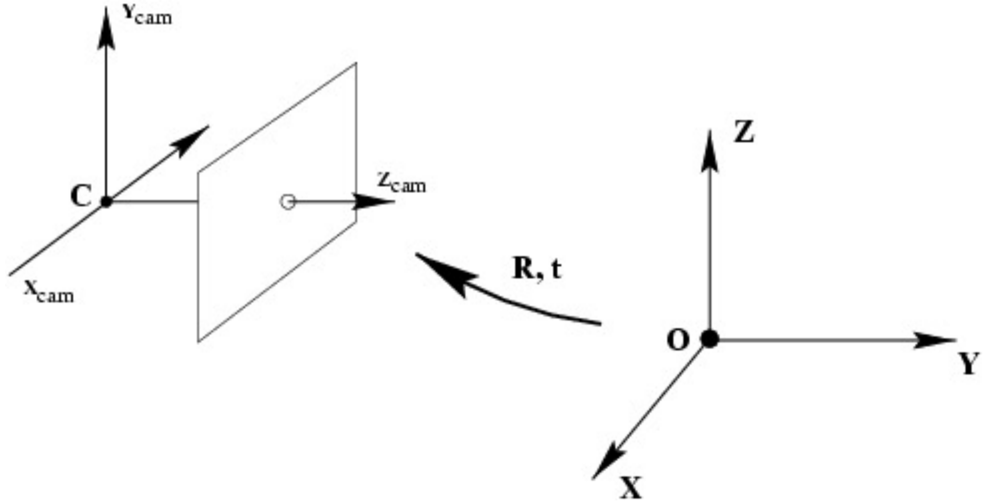


Pixel size: $\dfrac{1}{m_x} \times \dfrac{1}{m_y}$

- m_x pixels per meter in horizontal direction,
  m_y pixels per meter in vertical direction

$$K = \underbrace{\begin{bmatrix} m_x & & \\ & m_y & \\ & & 1 \end{bmatrix}}_{\text{pixels/m}} \underbrace{\begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}}_{\text{m}} = \underbrace{\begin{bmatrix} \alpha_x & & \beta_x \\ & \alpha_y & \beta_y \\ & & 1 \end{bmatrix}}_{\text{pixels}}$$

Camera rotation and translation



- In general, the camera coordinate frame will be related to the world coordinate frame by a rotation and a translation

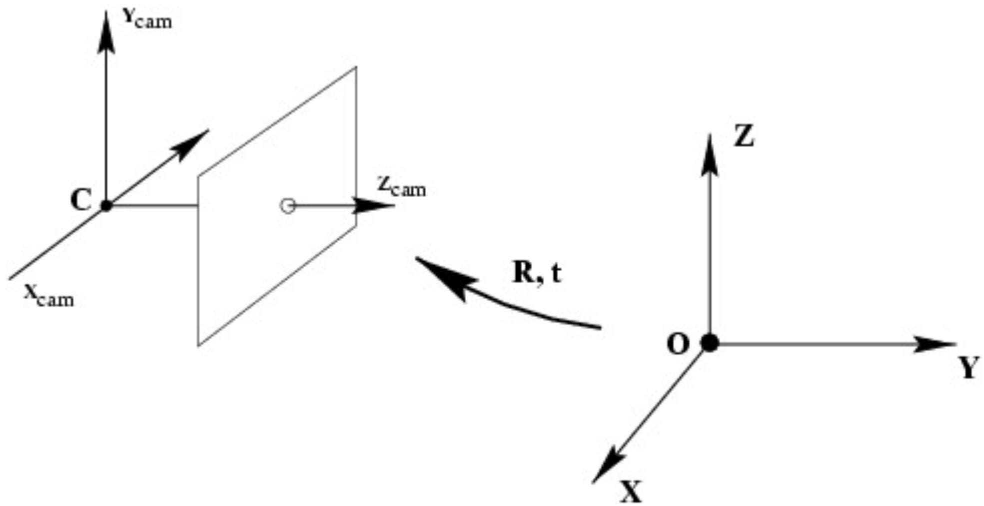
$$\tilde{X}_{cam} = R\left(\tilde{X} - \tilde{C}\right)$$

coords. of point in camera frame

coords. of a point in world frame (nonhomogeneous)

coords. of camera center in world frame

Camera rotation and translation



In non-homogeneous coordinates:

$$\tilde{X}_{cam} = R\left(\tilde{X} - \tilde{C}\right)$$

$$X_{cam} = \begin{bmatrix} R & -R\tilde{C} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} \tilde{X} \\ 1 \end{pmatrix} = \begin{bmatrix} R & -R\tilde{C} \\ 0 & 1 \end{bmatrix} X$$

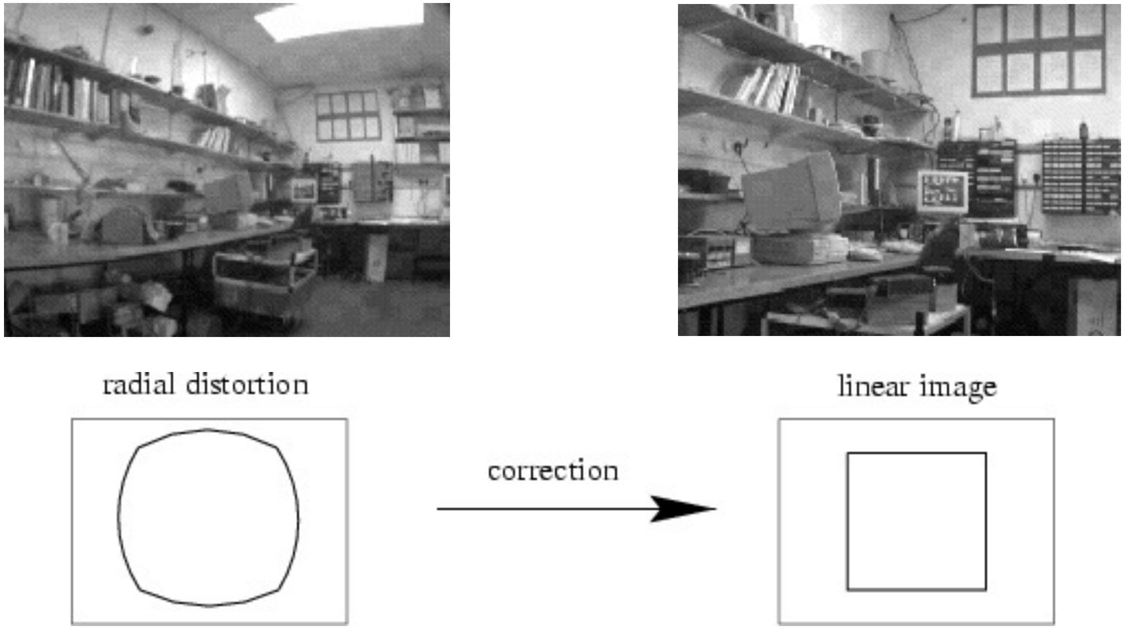
$$x = K[I \mid 0]X_{cam} = K\left[R \mid -R\tilde{C}\right]X \qquad P = K[R \mid t], \qquad t = -R\tilde{C}$$

Note: C is the null space of the camera projection matrix (PC=0)

Camera parameters

- Intrinsic parameters
  - Principal point coordinates
  - Focal length
  - Pixel magnification factors
  - *Skew (non-rectangular pixels)*
  - *Radial distortion*

$$K = \begin{bmatrix} m_x & & \\ & m_y & \\ & & 1 \end{bmatrix} \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} = \begin{bmatrix} \alpha_x & & \beta_x \\ & \alpha_y & \beta_y \\ & & 1 \end{bmatrix}$$

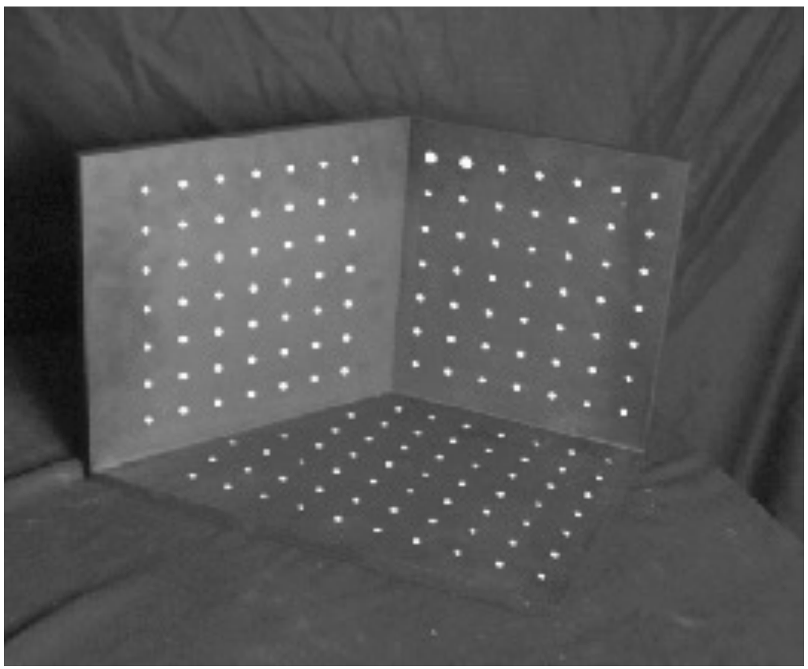


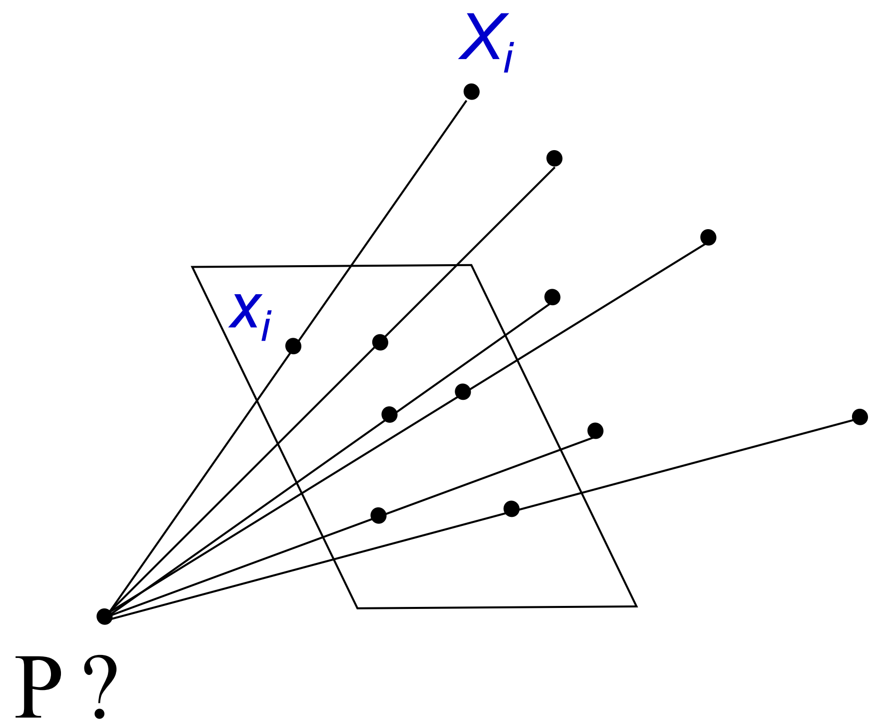
Camera parameters

- Intrinsic parameters
  - Principal point coordinates
  - Focal length
  - Pixel magnification factors
  - *Skew (non-rectangular pixels)*
  - *Radial distortion*
- Extrinsic parameters
  - Rotation and translation relative to world coordinate system

Camera calibration

- Given n points with known 3D coordinates X_i and known image projections x_i, estimate the camera parameters





Camera calibration

$$\lambda \mathrm{x}_i = \mathrm{P}\mathrm{X}_i \qquad \mathrm{x}_i \times \mathrm{P}\mathrm{X}_i = 0 \qquad \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \times \begin{bmatrix} \mathrm{P}_1^T \mathrm{X}_i \\ \mathrm{P}_2^T \mathrm{X}_i \\ \mathrm{P}_3^T \mathrm{X}_i \end{bmatrix} = 0$$

$$\begin{bmatrix} 0 & -\mathrm{X}_i^T & y_i \mathrm{X}_i^T \\ \mathrm{X}_i^T & 0 & -x_i \mathrm{X}_i^T \\ -y_i \mathrm{X}_i^T & x_i \mathrm{X}_i^T & 0 \end{bmatrix} \begin{pmatrix} \mathrm{P}_1 \\ \mathrm{P}_2 \\ \mathrm{P}_3 \end{pmatrix} = 0$$

Two linearly independent equations

Camera calibration

$$\begin{bmatrix} 0^T & X_1^T & -y_1 X_1^T \\ X_1^T & 0^T & -x_1 X_1^T \\ \cdots & \cdots & \cdots \\ 0^T & X_n^T & -y_n X_n^T \\ X_n^T & 0^T & -x_n X_n^T \end{bmatrix} \begin{pmatrix} P_1 \\ P_2 \\ P_3 \end{pmatrix} = 0 \qquad \mathrm{Ap} = 0$$

- P has 11 degrees of freedom (12 parameters, but scale is arbitrary)
- One 2D/3D correspondence gives us two linearly independent equations
- Homogeneous least squares
- 6 correspondences needed for a minimal solution

Camera calibration

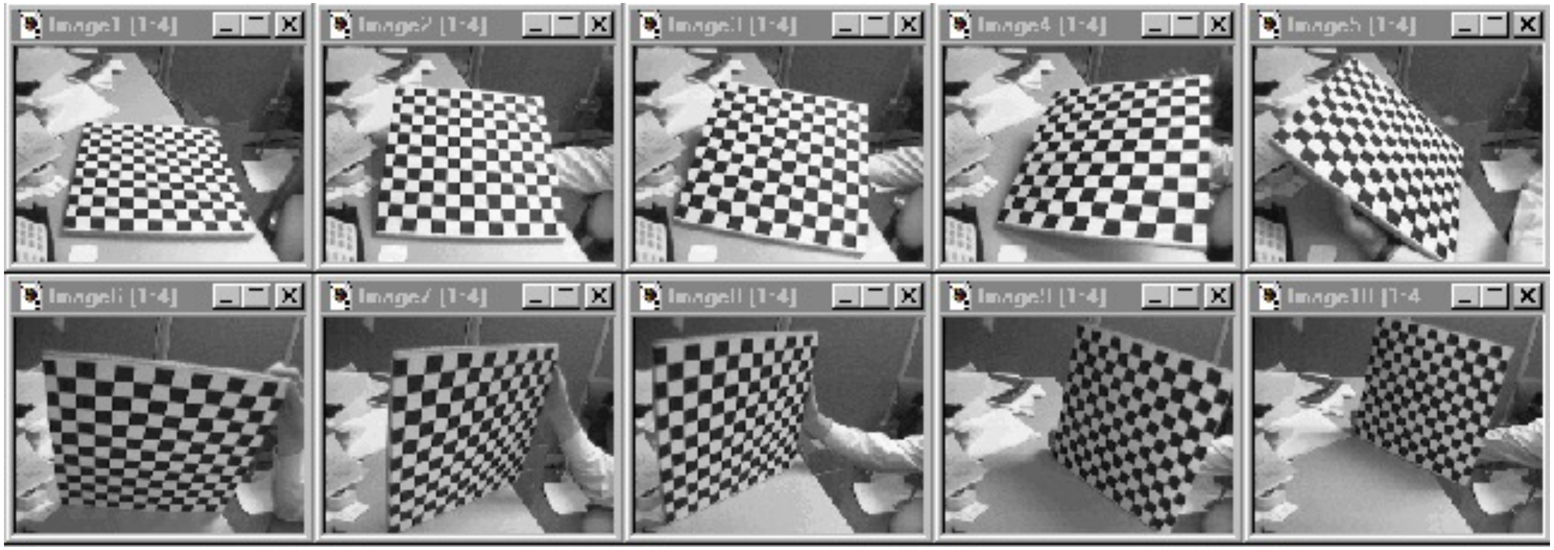
$$\begin{bmatrix} 0^T & X_1^T & -y_1 X_1^T \\ X_1^T & 0^T & -x_1 X_1^T \\ \cdots & \cdots & \cdots \\ 0^T & X_n^T & -y_n X_n^T \\ X_n^T & 0^T & -x_n X_n^T \end{bmatrix} \begin{pmatrix} P_1 \\ P_2 \\ P_3 \end{pmatrix} = 0 \qquad \mathrm{Ap} = 0$$

- Note: for coplanar points that satisfy $\Pi^T X=0$, we will get degenerate solutions $(\Pi,0,0)$, $(0,\Pi,0)$, or $(0,0,\Pi)$

Camera calibration

- Once we've recovered the numerical form of the camera matrix, we still have to figure out the intrinsic and extrinsic parameters
- This is a matrix decomposition problem, not an estimation problem (see F&P sec. 3.2, 3.3)

Alternative: multi-plane calibration



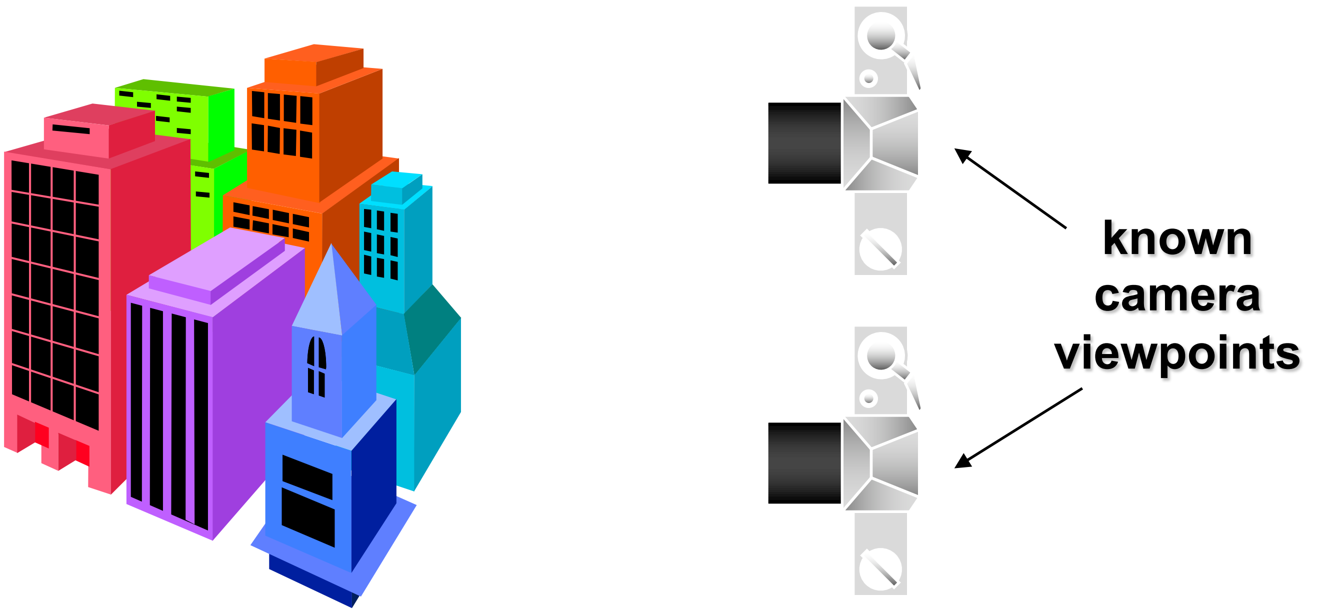
Images courtesy Jean-Yves Bouguet, Intel Corp.

Advantage

- Only requires a plane
- Don’t have to know positions/orientations
- Good code available online!
  - Intel’s OpenCV library: http://www.intel.com/research/mrl/research/opencv/
  - Matlab version by Jean-Yves Bouget: http://www.vision.caltech.edu/bouguetj/calib_doc/index.html
  - Zhengyou Zhang’s web site: http://research.microsoft.com/~zhang/Calib/

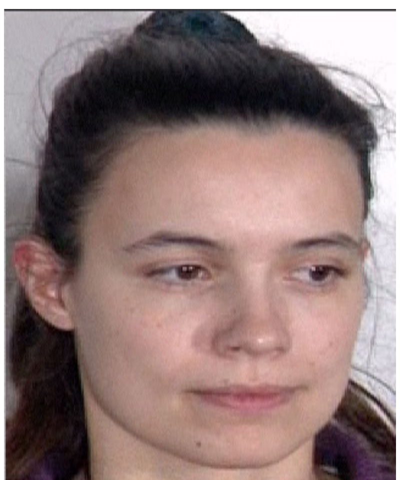
Stereo Reconstruction

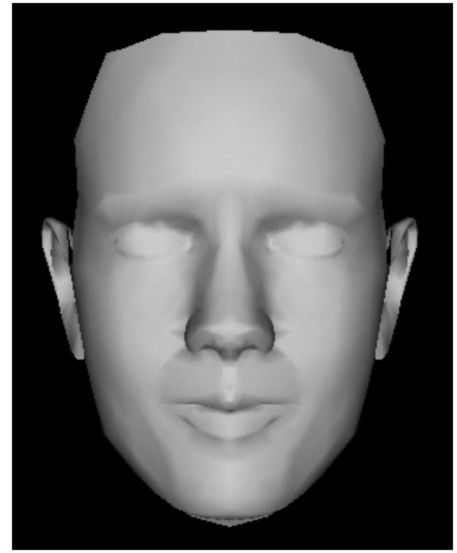
Shape (3D) from two (or more) images



Example

images





shape

surface reflectance

Scenarios

The two images can arise from

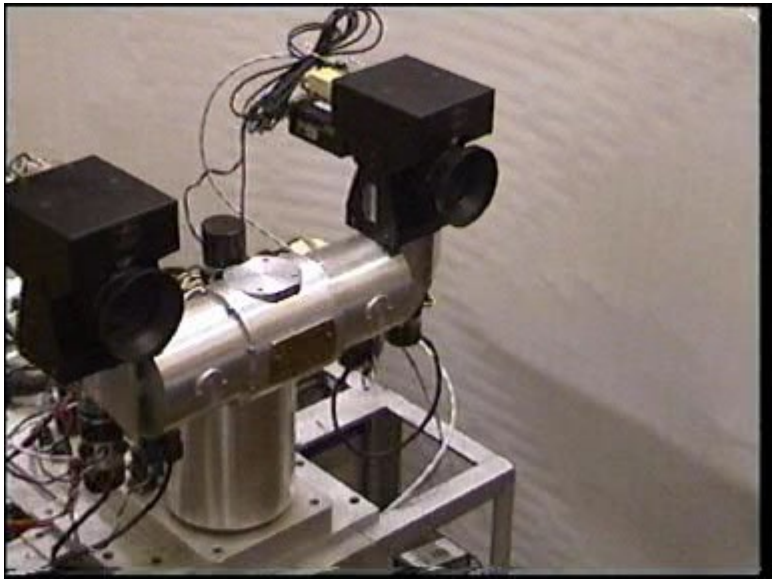
- A stereo rig consisting of two cameras
  - the two images are acquired simultaneously

or

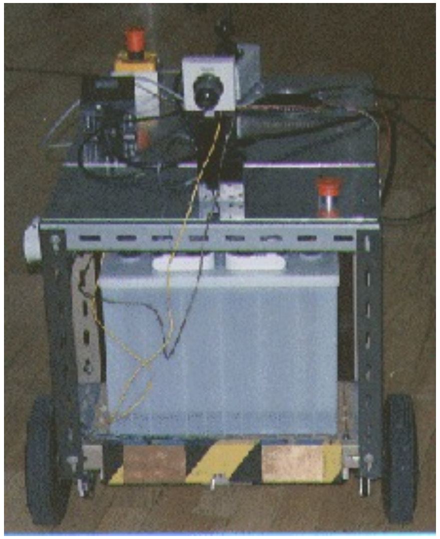
- A single moving camera (static scene)
  - the two images are acquired sequentially

The two scenarios are geometrically equivalent

Stereo head



Camera on a mobile vehicle



The objective

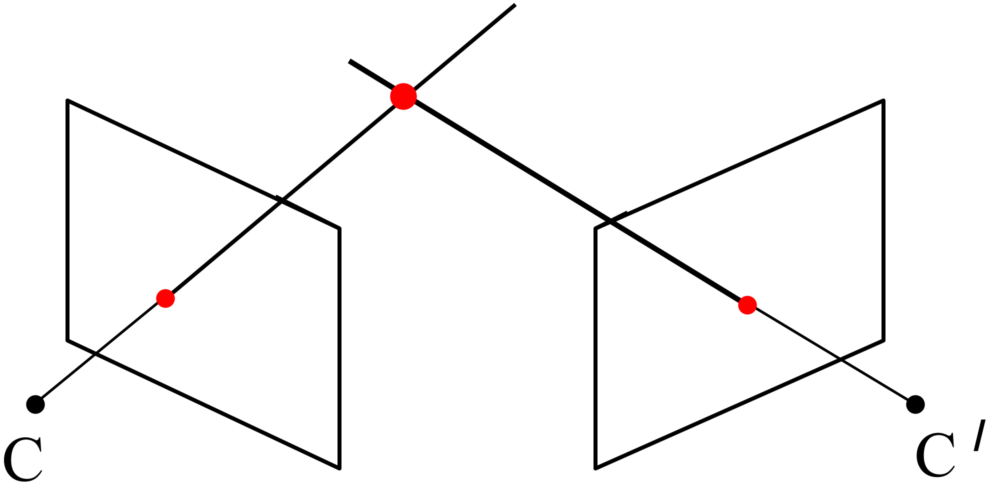
Given two images of a scene acquired by known cameras compute the 3D position of the scene (structure recovery)

Basic principle: triangulate from corresponding image points

- Determine 3D point at intersection of two back-projected rays

Corresponding points are images of the same scene point

Triangulation



The back-projected points generate rays which intersect at the 3D scene point

An algorithm for stereo reconstruction

1. For each point in the first image determine the corresponding point in the second image

   (this is a search problem)

2. For each pair of matched points determine the 3D point by triangulation

   (this is an estimation problem)

The correspondence problem

Given a point x in one image find the corresponding point in the other image

This appears to be a 2D search problem, but it is reduced to a 1D search by the epipolar constraint

Outline

1. Epipolar geometry

   - the geometry of two cameras
   - reduces the correspondence problem to a line search

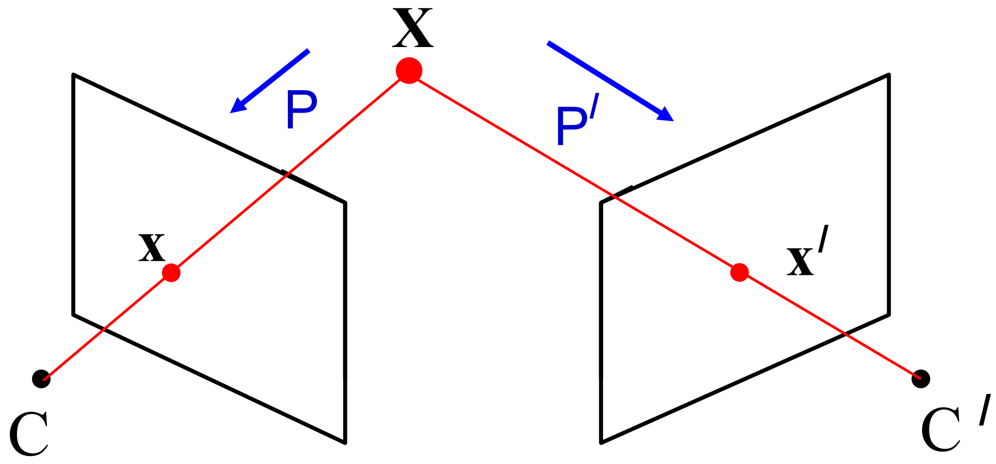
2. Stereo correspondence algorithms

3. Triangulation

Notation

The two cameras are P and P', and a 3D point $\mathbf{X}$ is imaged as

$$\mathbf{x} = \mathrm{P}\mathbf{X} \qquad \mathbf{x}' = \mathrm{P}'\mathbf{X}$$



P : 3×4 matrix

$\mathbf{X}$: 4-vector

$\mathbf{x}$: 3-vector

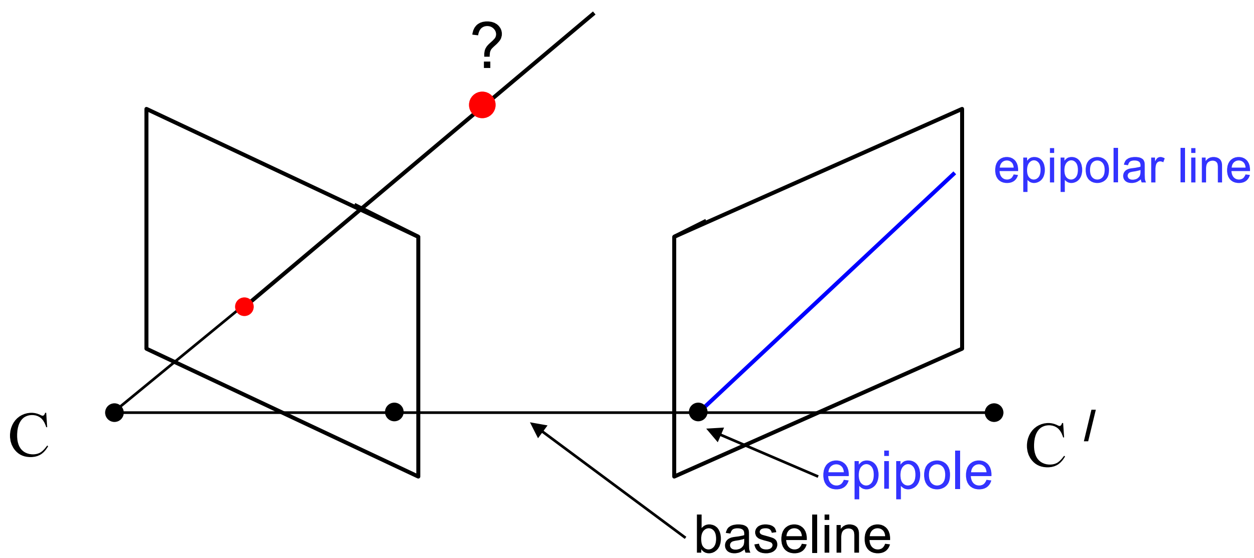
Warning

for equations involving homogeneous quantities '=' means 'equal up to scale'

Epipolar geometry

Epipolar geometry

Given an image point in one view, where is the corresponding point in the other view?



- A point in one view "generates" an epipolar line in the other view
- The corresponding point lies on this line

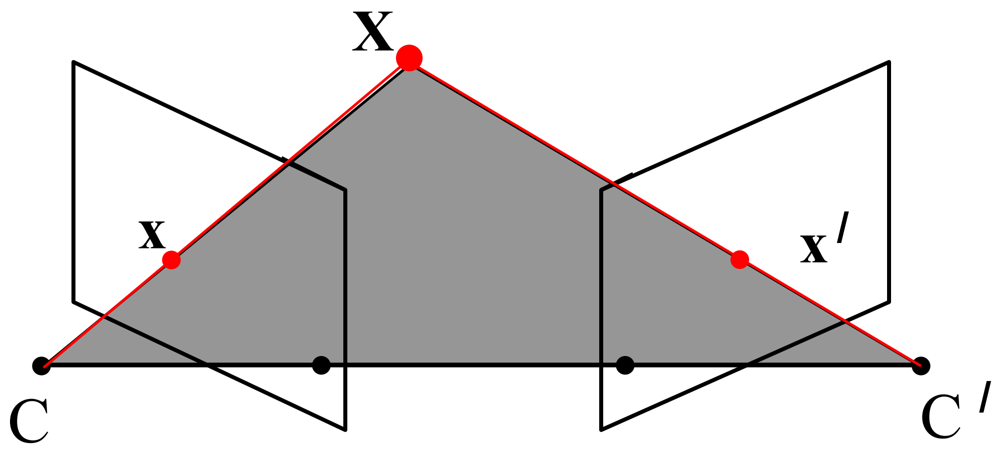
Epipolar line

Epipolar constraint

- Reduces correspondence problem to 1D search along an epipolar line

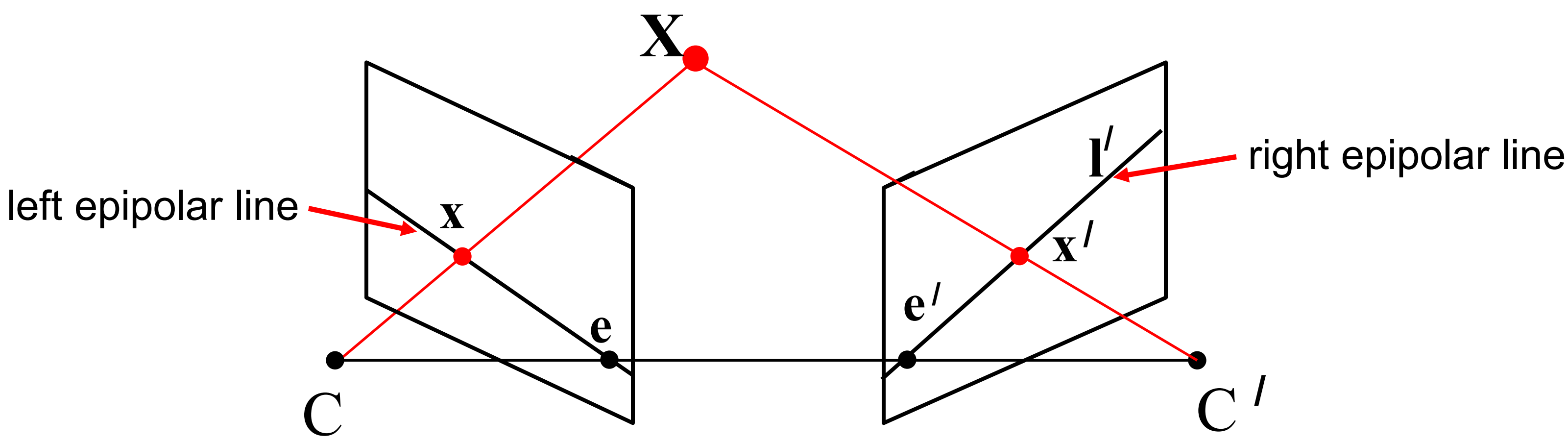
Epipolar geometry continued

Epipolar geometry is a consequence of the coplanarity of the camera centres and scene point



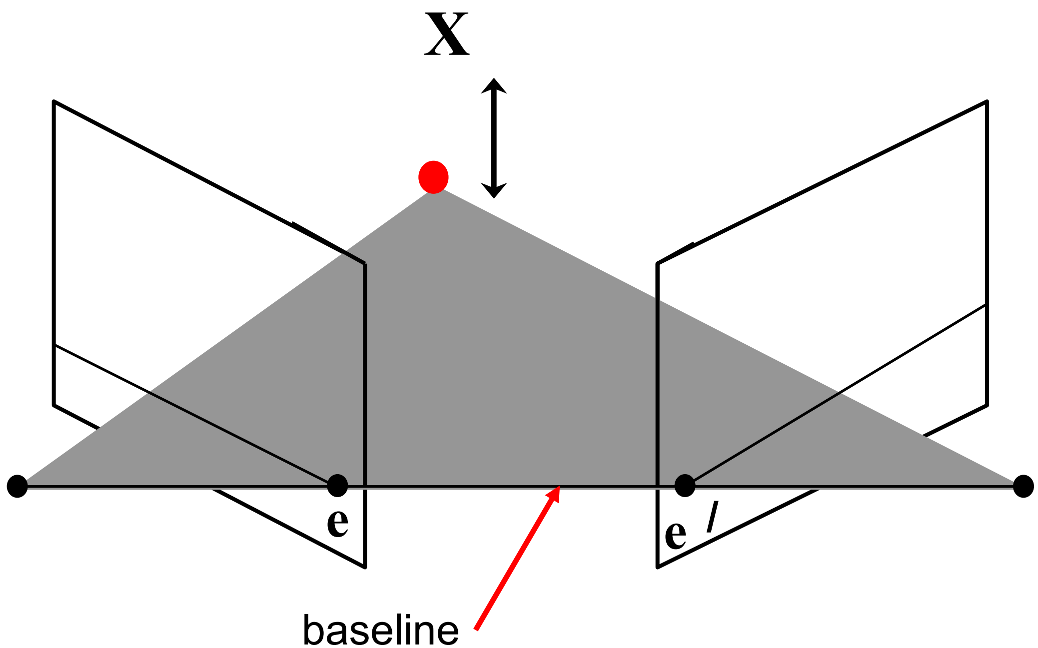
The camera centres, corresponding points and scene point lie in a single plane, known as the epipolar plane

Nomenclature



- The epipolar line $\mathbf{l}'$ is the image of the ray through $\mathbf{x}$
- The epipole $\mathbf{e}$ is the point of intersection of the line joining the camera centres with the image plane
  - this line is the baseline for a stereo rig, and
  - the translation vector for a moving camera
- The epipole is the image of the centre of the other camera: $\mathbf{e} = \mathrm{P}\mathbf{C}'$, $\mathbf{e}' = \mathrm{P}'\mathbf{C}$

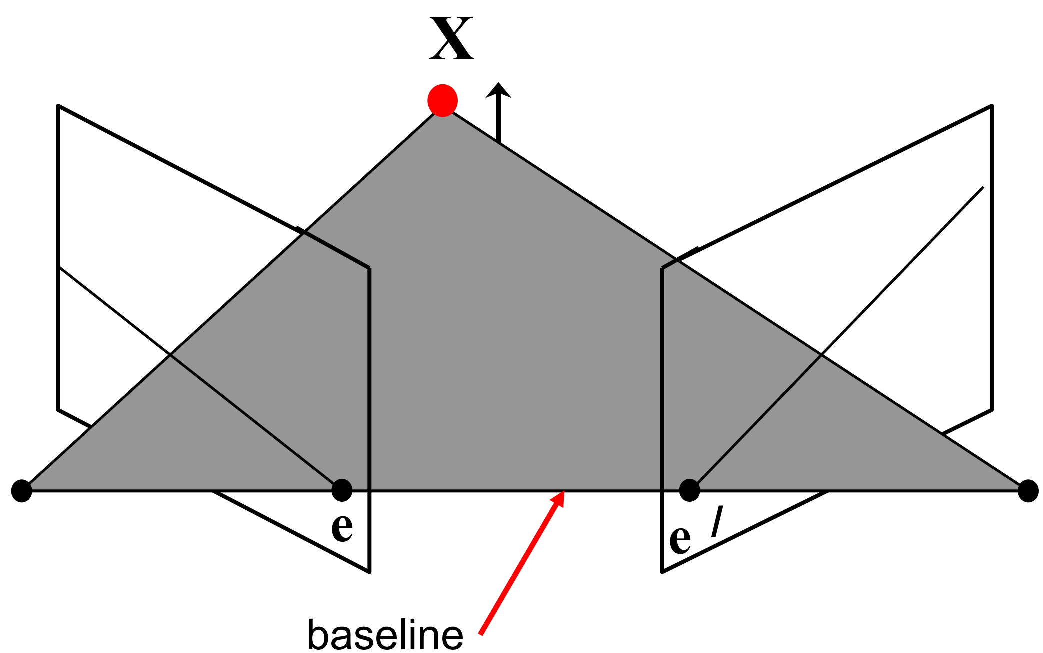
The epipolar pencil



As the position of the 3D point $\mathbf{X}$ varies, the epipolar planes “rotate” about the baseline. This family of planes is known as an epipolar pencil. All epipolar lines intersect at the epipole.

(a pencil is a one parameter family)

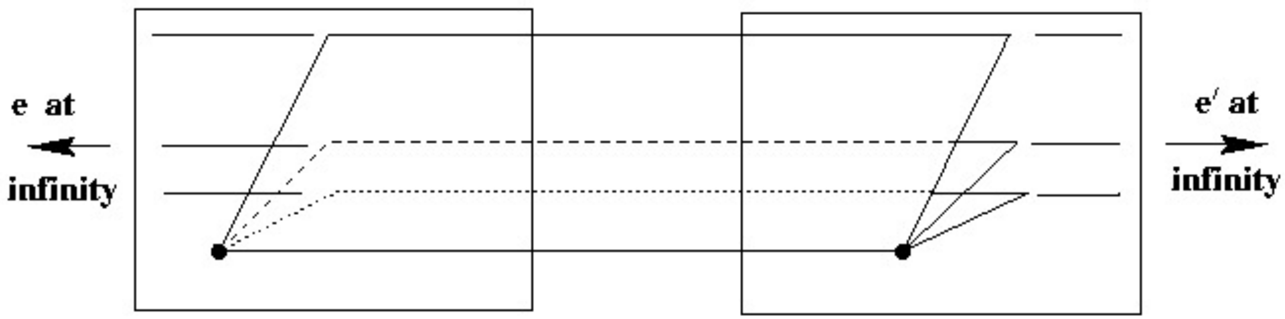
The epipolar pencil

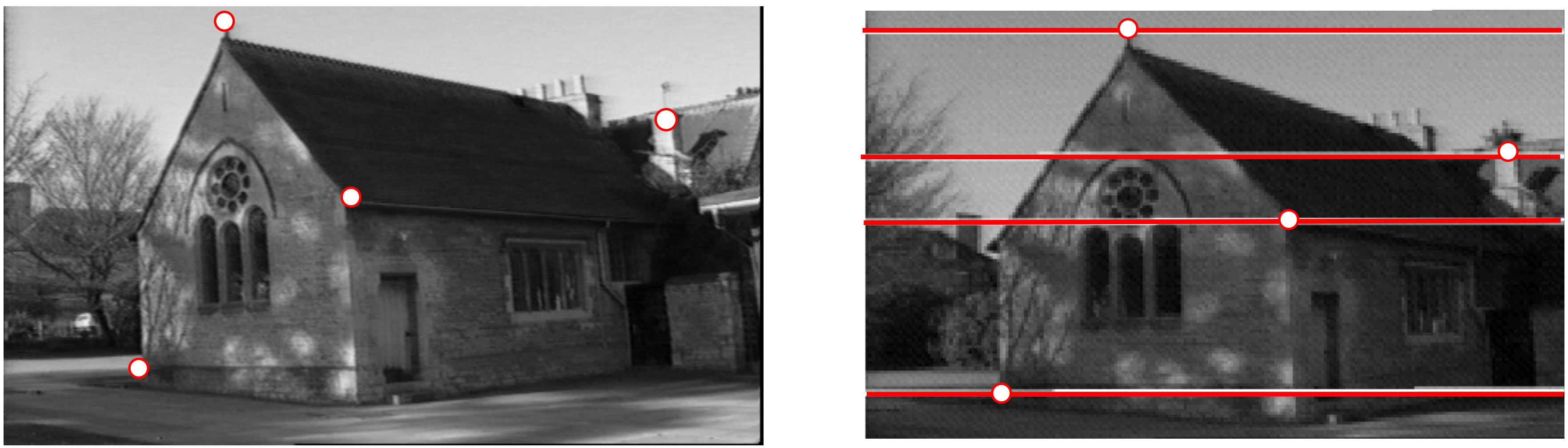


As the position of the 3D point $\mathbf{X}$ varies, the epipolar planes "rotate" about the baseline. This family of planes is known as an epipolar pencil. All epipolar lines intersect at the epipole.

(a pencil is a one parameter family)

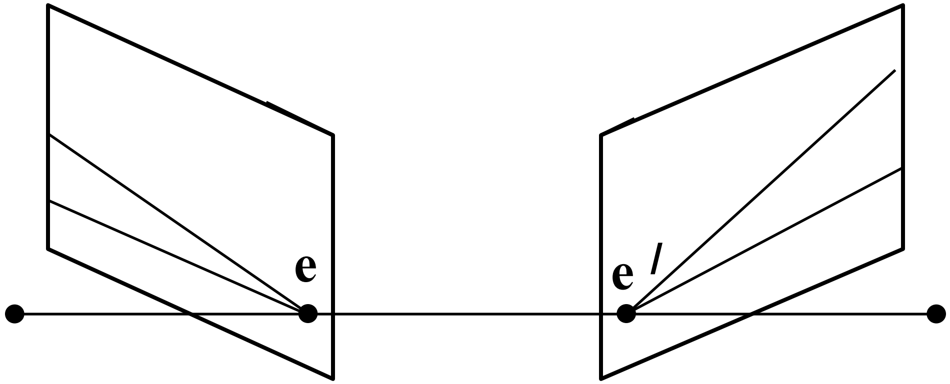
Epipolar geometry example I: parallel cameras

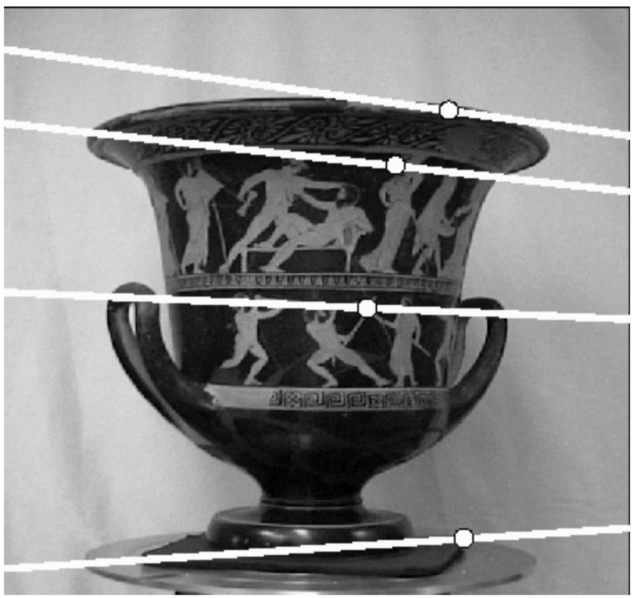


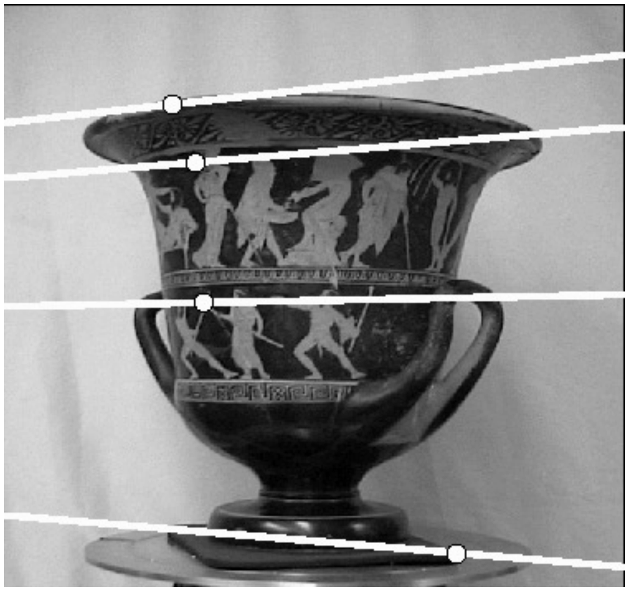


Epipolar geometry depends only on the relative pose (position and orientation) and internal parameters of the two cameras, i.e. the position of the camera centres and image planes. It does not depend on the scene structure (3D points external to the camera).

Epipolar geometry example II: converging cameras







Note, epipolar lines are in general not parallel

Homogeneous notation for lines

Recall that a point (x, y) in 2D is represented by the homogeneous 3-vector $\mathbf{x} = (x_1, x_2, x_3)^\top$, where $x = x_1/x_3, y = x_2/x_3$

A line in 2D is represented by the homogeneous 3-vector

$$\mathbf{l} = \begin{pmatrix} l_1 \\ l_2 \\ l_3 \end{pmatrix}$$

which is the line $l_1 x + l_2 y + l_3 = 0$.

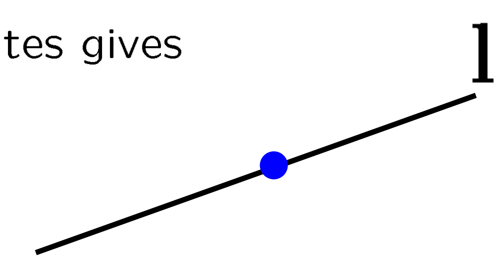
Example represent the line $y = 1$ as a homogeneous vector.

Write the line as $-y + 1 = 0$ then $l_1 = 0, l_2 = -1, l_3 = 1$, and $\mathbf{l} = (0, -1, 1)^\top$.

Note that $\mu(l_1 x + l_2 y + l_3) = 0$ represents the same line (only the ratio of the homogeneous line coordinates is significant).

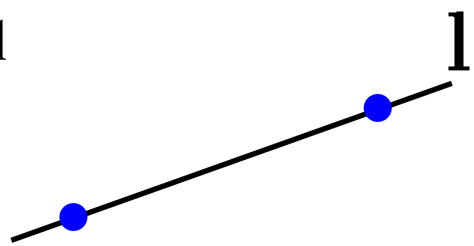
Writing both the point and line in homogeneous coordinates gives

$$l_1 x_1 + l_2 x_2 + l_3 x_3 = 0$$



- point on line $\mathbf{l}.\mathbf{x} = 0$ or $\mathbf{l}^\top \mathbf{x} = 0$ or $\mathbf{x}^\top \mathbf{l} = 0$

- The line $\mathbf{l}$ through the two points $\mathbf{p}$ and $\mathbf{q}$ is $\mathbf{l} = \mathbf{p} \times \mathbf{q}$



Proof

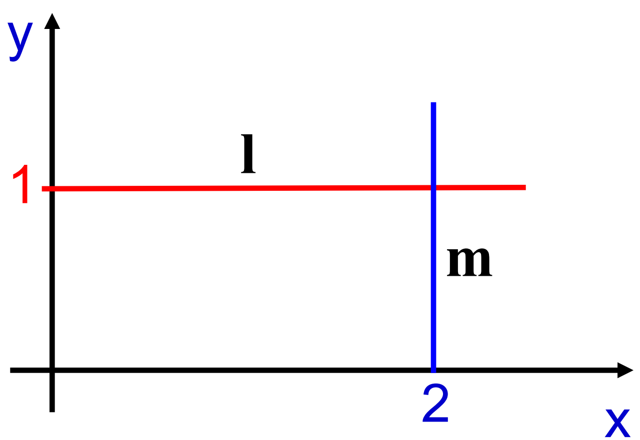
$$\mathbf{l}.\mathbf{p} = (\mathbf{p} \times \mathbf{q}).\mathbf{p} = 0 \qquad \mathbf{l}.\mathbf{q} = (\mathbf{p} \times \mathbf{q}).\mathbf{q} = 0$$

- The intersection of two lines $\mathbf{l}$ and $\mathbf{m}$ is the point $\mathbf{x} = \mathbf{l} \times \mathbf{m}$

Example: compute the point of intersection of the two lines $\mathbf{l}$ and $\mathbf{m}$ in the figure below

$$\mathbf{l} = \begin{pmatrix} 0 \\ -1 \\ 1 \end{pmatrix} \qquad \mathbf{m} = \begin{pmatrix} -1 \\ 0 \\ 2 \end{pmatrix}$$

$$\mathbf{x} = \mathbf{l} \times \mathbf{m} = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ 0 & -1 & 1 \\ -1 & 0 & 2 \end{vmatrix} = \begin{pmatrix} -2 \\ -1 \\ -1 \end{pmatrix}$$



which is the point (2,1)

Matrix representation of the vector cross product

The vector product $\mathbf{v} \times \mathbf{x}$ can be represented as a matrix multiplication

$$\mathbf{v} \times \mathbf{x} = \begin{pmatrix} v_2 x_3 - v_3 x_2 \\ v_3 x_1 - v_1 x_3 \\ v_1 x_2 - v_2 x_1 \end{pmatrix} = [\mathbf{v}]_\times \mathbf{x}$$

where

$$[\mathbf{v}]_\times = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}$$

- $[\mathbf{v}]_\times$ is a 3×3 skew-symmetric matrix of rank 2.
- $\mathbf{v}$ is the null-vector of $[\mathbf{v}]_\times$, since $\mathbf{v} \times \mathbf{v} = [\mathbf{v}]_\times \mathbf{v} = \mathbf{0}$.

Example: compute the cross product of **l** and **m**

$$\mathbf{l} = \begin{pmatrix} 0 \\ -1 \\ 1 \end{pmatrix} \qquad \mathbf{m} = \begin{pmatrix} -1 \\ 0 \\ 2 \end{pmatrix} \qquad [\mathbf{v}]_\times = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}$$

$$\mathbf{x} = \mathbf{l} \times \mathbf{m} = [\mathbf{l}]_\times \mathbf{m} = \begin{bmatrix} 0 & -1 & -1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{pmatrix} -1 \\ 0 \\ 2 \end{pmatrix} = \begin{pmatrix} -2 \\ -1 \\ -1 \end{pmatrix}$$

Note

$$\begin{bmatrix} 0 & -1 & -1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{pmatrix} 0 \\ -1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Algebraic representation of epipolar geometry

We know that the epipolar geometry defines a mapping

$$\mathbf{x} \quad \rightarrow \quad \mathbf{l}'$$

point in first image

epipolar line in second image

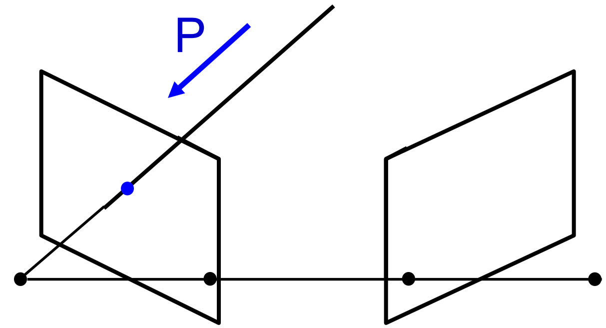
- the map ony depends on the cameras $\mathtt{P}, \mathtt{P}'$ (not on structure)

- it will be shown that the map is linear and can be written as $\mathbf{l}' = \mathtt{F}\mathbf{x}$, where $\mathtt{F}$ is a 3×3 matrix called the fundamental matrix

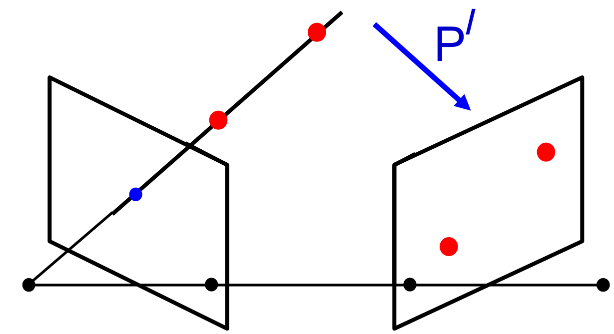
Derivation of the algebraic expression $\mathbf{l}' = \mathtt{F}\mathbf{x}$

Outline

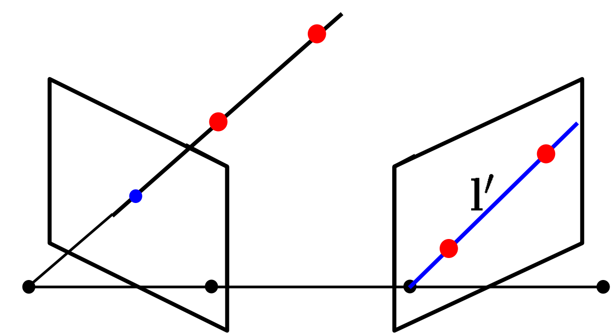
Step 1: for a point x in the first image back project a ray with camera P



Step 2: choose two points on the ray and project into the second image with camera P'



Step 3: compute the line through the two image points using the relation $\mathbf{l}' = \mathbf{p} \times \mathbf{q}$



- choose camera matrices

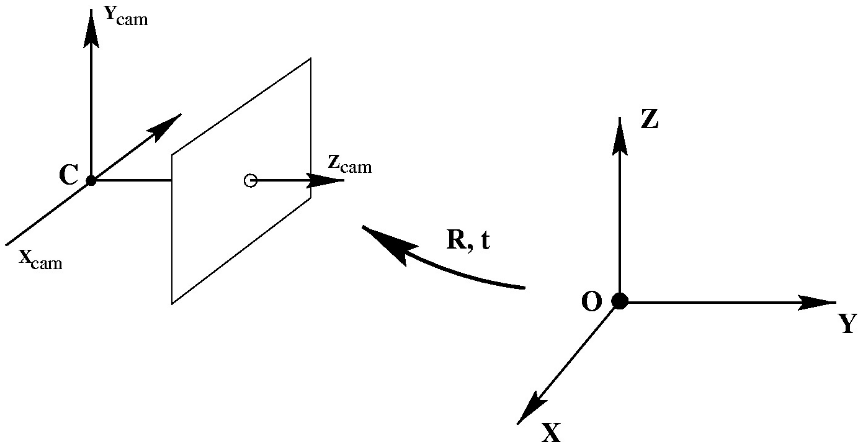
$$P = K[R \mid t]$$

internal calibration

rotation

translation

from world to camera coordinate frame

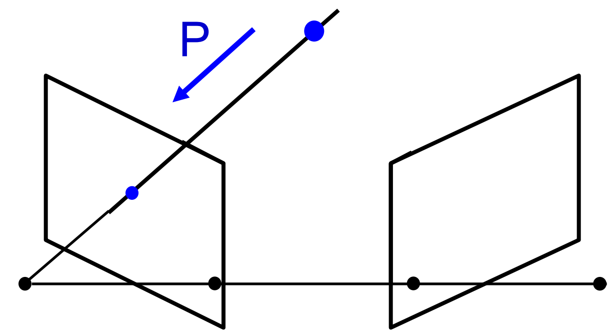


- first camera $\quad P = K[I \mid 0]$

world coordinate frame aligned with first camera

- second camera $\quad P' = K'[R \mid t]$

Step 1: for a point x in the first image
back project a ray with camera $\mathtt{P} = \mathtt{K}\,[\,\mathtt{I} \mid \mathbf{0}\,]$



A point $\mathbf{x}$ back projects to a ray

$$\begin{pmatrix} \mathrm{X} \\ \mathrm{Y} \\ \mathrm{Z} \end{pmatrix} = \mathrm{z}\mathtt{K}^{-1} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \mathrm{z}\mathtt{K}^{-1}\mathbf{x}$$

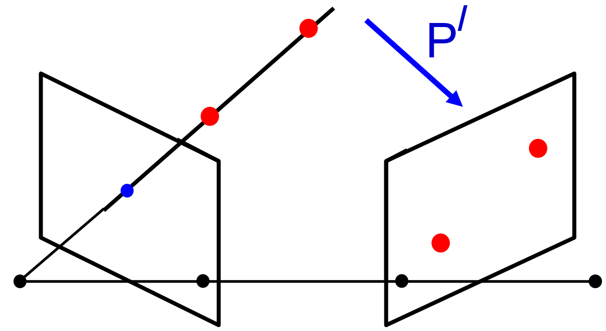
where $\mathbf{Z}$ is the point's depth, since

$$\mathrm{X}(\mathrm{z}) = \begin{pmatrix} \mathrm{z}\mathtt{K}^{-1}\mathbf{x} \\ 1 \end{pmatrix}$$

satisfies

$$\mathtt{P}\mathrm{X}(\mathrm{z}) = \mathtt{K}[\mathtt{I} \mid \mathbf{0}]\mathrm{X}(\mathrm{z}) = \mathbf{x}$$

Step 2: choose two points on the ray and project into the second image with camera P'



Consider two points on the ray $X(z) = \begin{pmatrix} zK^{-1}\mathbf{x} \\ 1 \end{pmatrix}$

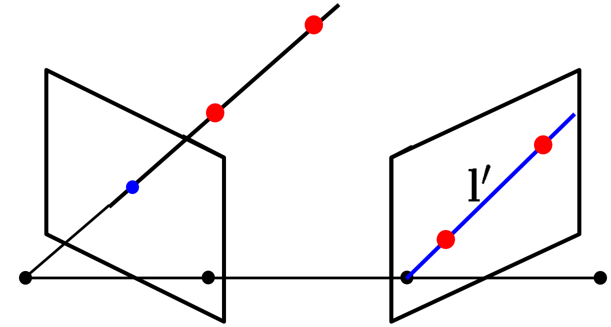
- z = 0 is the camera centre $\begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix}$
- z = ∞ is the point at infinity $\begin{pmatrix} K^{-1}\mathbf{x} \\ 0 \end{pmatrix}$

Project these two points into the second view

$$P' \begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix} = K'[R \mid \mathbf{t}] \begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix} = K'\mathbf{t}$$

$$P' \begin{pmatrix} K^{-1}\mathbf{x} \\ 0 \end{pmatrix} = K'[R \mid \mathbf{t}] \begin{pmatrix} K^{-1}\mathbf{x} \\ 0 \end{pmatrix} = K'RK^{-1}\mathbf{x}$$

Step 3: compute the line through the two image points using the relation $\mathbf{l}' = \mathbf{p} \times \mathbf{q}$



Compute the line through the points $\mathbf{l}' = (\mathtt{K}'\mathbf{t}) \times (\mathtt{K}'\mathtt{R}\mathtt{K}^{-1}\mathbf{x})$

Using the identity $(\mathtt{M}\mathbf{a}) \times (\mathtt{M}\mathbf{b}) = \mathtt{M}^{-\top}(\mathbf{a} \times \mathbf{b})$ where $\mathtt{M}^{-\top} = (\mathtt{M}^{-1})^{\top} = (\mathtt{M}^{\top})^{-1}$

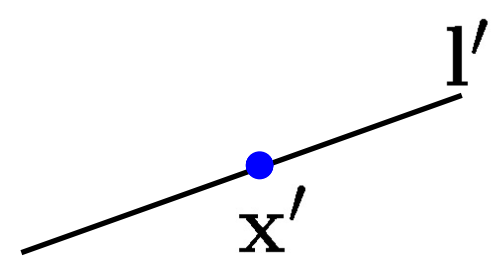
$$\mathbf{l}' = \mathtt{K}'^{-\top}\left(\mathbf{t} \times (\mathtt{R}\mathtt{K}^{-1}\mathbf{x})\right) = \underbrace{\mathtt{K}'^{-\top}[\mathbf{t}]_{\times}\mathtt{R}\mathtt{K}^{-1}}_{\mathrm{F}}\mathbf{x}$$

F is the fundamental matrix

$$\boxed{\mathbf{l}' = \mathtt{F}\mathbf{x} \qquad \mathtt{F} = \mathtt{K}'^{-\top}[\mathbf{t}]_{\times}\mathtt{R}\mathtt{K}^{-1}}$$

Points $\mathbf{x}$ and $\mathbf{x}'$ correspond ($\mathbf{x} \leftrightarrow \mathbf{x}'$) then $\mathbf{x}'^{\top}\mathbf{l}' = 0$

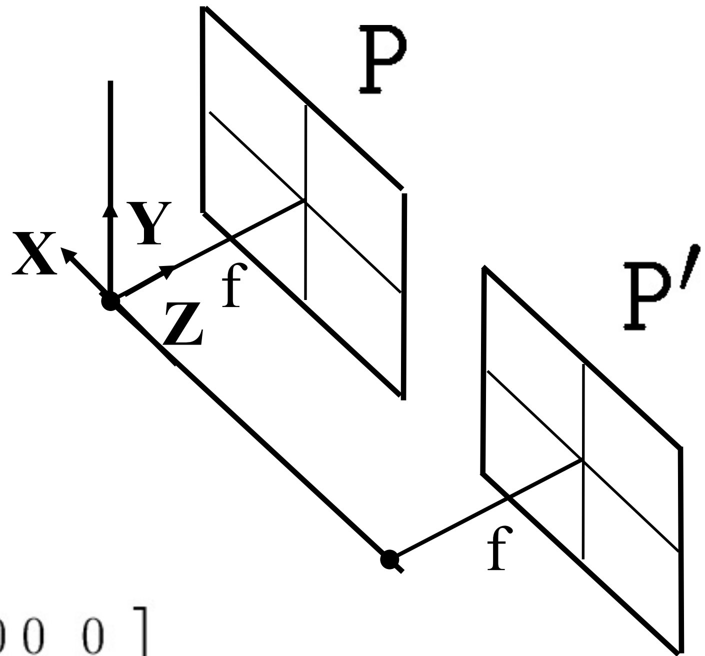
$$\boxed{\mathbf{x}'^{\top}\mathtt{F}\mathbf{x} = 0}$$



Example I: compute the fundamental matrix for a parallel camera stereo rig

$$\mathtt{P} = \mathtt{K}[\mathtt{I} \mid \mathbf{0}] \qquad \mathtt{P}' = \mathtt{K}'[\mathtt{R} \mid \mathbf{t}]$$

$$\mathtt{K} = \mathtt{K}' = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathtt{R} = \mathtt{I} \quad \mathbf{t} = \begin{pmatrix} t_x \\ 0 \\ 0 \end{pmatrix}$$



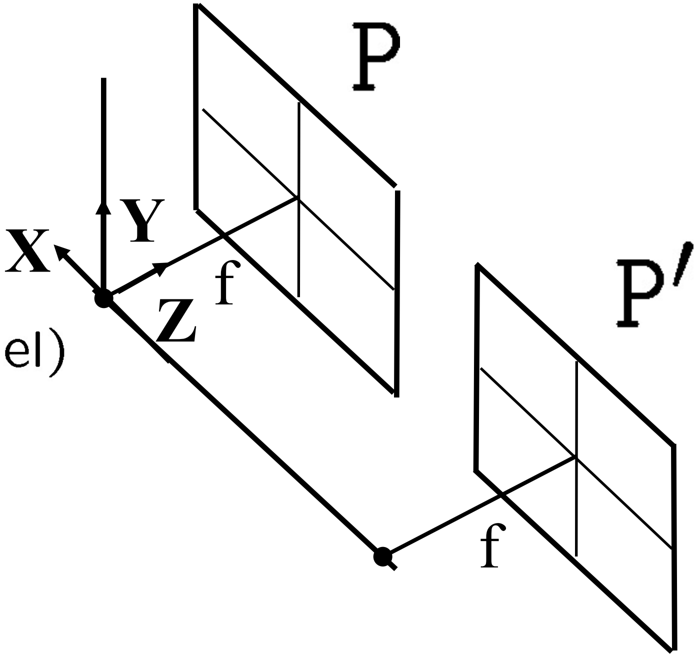
$$\mathtt{F} = \mathtt{K}'^{-\top}[\mathbf{t}]_\times \mathtt{R}\mathtt{K}^{-1}$$

$$= \begin{bmatrix} 1/f & 0 & 0 \\ 0 & 1/f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -t_x \\ 0 & t_x & 0 \end{bmatrix} \begin{bmatrix} 1/f & 0 & 0 \\ 0 & 1/f & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$\mathbf{x}'^\top \mathtt{F} \mathbf{x} = \begin{pmatrix} x' & y' & 1 \end{pmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0$$

- reduces to $y = y'$, i.e. raster correspondence (horizontal scan-lines)

$\mathtt{F}$ is a rank 2 matrix



The epipole $\mathbf{e}$ is the null-space vector (kernel) of $\mathtt{F}$ (exercise), i.e. $\mathtt{F}\mathbf{e} = \mathbf{0}$

In this case

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \mathbf{0}$$

so that

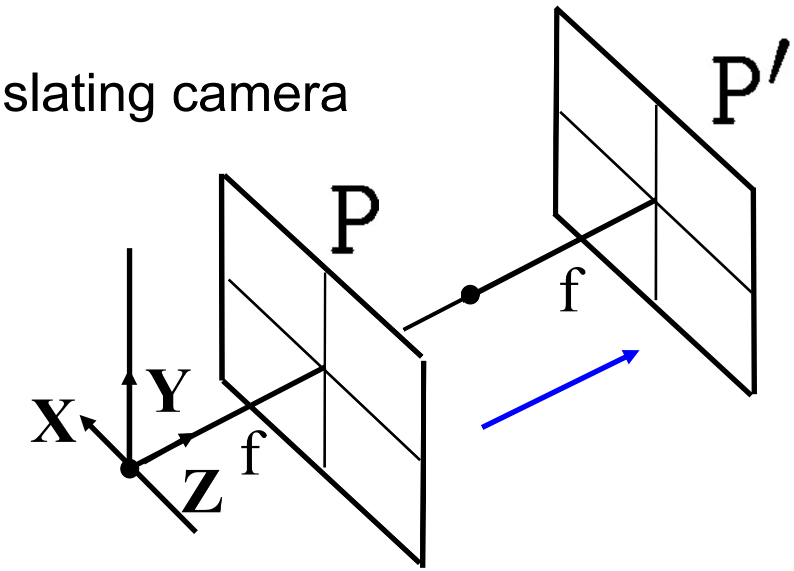
$$\mathbf{e} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

Geometric interpretation ?

Example II: compute F for a forward translating camera

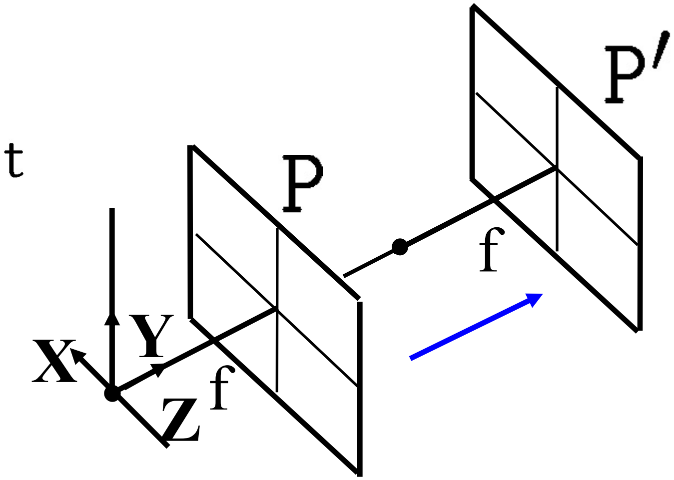
$$\mathtt{P} = \mathtt{K}[\mathtt{I} \mid \mathbf{0}] \qquad \mathtt{P}' = \mathtt{K}'[\mathtt{R} \mid \mathbf{t}]$$

$$\mathtt{K} = \mathtt{K}' = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \mathtt{R} = \mathtt{I} \quad \mathbf{t} = \begin{pmatrix} 0 \\ 0 \\ t_z \end{pmatrix}$$



$$\begin{aligned} \mathtt{F} &= \mathtt{K}'^{-\top}[\mathbf{t}]_\times \mathtt{R}\mathtt{K}^{-1} \\ &= \begin{bmatrix} 1/f & 0 & 0 \\ 0 & 1/f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & -t_z & 0 \\ t_z & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1/f & 0 & 0 \\ 0 & 1/f & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{aligned}$$

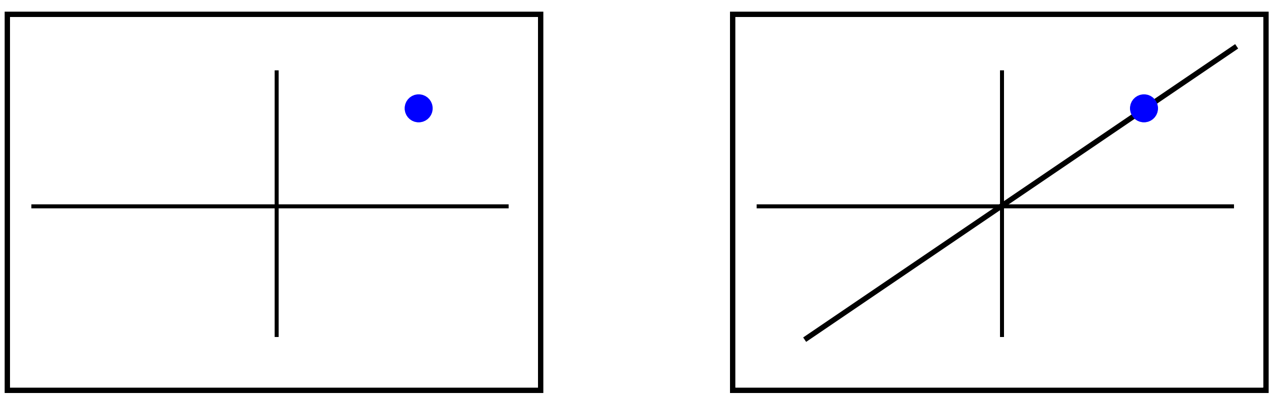
From $\mathbf{l}' = \mathtt{F}\mathbf{x}$ the epipolar line for the point $\mathbf{x} = (x, y, 1)^\top$ is



$$\mathbf{l}' = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} -y \\ x \\ 0 \end{pmatrix}$$

The points $(x, y, 1)^\top$ and $(0, 0, 1)^\top$ lie on this line

first image   second image




OXFORD


OXFORD

Summary: Properties of the Fundamental matrix

- $\mathtt{F}$ is a rank 2 homogeneous matrix with 7 degrees of freedom.

- Point correspondence:

  if $\mathbf{x}$ and $\mathbf{x}'$ are corresponding image points, then $\mathbf{x}'^\top \mathtt{F}\mathbf{x} = 0$.

- Epipolar lines:

  ◇ $\mathbf{l}' = \mathtt{F}\mathbf{x}$ is the epipolar line corresponding to $\mathbf{x}$.

  ◇ $\mathbf{l} = \mathtt{F}^\top \mathbf{x}'$ is the epipolar line corresponding to $\mathbf{x}'$.

- Epipoles:

  ◇ $\mathtt{F}\mathbf{e} = \mathbf{0}$.

  ◇ $\mathtt{F}^\top \mathbf{e}' = \mathbf{0}$.

- Computation from camera matrices $\mathtt{P}, \mathtt{P}'$:

  $\mathtt{P} = \mathtt{K}[\mathtt{I} \mid \mathbf{0}],\ \mathtt{P}' = \mathtt{K}'[\mathtt{R} \mid \mathbf{t}],\ \mathtt{F} = \mathtt{K}'^{-\top}[\mathbf{t}]_\times \mathtt{R}\mathtt{K}^{-1}$

Admin Interlude

Class Project details

On Thurs Dec 16^{th}:

1. Presentation session (7.10pm)
   2 Slides / 2 mins per team
   Google slide deck
   Upload to Google folder here

2. Project report (5-8 pages)
   I will be grading, not TAs
   Upload to Google folder here
   One upload per team

Stereo correspondence algorithms

Problem statement

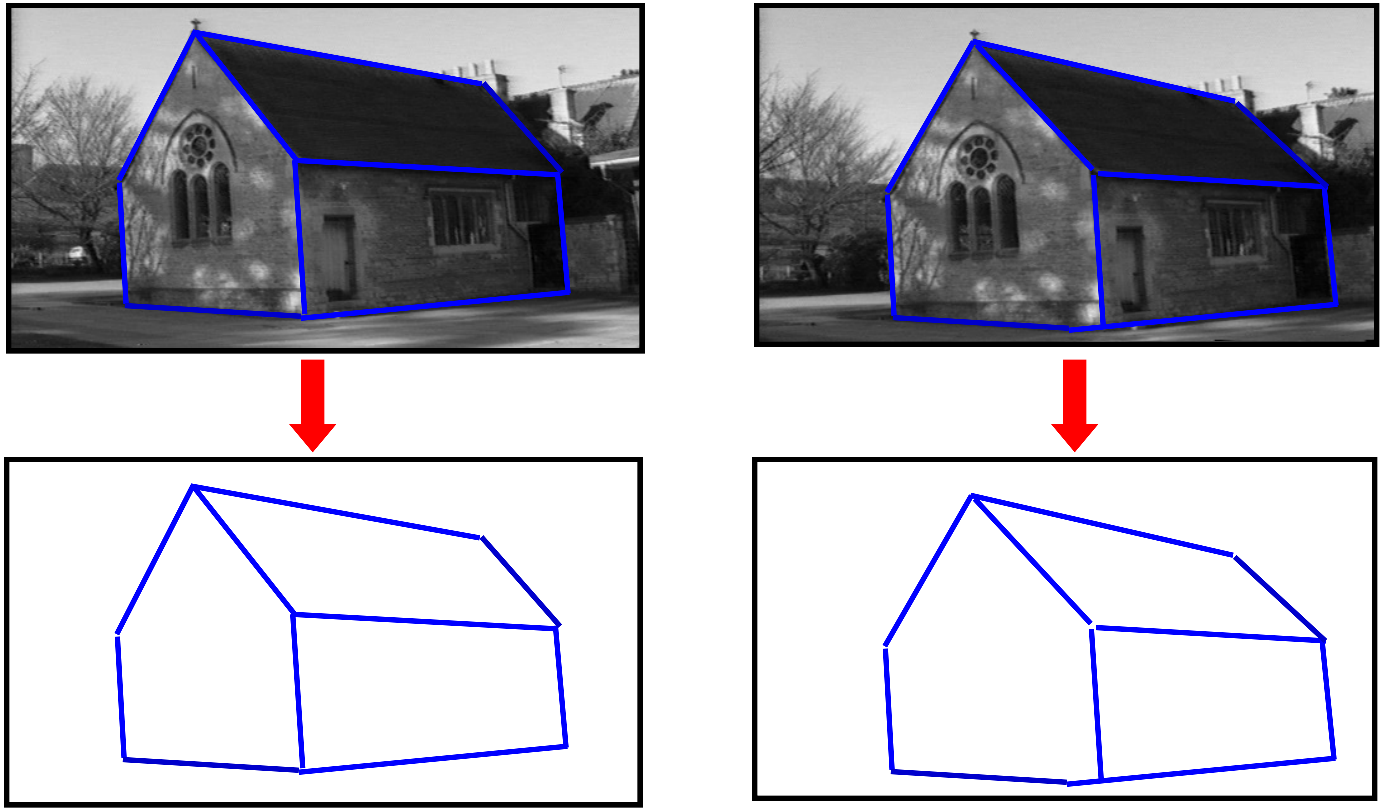
Given: two images and their associated cameras compute corresponding image points.

Algorithms may be classified into two types:

1. Dense: compute a correspondence at every pixel
2. Sparse: compute correspondences only for features

The methods may be top down or bottom up

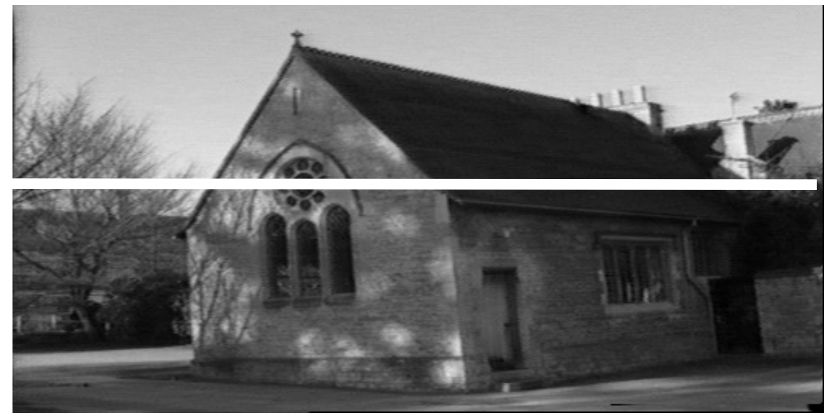
Top down matching



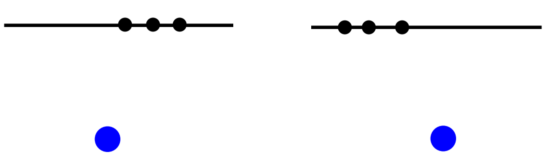
1. Group model (house, windows, etc) independently in each image
2. Match points (vertices) between images

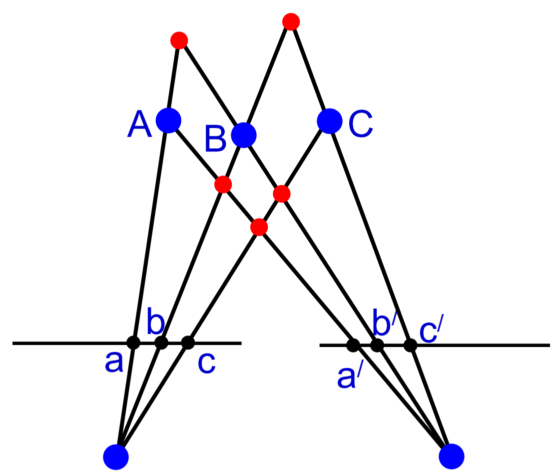
Bottom up matching

- epipolar geometry reduces the correspondence search from 2D to a 1D search on corresponding epipolar lines



- 1D correspondence problem





Correspondence algorithms

Algorithms may be top down or bottom up – random dot stereograms are an existence proof that bottom up algorithms are possible

From here on only consider bottom up algorithms

Algorithms may be classified into two types:

1. Dense: compute a correspondence at every pixel
2. Sparse: compute correspondences only for features

Example image pair – parallel cameras

First image

Second image

Dense correspondence algorithm

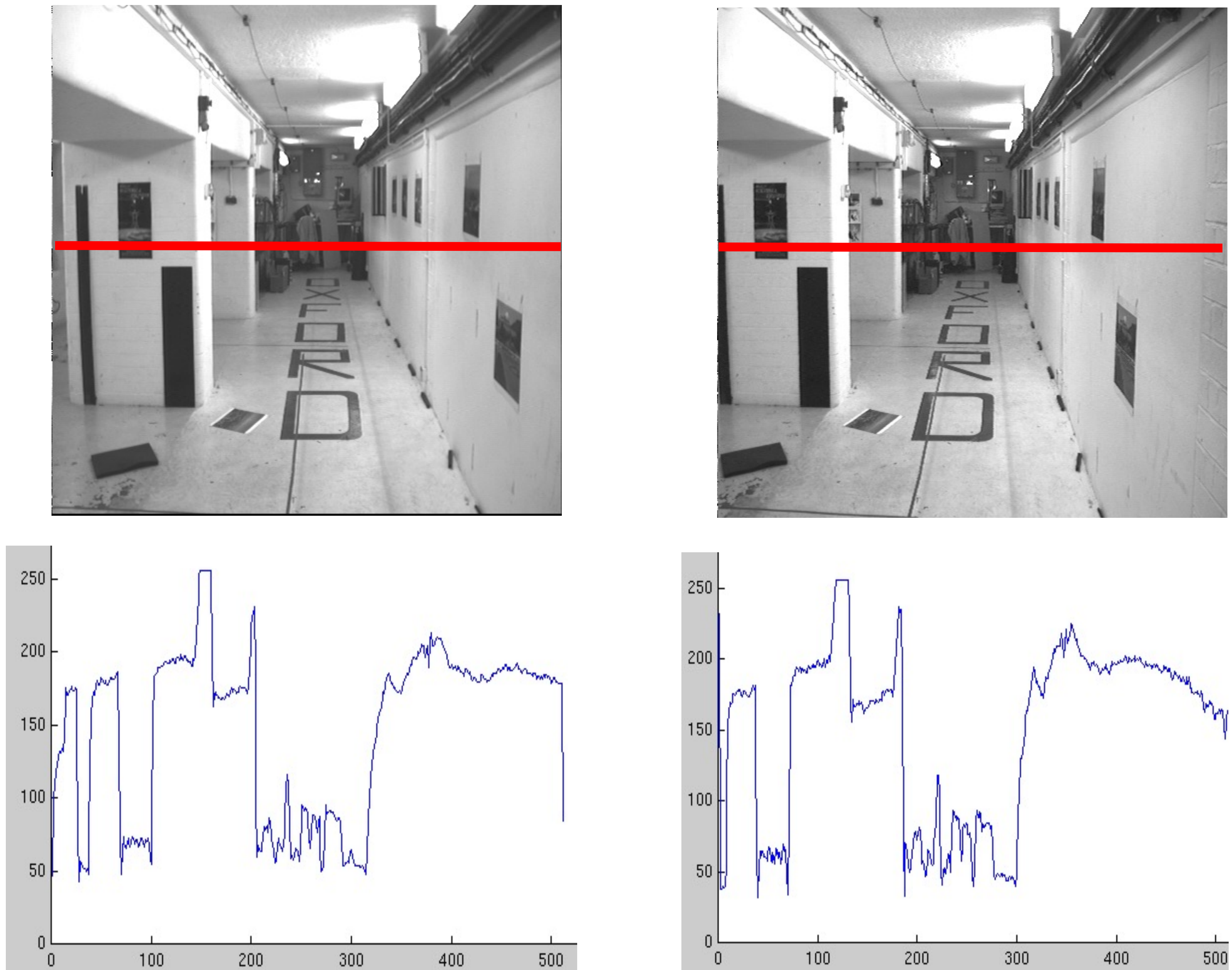
Parallel camera example – epipolar lines are corresponding rasters

Search problem (geometric constraint): for each point in the left image, the corresponding point in the right image lies on the epipolar line (1D ambiguity)

Disambiguating assumption (photometric constraint): the intensity neighbourhood of corresponding points are similar across images

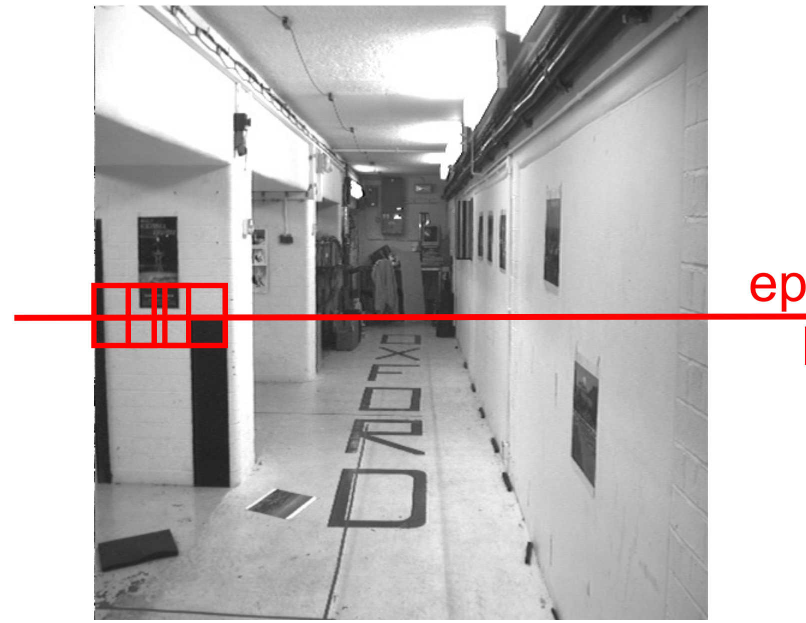
Measure similarity of neighbourhood intensity by cross-correlation

Intensity profiles



- Clear correspondence between intensities, but also noise and ambiguity

Cross-correlation of neighbourhood regions



regions A, B, write as vectors $\mathbf{a}, \mathbf{b}$

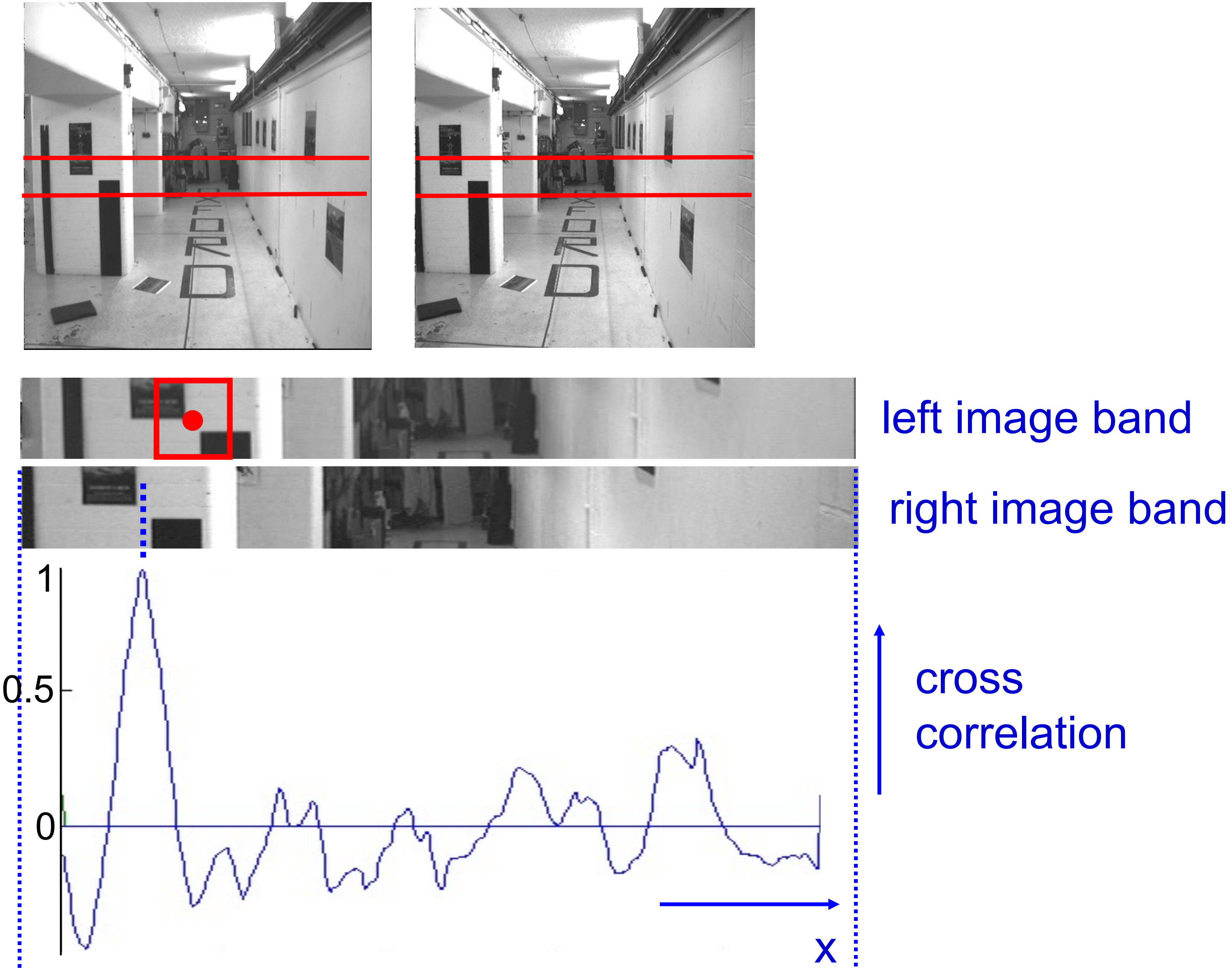
translate so that mean is zero

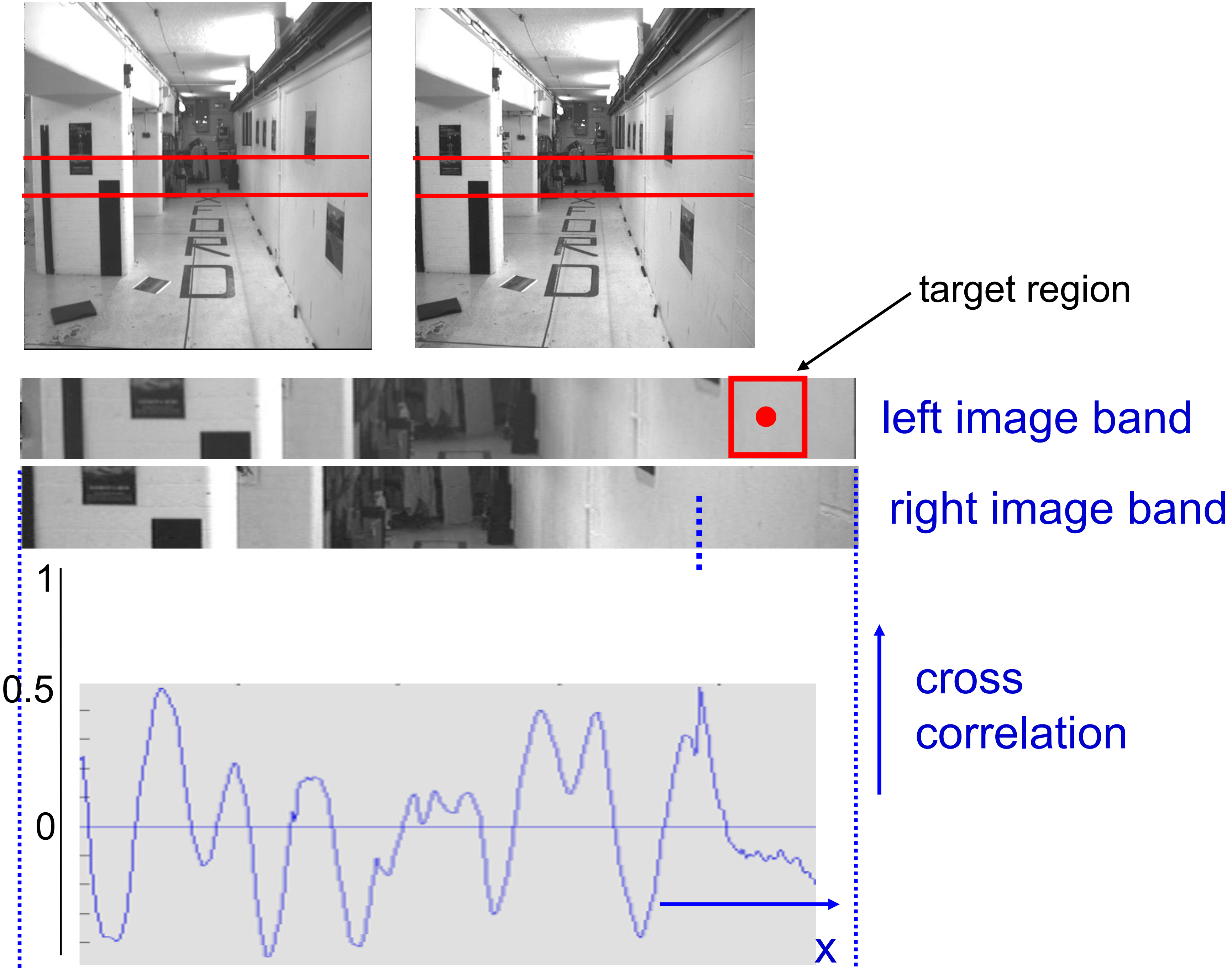
$$\mathtt{a} \rightarrow \mathtt{a} - \langle \mathbf{a} \rangle, \quad \mathtt{b} \rightarrow \mathtt{b} - \langle \mathbf{b} \rangle$$

$$\text{cross correlation} = \frac{\mathbf{a}.\mathbf{b}}{|\mathbf{a}||\mathbf{b}|}$$

Invariant to $I \rightarrow \alpha I + \beta$

(exercise)

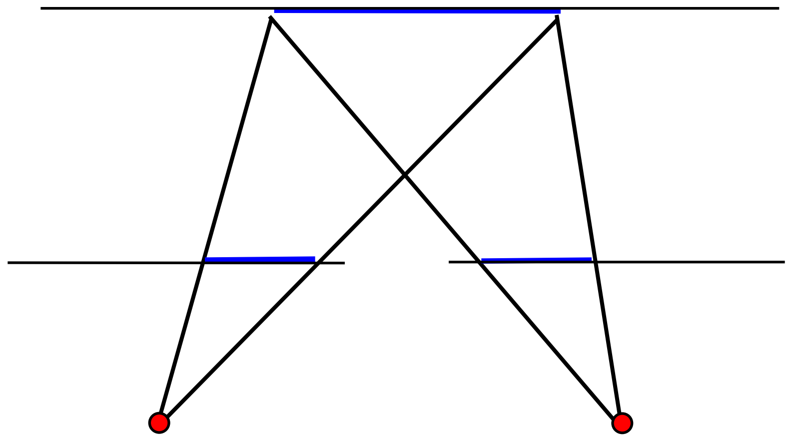

left image band
right image band
1
0.5
0
cross
correlation
x


target region
left image band
right image band
1
0.5
0
cross
correlation
x

Why is cross-correlation such a poor measure in the second case?

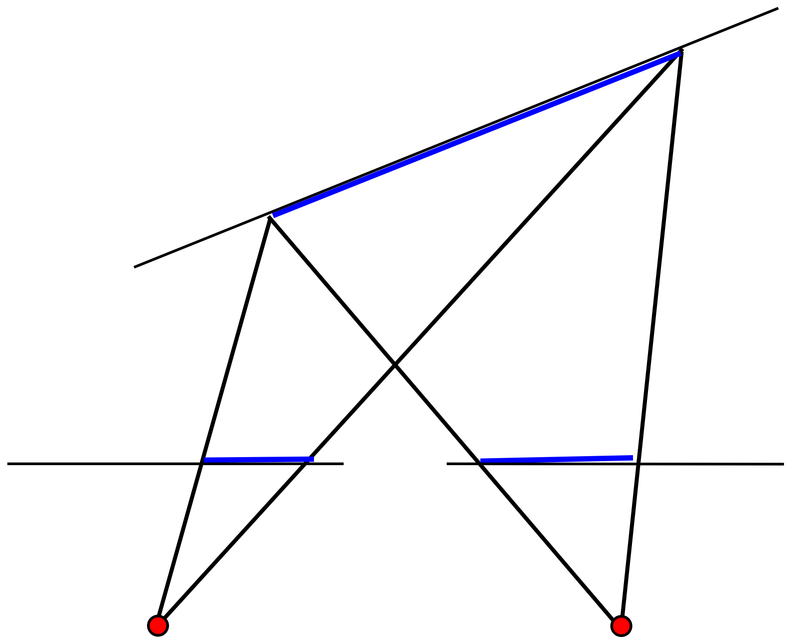
1. The neighbourhood region does not have a "distinctive" spatial intensity distribution

2. Foreshortening effects



fronto-parallel surface

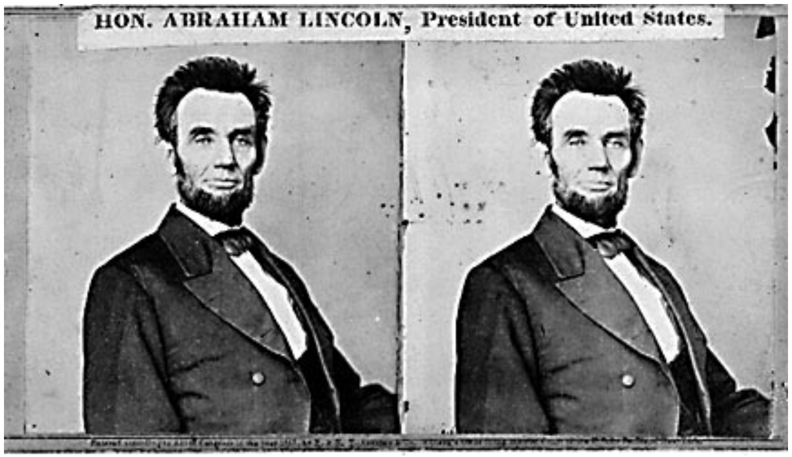
imaged length the same



slanting surface

imaged lengths differ

Limitations of similarity constraint



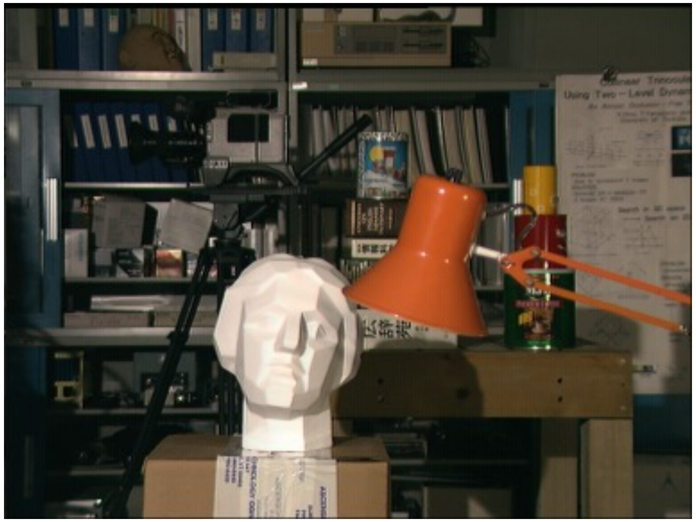
Textureless surfaces

Occlusions, repetition

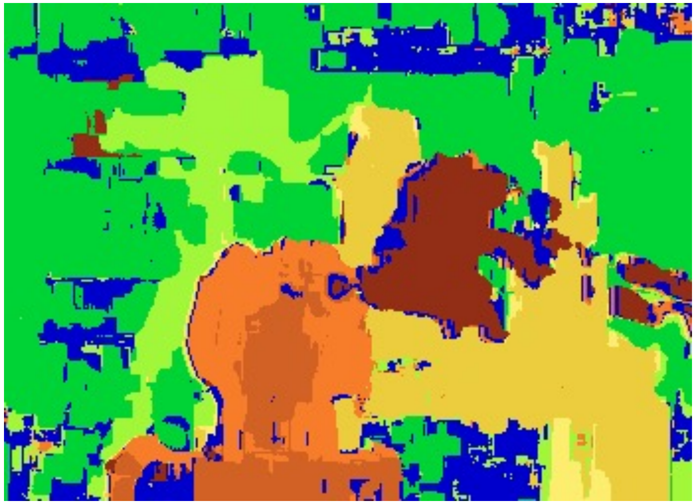
Non-Lambertian surfaces, specularities

Results with window search

Data



Window-based matching



Ground truth

Sketch of a dense correspondence algorithm

For each pixel in the left image

- compute the neighbourhood cross correlation along the corresponding epipolar line in the right image
- the corresponding pixel is the one with the highest cross correlation

Parameters

- size (scale) of neighbourhood
- search disparity

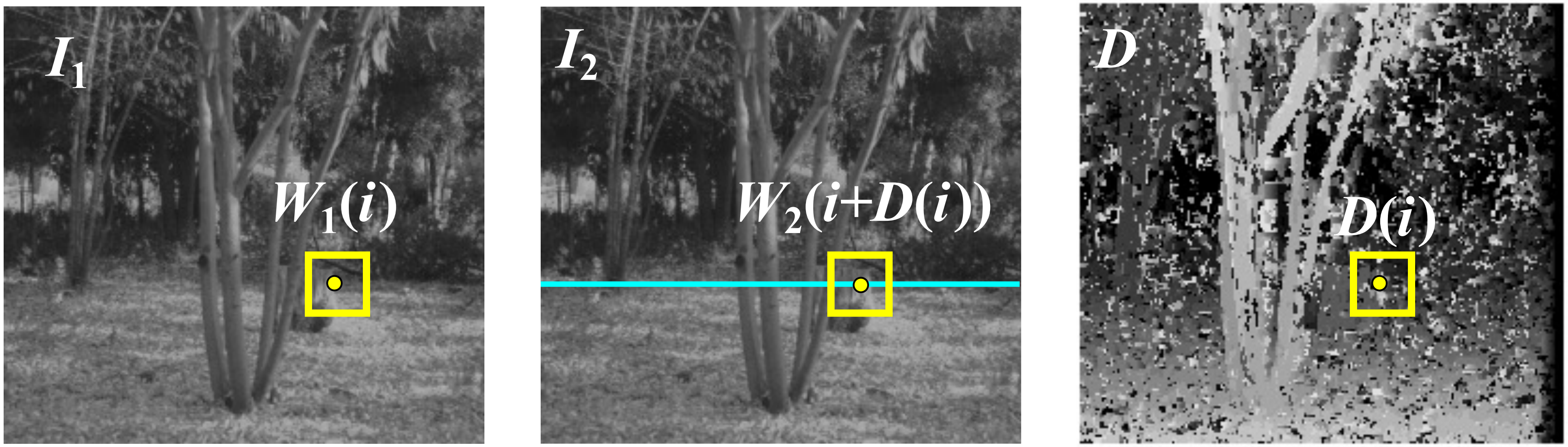
Other constraints

- uniqueness
- ordering
- smoothness of disparity field

Applicability

- textured scene, largely fronto-parallel

Stereo matching as energy minimization



MAP estimate of disparity image *D*: $P(D \mid I_1, I_2) \propto P(I_1, I_2 \mid D) P(D)$

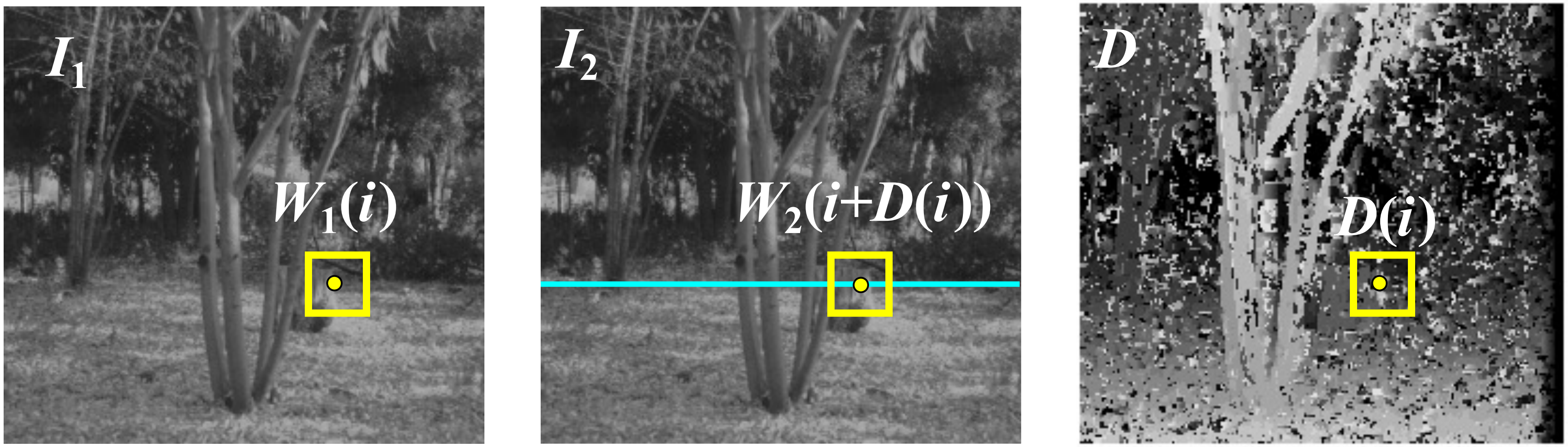
$$-\log P(D \mid I_1, I_2) \propto -\log P(I_1, I_2 \mid D) - \log P(D)$$

$$\boxed{E = \alpha E_{\text{data}}(I_1, I_2, D) + \beta E_{\text{smooth}}(D)}$$

$$\boxed{E_{\text{data}} = \sum_i \big(W_1(i) - W_2(i + D(i))\big)^2}$$

$$\boxed{E_{\text{smooth}} = \sum_{\text{neighbors } i,j} \rho\big(D(i) - D(j)\big)}$$

Stereo matching as energy minimization



$$E = \alpha E_{\text{data}}(I_1, I_2, D) + \beta E_{\text{smooth}}(D)$$

$$E_{\text{data}} = \sum_i \left(W_1(i) - W_2(i + D(i))\right)^2$$

$$E_{\text{smooth}} = \sum_{\text{neighbors } i,j} \rho\left(D(i) - D(j)\right)$$

- Energy functions of this form can be minimized using *graph cuts*

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

Graph cuts solution

Graph cuts

Ground truth

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

For the latest and greatest: http://www.middlebury.edu/stereo/

Example dense correspondence algorithm

left image

right image

3D reconstruction

right image

depth map

intensity = depth

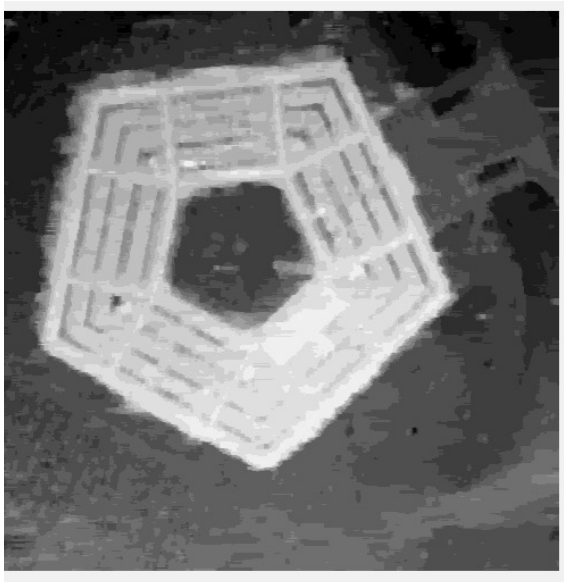
Texture mapped 3D triangulation

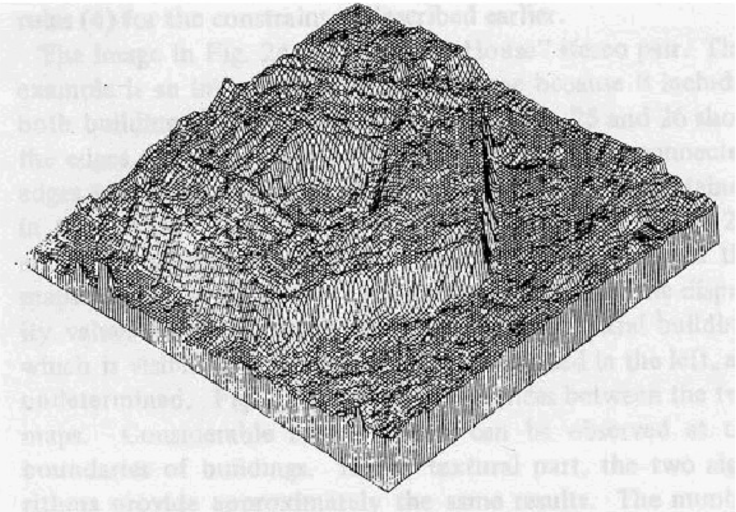
Pentagon example

left image

right image

range map

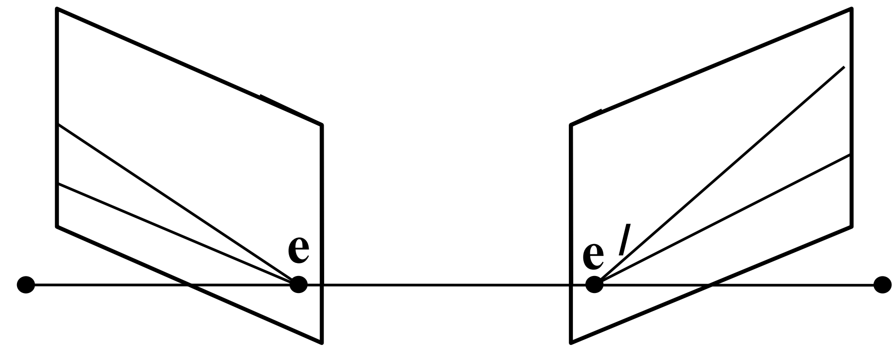


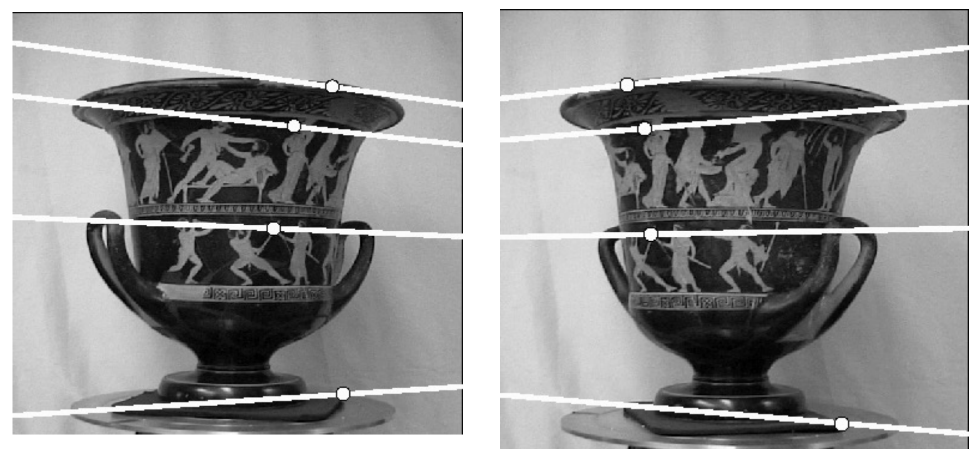


Rectification

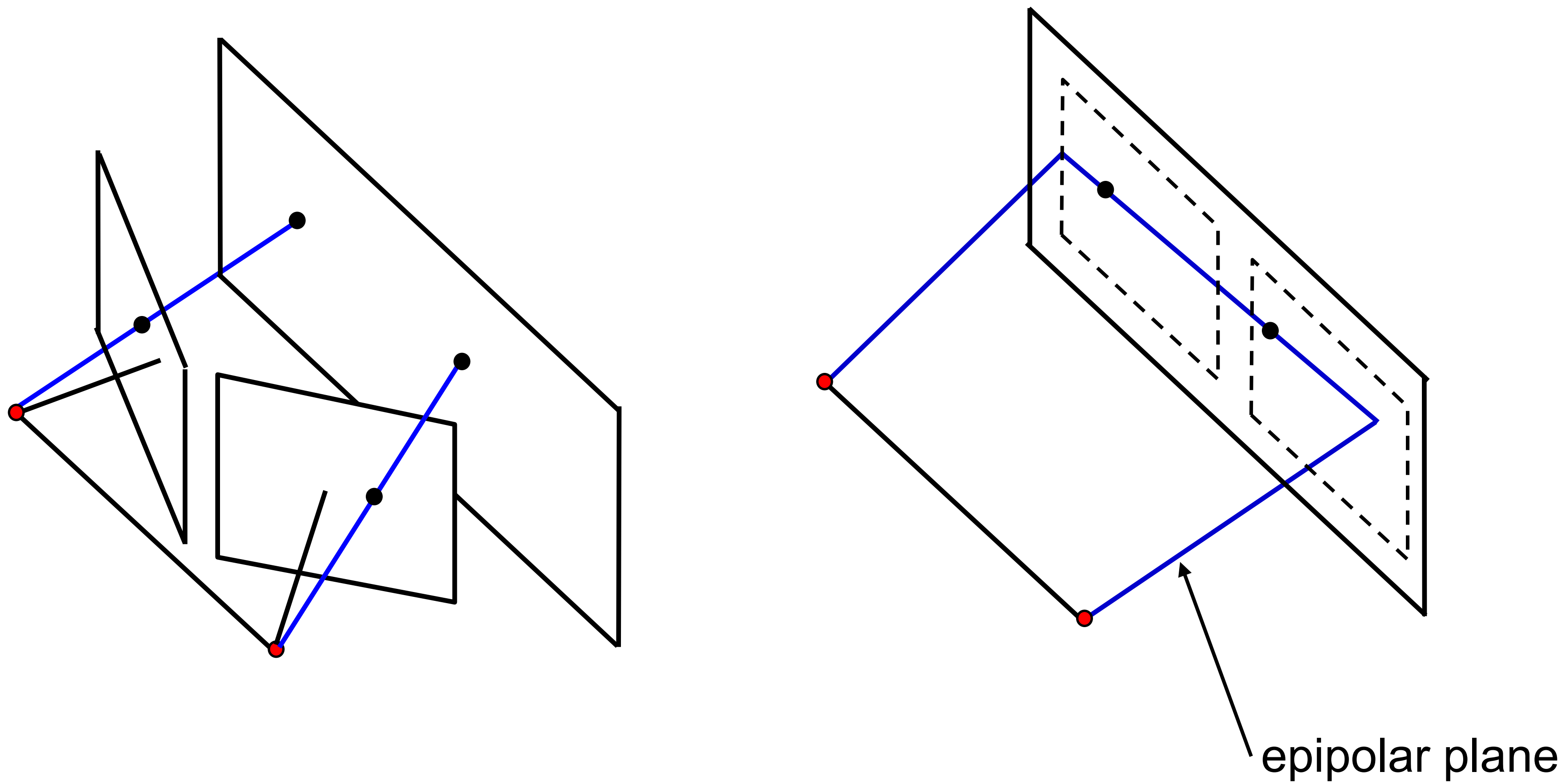
For converging cameras

- epipolar lines are not parallel





Project images onto plane parallel to baseline



Rectification continued

Convert converging cameras to parallel camera geometry by an image mapping

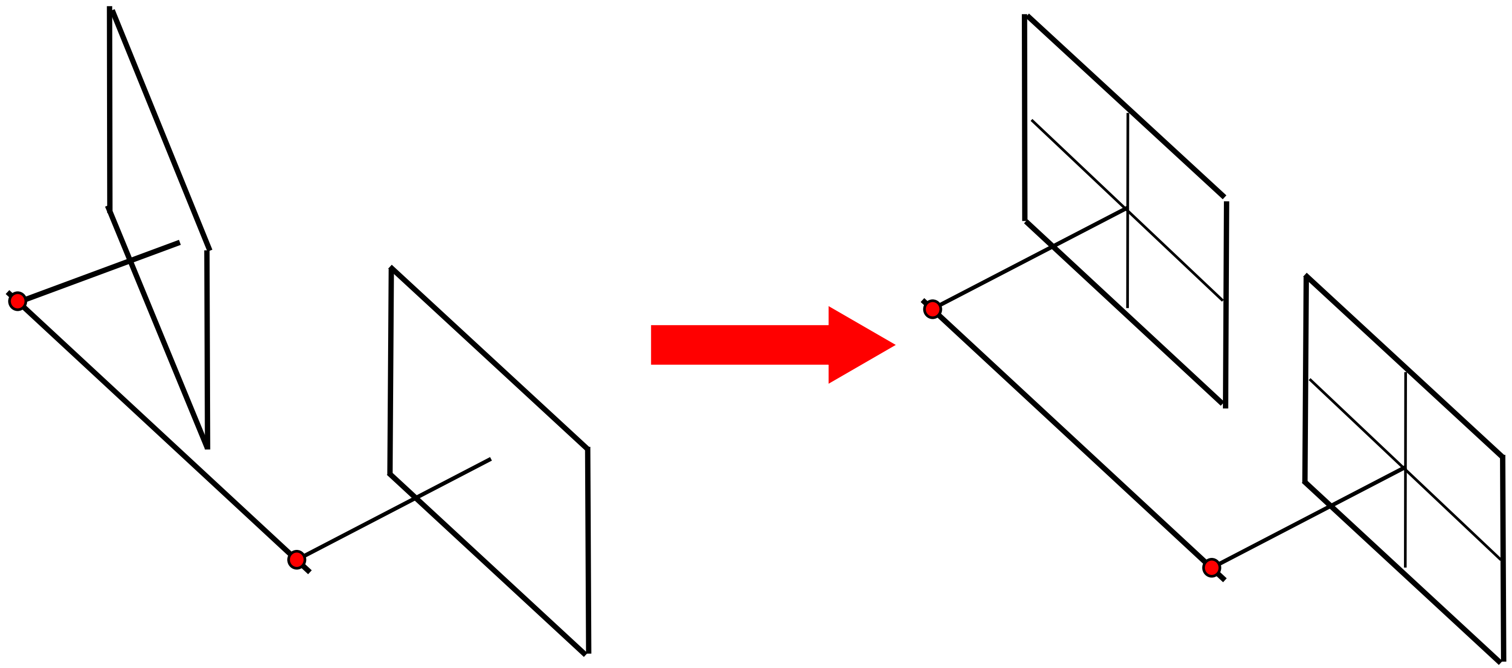


Image mapping is a 2D homography (projective transformation)

$$H = KRK^{-1}$$ (exercise)

Rectification continued

Convert converging cameras to parallel camera geometry by an image mapping

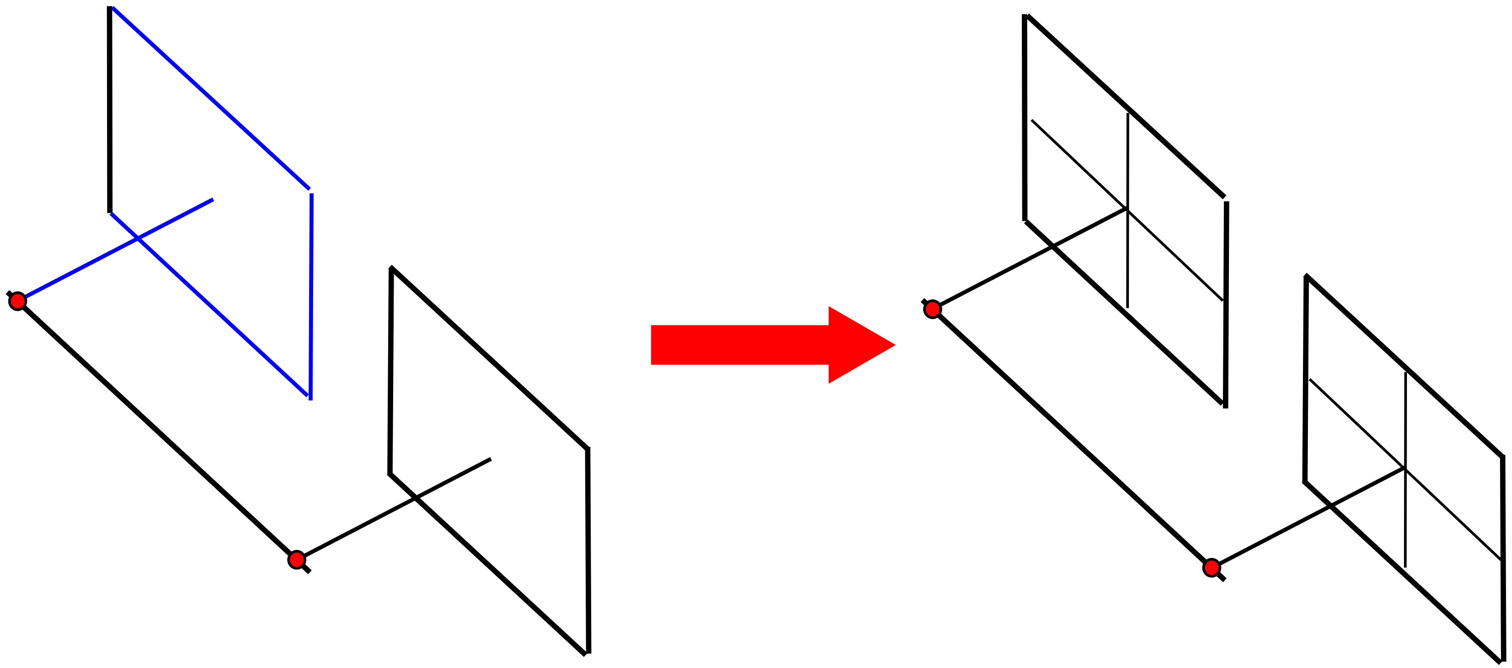
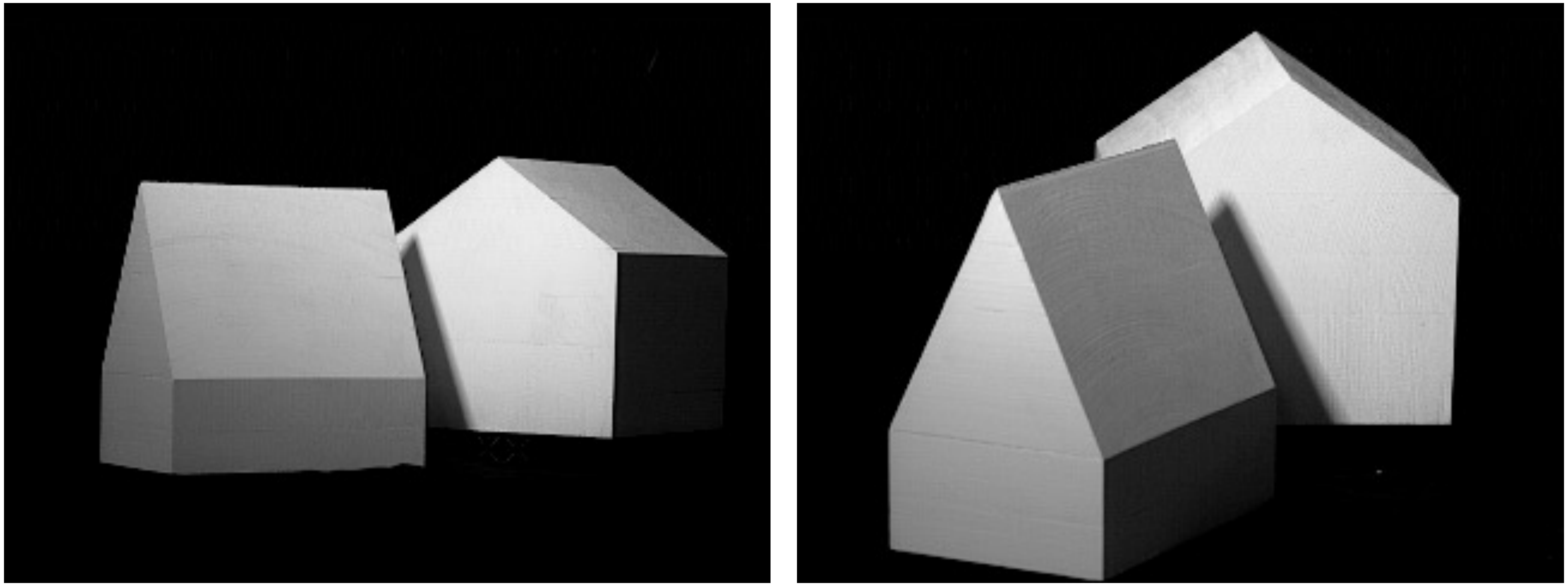


Image mapping is a 2D homography (projective transformation)

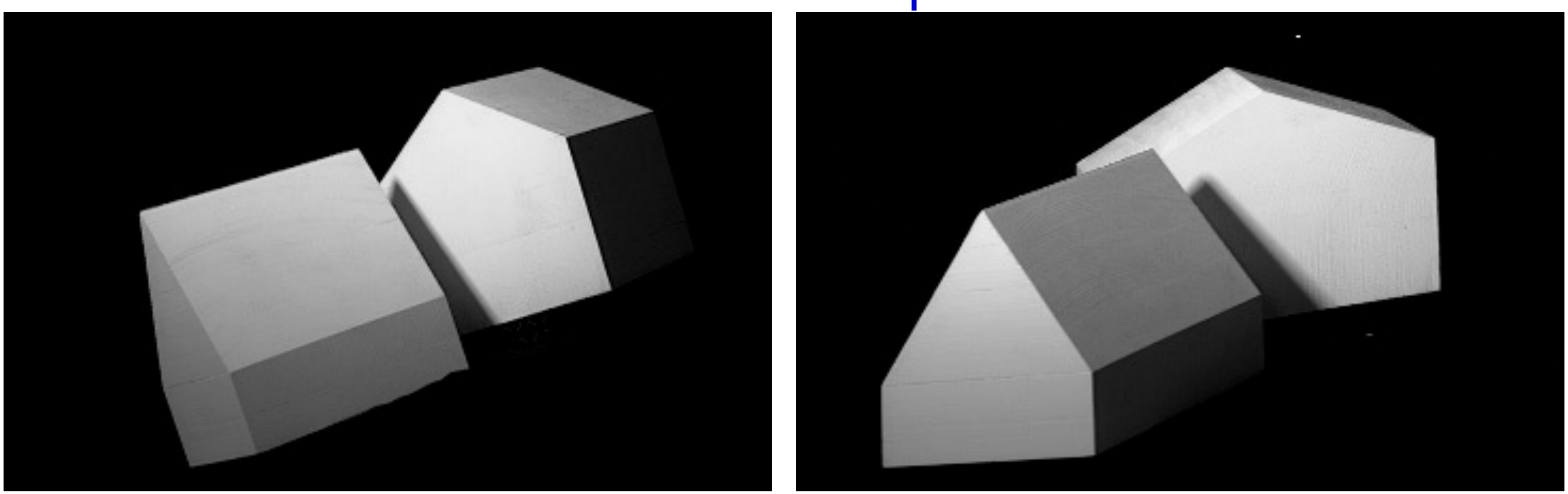
$\mathrm{H} = \mathrm{K}\mathrm{R}\mathrm{K}^{-1}$ (exercise)

Example

original stereo pair



rectified stereo pair



Example: depth and disparity for a parallel camera stereo rig

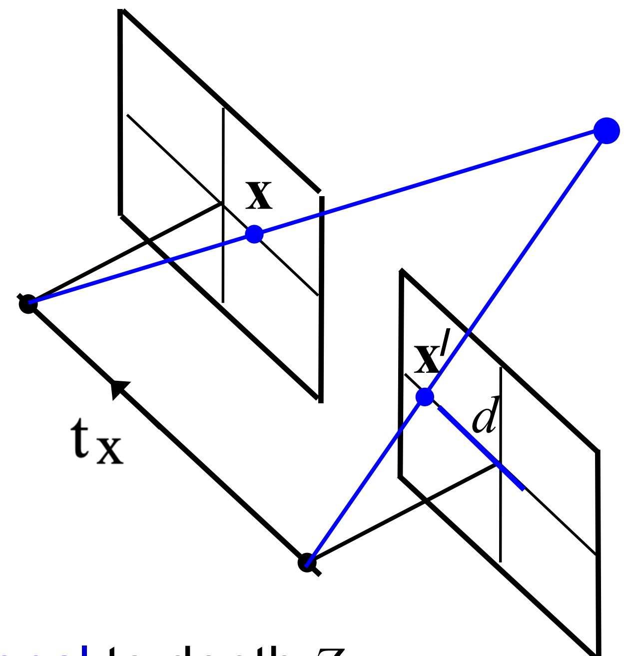
$$\mathtt{K} = \mathtt{K}' = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathtt{R} = \mathtt{I} \quad \mathbf{t} = \begin{pmatrix} t_x \\ 0 \\ 0 \end{pmatrix}$$

Then, $y' = y$, and the disparity $d = x' - x = \dfrac{f t_x}{Z}$

Derivation

$$\frac{x}{f} = \frac{X}{Z} \qquad \frac{x'}{f} = \frac{X + t_x}{Z}$$

$$\frac{x'}{f} = \frac{x}{f} + \frac{t_x}{Z}$$



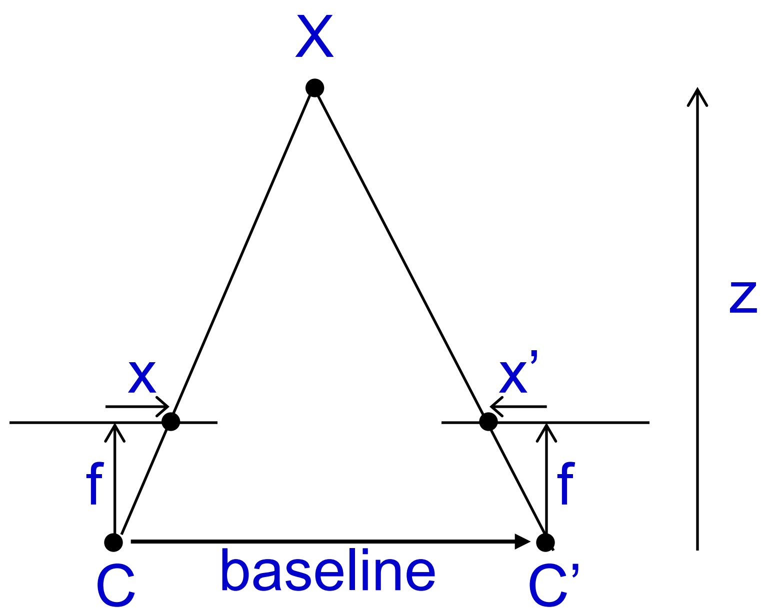
Note

- image movement (disparity) is inversely proportional to depth Z

  as $z \to \infty, \; d \to 0$

- depth is inversely proportional to disparity

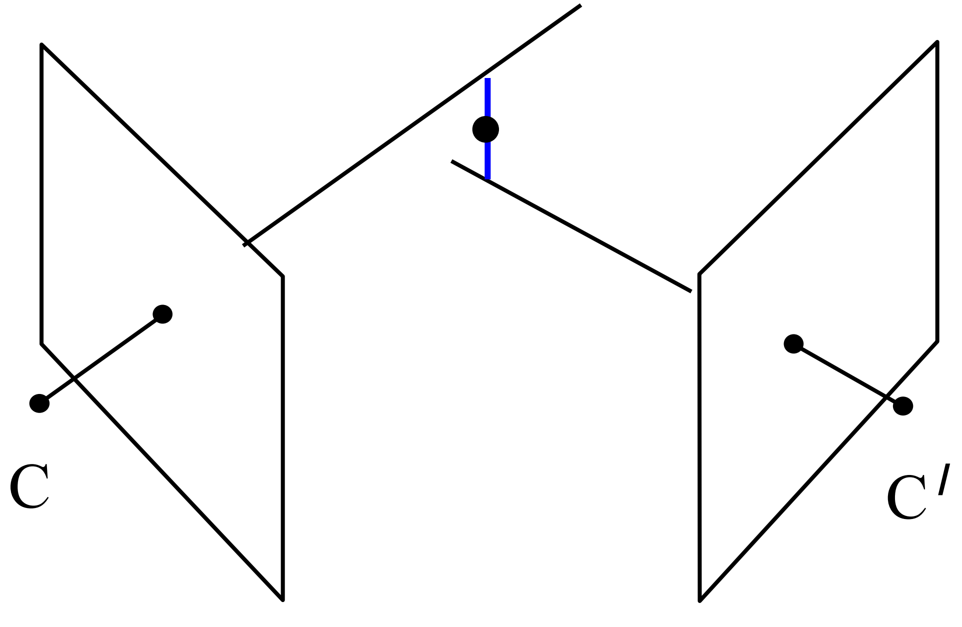
Depth from disparity



$$disparity = x - x' = \frac{baseline * f}{z}$$

Triangulation

1. Vector solution



Compute the mid-point of the shortest line between the two rays

2. Linear triangulation (algebraic solution)

Use the equations $\mathbf{x} = \mathtt{P}\mathbf{X}$ and $\mathbf{x}' = \mathtt{P}'\mathbf{X}$ to solve for $\mathbf{X}$

For the first camera:

$$\mathtt{P} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} = \begin{bmatrix} \mathbf{p}^{1\top} \\ \mathbf{p}^{2\top} \\ \mathbf{p}^{3\top} \end{bmatrix}$$

where $\mathbf{p}^{i\top}$ are the rows of $\mathtt{P}$

- eliminate unknown scale in $\lambda\mathbf{x} = \mathtt{P}\mathbf{X}$ by forming a cross product $\mathbf{x} \times (\mathtt{P}\mathbf{X}) = \mathbf{0}$

$$\begin{aligned} x(\mathbf{p}^{3\top}\mathbf{X}) - (\mathbf{p}^{1\top}\mathbf{X}) &= 0 \\ y(\mathbf{p}^{3\top}\mathbf{X}) - (\mathbf{p}^{2\top}\mathbf{X}) &= 0 \\ x(\mathbf{p}^{2\top}\mathbf{X}) - y(\mathbf{p}^{1\top}\mathbf{X}) &= 0 \end{aligned}$$

- rearrange as (first two equations only)

$$\begin{bmatrix} x\mathbf{p}^{3\top} - \mathbf{p}^{1\top} \\ y\mathbf{p}^{3\top} - \mathbf{p}^{2\top} \end{bmatrix} \mathbf{X} = \mathbf{0}$$

Similarly for the second camera:

$$\begin{bmatrix} x'\mathbf{p}'^{3\top} - \mathbf{p}'^{1\top} \\ y'\mathbf{p}'^{3\top} - \mathbf{p}'^{2\top} \end{bmatrix} \mathbf{X} = \mathbf{0}$$

Collecting together gives

$$\mathtt{A}\mathbf{X} = \mathbf{0}$$

where $\mathtt{A}$ is the 4×4 matrix

$$\mathtt{A} = \begin{bmatrix} x\mathbf{p}^{3\top} - \mathbf{p}^{1\top} \\ y\mathbf{p}^{3\top} - \mathbf{p}^{2\top} \\ x'\mathbf{p}'^{3\top} - \mathbf{p}'^{1\top} \\ y'\mathbf{p}'^{3\top} - \mathbf{p}'^{2\top} \end{bmatrix}$$

from which $\mathbf{X}$ can be solved up to scale.

Problem: does not minimize anything meaningful

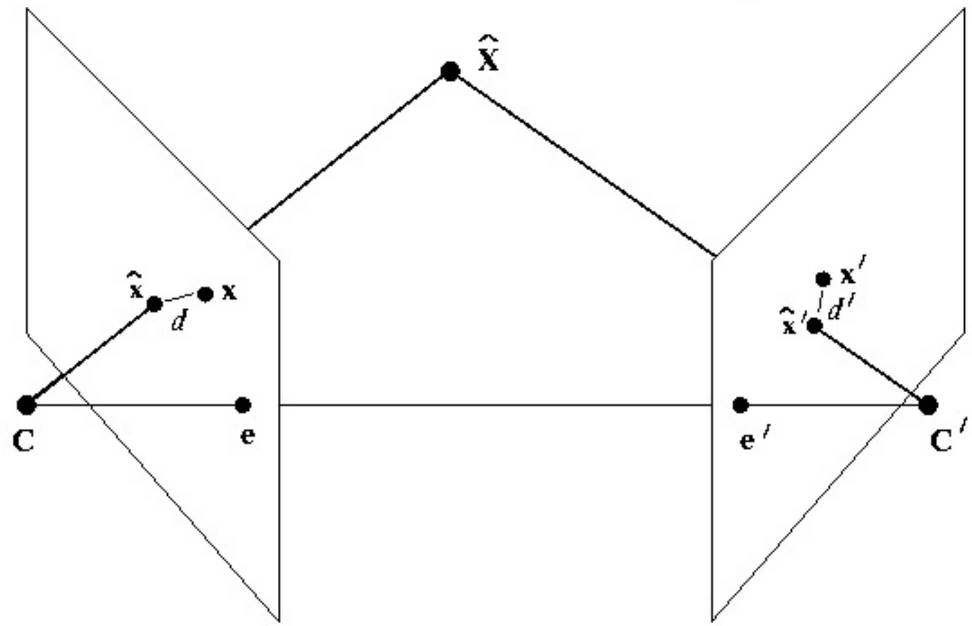
Advantage: extends to more than two views

3. Minimizing a geometric/statistical error

The idea is to estimate a 3D point $\widehat{\mathrm{X}}$ which exactly satisfies the supplied camera geometry, so it projects as

$$\hat{\mathrm{x}} = \mathrm{P}\widehat{\mathrm{X}} \qquad \hat{\mathrm{x}}' = \mathrm{P}'\widehat{\mathrm{X}}$$

and the aim is to estimate $\widehat{\mathrm{X}}$ from the image measurements $\mathbf{x}$ and $\mathbf{x}'$.



$$\min_{\widehat{\mathrm{X}}} \quad \mathcal{C}(\mathbf{x}, \mathbf{x}') = d(\mathbf{x}, \hat{\mathbf{x}})^2 + d(\mathbf{x}', \hat{\mathbf{x}}')^2$$

where $d(*, *)$ is the Euclidean distance between the points.

- It can be shown that if the measurement noise is Gaussian mean zero, $\sim N(0, \sigma^2)$, then minimizing geometric error is the Maximum Likelihood Estimate of X

- The minimization appears to be over three parameters (the position X), but the problem can be reduced to a minimization over one parameter

Different formulation of the problem

The minimization problem may be formulated differently:

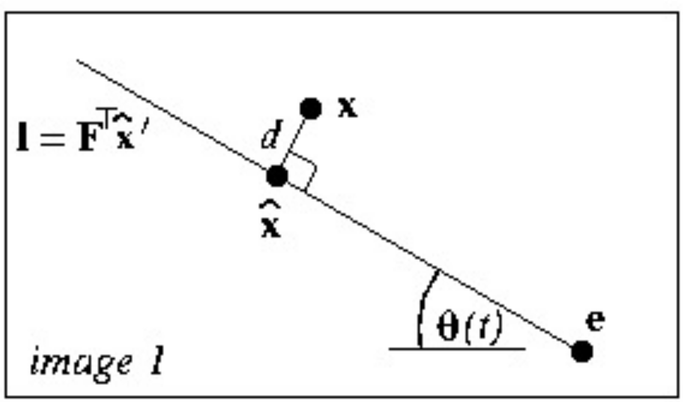
- Minimize

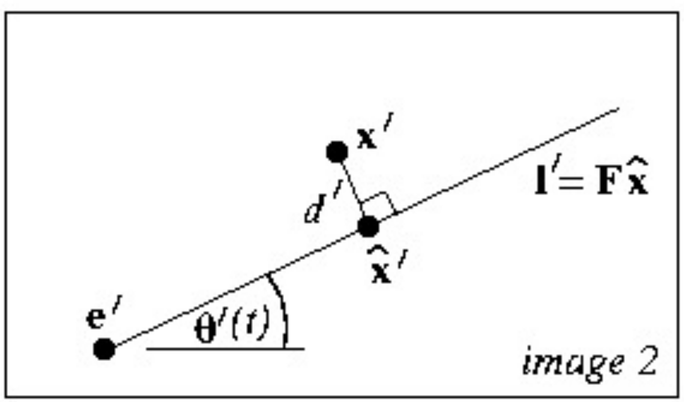
$$d(\mathbf{x}, \mathbf{l})^2 + d(\mathbf{x}', \mathbf{l}')^2$$

- $\mathbf{l}$ and $\mathbf{l}'$ range over all choices of corresponding epipolar lines.
- $\hat{\mathbf{x}}$ is the closest point on the line $\mathbf{l}$ to $\mathbf{x}$.
- Same for $\hat{\mathbf{x}}'$.

Minimization method

- Parametrize the pencil of epipolar lines in the first image by t, such that the epipolar line is $\mathbf{l}(t)$
- Using F compute the corresponding epipolar line in the second image $\mathbf{l}'(t)$
- Express the distance function $d(\mathbf{x}, \mathbf{l})^2 + d(\mathbf{x}', \mathbf{l}')^2$ explicitly as a function of t
- Find the value of t that minimizes the distance function
- Solution is a 6th degree polynomial in t





Typical Stereo Algorithm

- Define a matching cost function.
  - Sum of absolute differences.
  - The census transform.
- For each patch in the left image, search, along the epipolar line, for the patch in the right image with the smallest matching cost.
- Left image:

- Right image:

Zbontar & LeCun, Computing the Stereo Matching Cost with a Convolutional Neural Network, CVPR 2015.

- Learn the matching cost function.
  - Construct a binary classification dataset.
  - Use supervised learning.

Left patch	Right patch	Label
		Good match
		Bad match
⋮		⋮

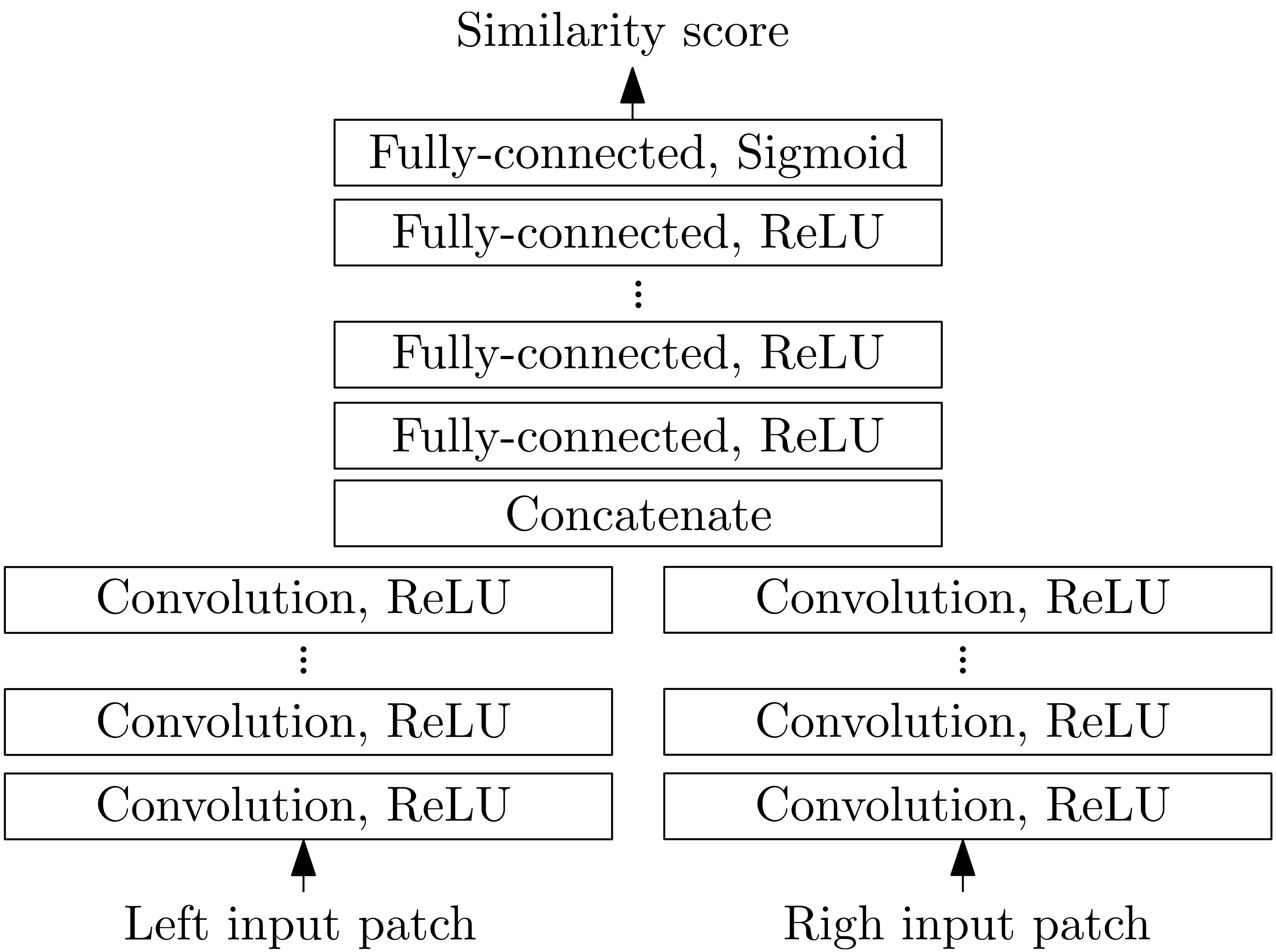
Constructing the Dataset

- One training example comprises two patches, one from the left and one from the right image:

$$< \mathcal{P}^{L}_{n\times n}(\mathbf{p}), \mathcal{P}^{R}_{n\times n}(\mathbf{q}) >$$

- $\mathcal{P}^{L}_{n\times n}(\mathbf{p})$ is a $n \times n$ patch from the left image, centered at $\mathbf{p} = (x, y)$
- The true disparity d is obtained from stereo datasets (KITTI and Middlebury).
- Positive example: $\mathbf{q} = (x - d, y)$
- Negative example: $\mathbf{q} = (x - d + o_{\text{neg}}, y)$
  - o_{neg} chosen randomly from $[-N_{\text{hi}}, -N_{\text{lo}}] \cup [N_{\text{lo}}, N_{\text{hi}}]$.
- N_{lo}, N_{hi}, and n are hyperparameters of the method.

The Accurate Architecture



The KITTI Stereo Dataset

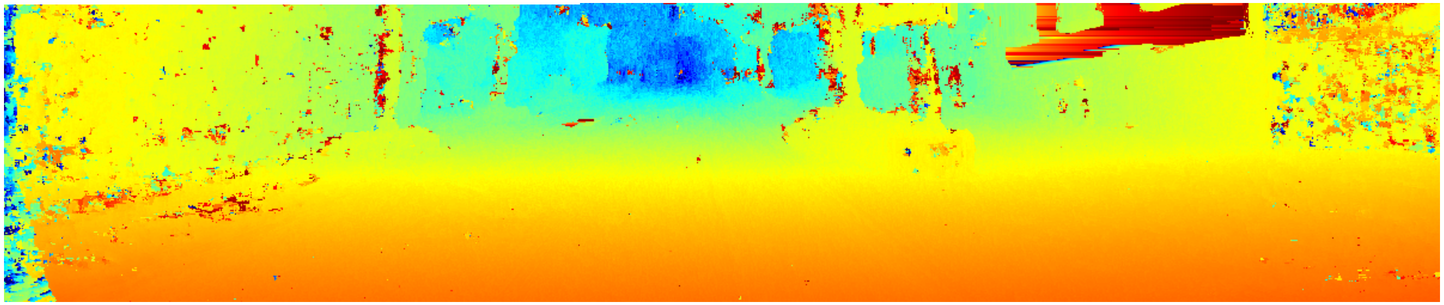
- Geiger et al. (2012). *Vision meets Robotics: The KITTI Dataset.*
- Menze, Geiger (2015). *Object Scene Flow for Autonomous Vehicles.*

- Ground truth is obtained by a LIDAR sensor.
- $\sim$200 training and $\sim$200 test image pairs at 1240×376.

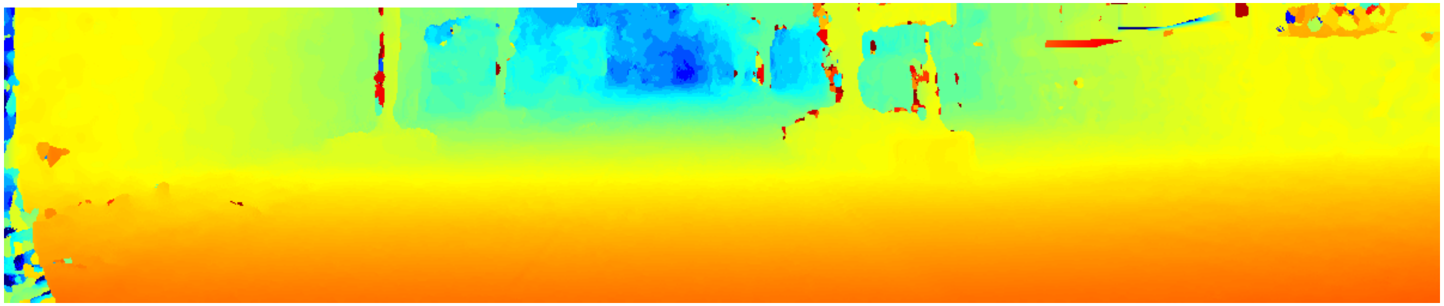
Cross-Based Cost Aggregation

- Left input image:

- Raw output from CNN

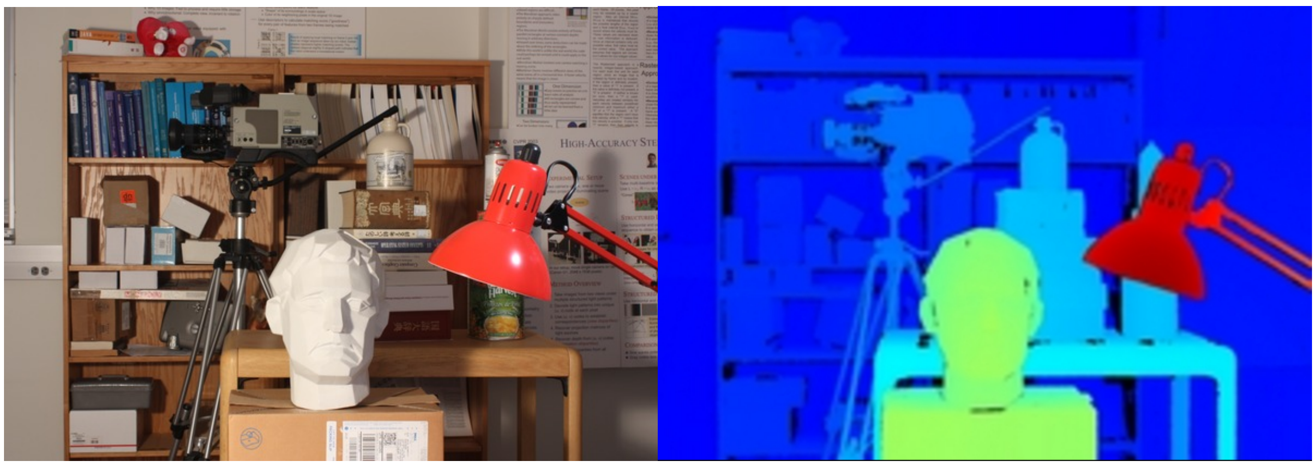


- After post-processing



The Middlebury Stereo Dataset

- Scharstein et al. (2014). *High-resolution stereo datasets with subpixel-accurate ground truth*.



- Ground truth is obtained by structured light.
- 60 training and 15 test image pairs at up to 3000×2000.

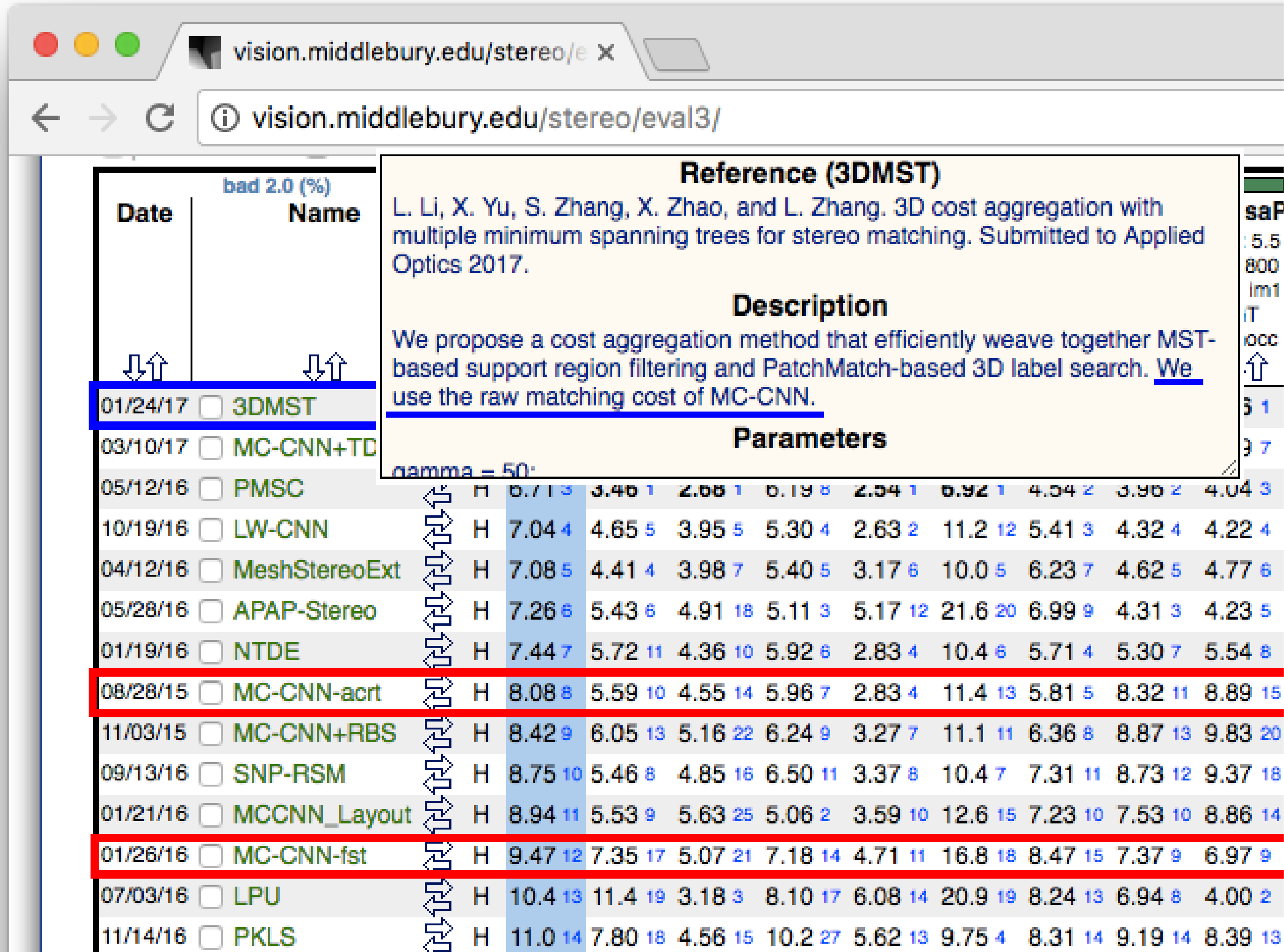
Results on the Middlebury stereo dataset ~2017

vision.middlebury.edu/stereo/eval3/

bad 2.0 (%)

Date	Name	Res	Weight Avg	Austr MP: 5.6 nd: 290 im0 im1 GT nonocc	AustrP MP: 5.6 nd: 290 im0 im1 GT nonocc	Bicyc2 MP: 5.6 nd: 250 im0 im1 GT nonocc	Class MP: 5.7 nd: 610 im0 im1 GT nonocc	ClassE MP: 5.7 nd: 610 im0 im1 GT nonocc	Compu MP: 1.5 nd: 256 im0 im1 GT nonocc	Crusa MP: 5.5 nd: 800 im0 im1 GT nonocc	CrusaP MP: 5.5 nd: 800 im0 im1 GT nonocc
01/24/17	3DMST	H	**5.92** 1	3.71 2	2.78 2	**4.75** 1	2.72 3	7.36 3	**4.28** 1	**3.44** 1	**3.76** 1
03/10/17	MC-CNN+TDSR	F	6.35 2	5.45 7	4.45 11	6.80 12	3.46 9	10.7 9	6.05 6	5.01 6	5.19 7
05/12/16	PMSC	H	6.71 3	**3.46** 1	**2.68** 1	6.19 8	**2.54** 1	**6.92** 1	4.54 2	3.96 2	4.04 3
10/19/16	LW-CNN	H	7.04 4	4.65 5	3.95 5	5.30 4	2.63 2	11.2 12	5.41 3	4.32 4	4.22 4
04/12/16	MeshStereoExt	H	7.08 5	4.41 4	3.98 7	5.40 5	3.17 6	10.0 5	6.23 7	4.62 5	4.77 6
05/28/16	APAP-Stereo	H	7.26 6	5.43 6	4.91 18	5.11 3	5.17 12	21.6 20	6.99 9	4.31 3	4.23 5
01/19/16	NTDE	H	7.44 7	5.72 11	4.36 10	5.92 6	2.83 4	10.4 6	5.71 4	5.30 7	5.54 8
08/28/15	MC-CNN-acrt	H	8.08 8	5.59 10	4.55 14	5.96 7	2.83 4	11.4 13	5.81 5	8.32 11	8.89 15
11/03/15	MC-CNN+RBS	H	8.42 9	6.05 13	5.16 22	6.24 9	3.27 7	11.1 11	6.36 8	8.87 13	9.83 20
09/13/16	SNP-RSM	H	8.75 10	5.46 8	4.85 16	6.50 11	3.37 8	10.4 7	7.31 11	8.73 12	9.37 18
01/21/16	MCCNN_Layout	H	8.94 11	5.53 9	5.63 25	5.06 2	3.59 10	12.6 15	7.23 10	7.53 10	8.86 14
01/26/16	MC-CNN-fst	H	9.47 12	7.35 17	5.07 21	7.18 14	4.71 11	16.8 18	8.47 15	7.37 9	6.97 9
07/03/16	LPU	H	10.4 13	11.4 19	3.18 3	8.10 17	6.08 14	20.9 19	8.24 13	6.94 8	4.00 2
11/14/16	PKLS	H	11.0 14	7.80 18	4.56 15	10.2 27	5.62 13	9.75 4	8.31 14	9.19 14	8.39 13

Results on the Middlebury stereo dataset ~2017



Date	Name	Res									
01/24/17	3DMST										
03/10/17	MC-CNN+TD										
05/12/16	PMSC	H	6.71 3	3.46 1	2.68 1	6.19 8	2.54 1	6.92 1	4.54 2	3.96 2	4.04 3
10/19/16	LW-CNN	H	7.04 4	4.65 5	3.95 5	5.30 4	2.63 2	11.2 12	5.41 3	4.32 4	4.22 4
04/12/16	MeshStereoExt	H	7.08 5	4.41 4	3.98 7	5.40 5	3.17 6	10.0 5	6.23 7	4.62 5	4.77 6
05/28/16	APAP-Stereo	H	7.26 6	5.43 6	4.91 18	5.11 3	5.17 12	21.6 20	6.99 9	4.31 3	4.23 5
01/19/16	NTDE	H	7.44 7	5.72 11	4.36 10	5.92 6	2.83 4	10.4 6	5.71 4	5.30 7	5.54 8
08/28/15	MC-CNN-acrt	H	8.08 8	5.59 10	4.55 14	5.96 7	2.83 4	11.4 13	5.81 5	8.32 11	8.89 15
11/03/15	MC-CNN+RBS	H	8.42 9	6.05 13	5.16 22	6.24 9	3.27 7	11.1 11	6.36 8	8.87 13	9.83 20
09/13/16	SNP-RSM	H	8.75 10	5.46 8	4.85 16	6.50 11	3.37 8	10.4 7	7.31 11	8.73 12	9.37 18
01/21/16	MCCNN_Layout	H	8.94 11	5.53 9	5.63 25	5.06 2	3.59 10	12.6 15	7.23 10	7.53 10	8.86 14
01/26/16	MC-CNN-fst	H	9.47 12	7.35 17	5.07 21	7.18 14	4.71 11	16.8 18	8.47 15	7.37 9	6.97 9
07/03/16	LPU	H	10.4 13	11.4 19	3.18 3	8.10 17	6.08 14	20.9 19	8.24 13	6.94 8	4.00 2
11/14/16	PKLS	H	11.0 14	7.80 18	4.56 15	10.2 27	5.62 13	9.75 4	8.31 14	9.19 14	8.39 13

Middlebury Stereo Evaluation - Version 3

Mouseover the table cells to see the produced disparity map. Clicking a cell will blink the ground truth for comparison. To change the table type, click the links below. For more information, please see the description of new features.

Submit and evaluate your own results. See snapshots of previous results. See the evaluation v.2 (no longer active).

Set: test dense test sparse training dense training sparse
Metric: bad 0.5 bad 1.0 bad 2.0 bad 4.0 avgerr rms A50 A90 A95 A99 time time/MP time/GD
Mask: nonocc all

plot selected show invalid **Reset sort** **Reference list**

Date	bad 2.0 (%) Name	Res	Weight Avg	Austr MP: 5.6 nd: 290 im0 im1 GT nonocc	AustrP MP: 5.6 nd: 290 im0 im1 GT nonocc	Bicyc2 MP: 5.6 nd: 250 im0 im1 GT nonocc	Class MP: 5.7 nd: 610 im0 im1 GT nonocc	ClassE MP: 5.7 nd: 610 im0 im1 GT nonocc	Compu MP: 1.5 nd: 256 im0 im1 GT nonocc	Crusa MP: 5.5 nd: 800 im0 im1 GT nonocc	CrusaP MP: 5.5 nd: 800 im0 im1 GT nonocc	Djemb MP: 5.7 nd: 320 im0 im1 GT nonocc	DjembL MP: 5.7 nd: 320 im0 im1 GT nonocc	Hoops MP: 5.7 nd: 410 im0 im1 GT nonocc	Livgrm MP: 5.9 nd: 320 im0 im1 GT nonocc	Nkuba MP: 5.5 nd: 570 im0 im1 GT nonocc	Plant MP: 5. nd: 32 im0 im GT nonoc
06/13/22	EAI-Stereo	F	**3.68** 1	4.02 14	3.32 13	**2.48** 1	**1.42** 1	4.19 2	2.37 3	**2.18** 1	**2.01** 1	**1.16** 1	10.2 12	8.84 6	**4.00** 1	7.15 2	**3.14** 1
11/10/21	CREStereo	F	3.71 2	4.73 22	3.94 23	5.07 24	1.96 5	**3.02** 1	**1.42** 1	2.28 2	2.05 2	1.51 3	6.86 2	**6.35** 1	4.25 2	**6.01** 1	4.60 4
10/01/22	CREStereo++_RVC	F	4.68 3	5.09 25	4.04 26	5.24 27	4.21 35	5.05 4	2.11 2	3.52 14	3.58 13	1.67 4	8.01 5	6.61 2	4.68 3	9.53 3	4.61 5
07/26/21	RAFT-Stereo	F	4.74 4	4.19 18	3.44 16	3.11 5	1.51 2	7.30 19	2.79 5	2.67 3	2.59 3	1.39 2	7.46 3	10.2 16	5.86 4	13.0 18	3.59 2
05/26/18	NOSS_ROB	H	5.01 5	3.57 4	2.84 3	3.99 12	1.93 4	5.15 6	3.34 7	3.32 8	3.15 7	2.32 15	8.55 6	7.45 4	7.06 11	12.5 13	5.20
07/23/21	HBP_ISP	H	5.20 6	3.70 9	3.05 10	3.57 7	2.34 11	7.80 22	3.79 14	3.34 9	3.09 6	1.87 6	9.85 10	10.1 15	7.82 16	11.2 7	5.26
06/22/17	LocalExp	H	5.43 7	3.65 6	2.87 5	2.98 3	1.99 7	5.59 9	3.37 8	3.48 12	3.35 10	2.05 8	10.3 16	9.75 11	8.57 19	14.4 39	5.40
03/09/19	3DMST-CM	H	5.47 8	4.10 17	3.37 15	2.99 4	2.95 23	7.63 21	4.55 19	3.26 6	3.95 19	2.16 10	10.2 13	8.28 5	6.37 6	13.2 20	5.86
06/23/21	ERW-LocalExp	H	5.53 9	3.64 5	2.84 3	2.66 2	1.97 6	5.68 10	4.87 20	3.27 7	3.25 9	2.36 18	10.5 19	11.5 23	7.46 14	14.7 42	5.55
03/05/21	LocalExp-RC	H	5.54 10	3.78 13	3.02 9	3.85 8	2.08 8	5.95 11	3.48 12	3.61 15	3.65 14	2.52 23	10.3 17	6.85 3	7.25 12	16.1 58	5.12
08/30/20	LE_PC	H	5.58 11	3.52 3	2.99 8	4.24 15	1.92 3	5.39 8	3.42 11	3.16 5	3.72 16	2.30 14	7.83 4	9.90 13	7.79 15	17.4 69	4.74 8
07/16/20	HLocalExp-CM	H	5.68 12	3.68 8	2.95 7	3.92 9	2.45 12	8.12 24	3.41 10	3.74 17	3.53 12	2.17 11	10.2 13	10.0 14	8.75 21	14.1 36	5.12
12/19/19	CRLE	H	5.75 13	3.66 7	3.11 11	5.92 41	2.14 9	6.01 12	3.39 9	3.49 13	3.68 15	2.34 16	10.2 13	9.63 10	8.04 17	14.9 45	5.45
01/24/17	3DMST	H	5.92 14	3.71 10	2.78 2	4.75 19	2.72 16	7.36 20	4.28 17	3.44 10	3.76 17	2.35 17	12.6 30	11.5 22	8.56 18	14.0 35	5.35
10/16/22	LMCR-Stereo	F	6.27 15	6.20 41	4.59 45	3.92 9	2.66 15	4.52 3	4.88 21	3.65 16	3.41 11	2.08 9	16.8 51	11.2 20	8.58 20	13.2 20	6.89
03/10/17	MC-CNN+TDSR	F	6.35 16	5.45 29	4.45 37	6.80 58	3.46 30	10.7 38	6.05 37	5.01 26	5.19 30	2.62 25	10.8 21	9.62 9	6.59 8	11.4 8	6.01

Practical Stereo Matching via Cascaded Recurrent Network with Adaptive Correlation

CVPR 2022

Jiankun Li[1] Peisen Wang[1*] Pengfei Xiong[2*] Tao Cai[1] Ziwei Yan[1] Lei Yang[1]
Jiangyu Liu[1] Haoqiang Fan[1] Shuaicheng Liu[3,1†]
[1]Megvii Research [2]Tencent
[3]University of Electronic Science and Technology of China
https://github.com/megvii-research/CREStereo

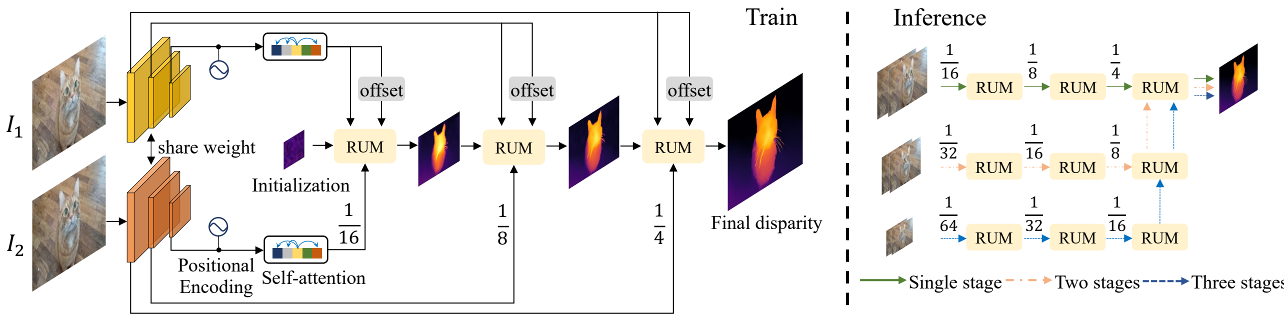


Figure 2. An overview of our proposed network. Left: A pair of stereo images I_1 and I_2 are fed into two shared-weight feature extraction networks to produce a 3-level feature pyramid, which is used to compute different scales of correlations in the 3 stages of cascaded recurrent networks. The feature pyramid of I_1 also provides context information for latter update blocks and offsets computation. In each stage of the cascades, the features and the predicted disparities are refined iteratively using the Recurrent Update Module (RUM, Sec. 3.2), and the final output disparity of the former stage is fed to the next as an initialization. For each iteration in RUM, we apply Adaptive Group Correlation Layer (AGCL, Sec. 3.1) to compute the correlation. Right: Our proposed stacked cascaded architecture in inference phase, which takes an image pyramid as input, taking advantage of multi-level context, as detailed in Sec. 3.3 .

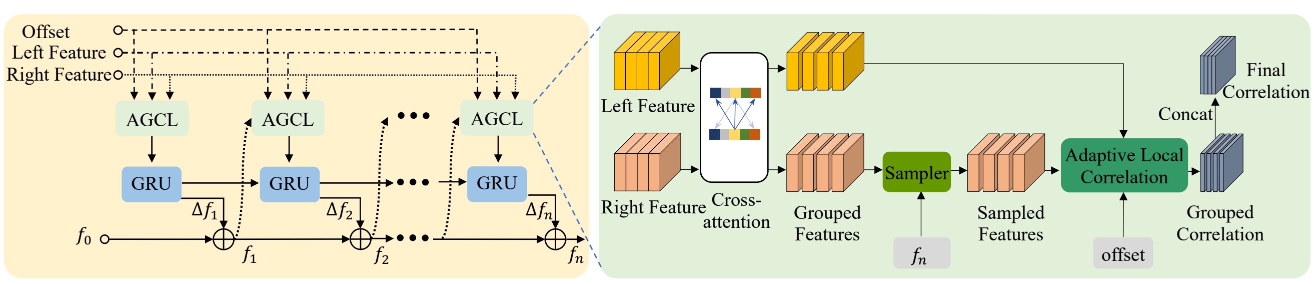
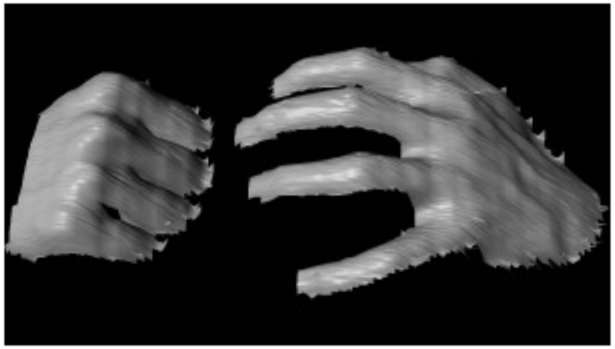


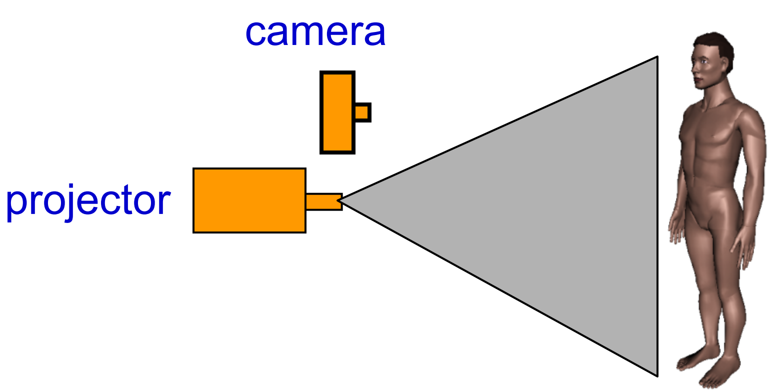
Figure 3. The architecture of proposed modules. Left: Recurrent Update Module (RUM). Right: Adaptive Group Correlation Layer (AGCL). Details are described in Sec. 3.2 and Sec. 3.1, respectively.

Other approaches to obtaining 3D structure

Active stereo with structured light

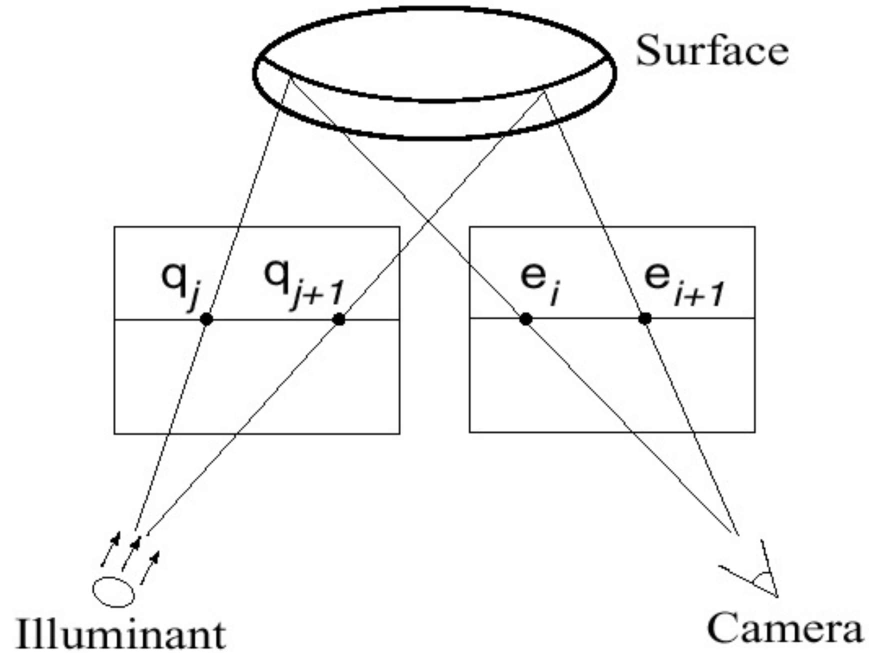


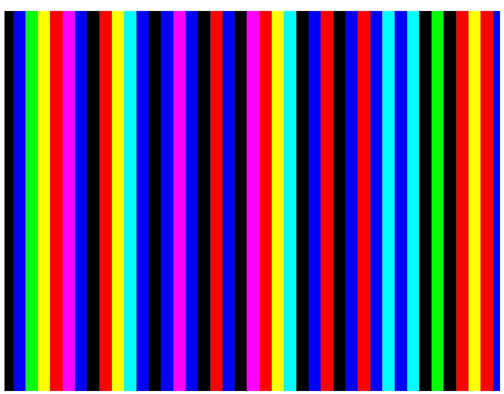
- Project "structured" light patterns onto the object
  - simplifies the correspondence problem
  - Allows us to use only one camera

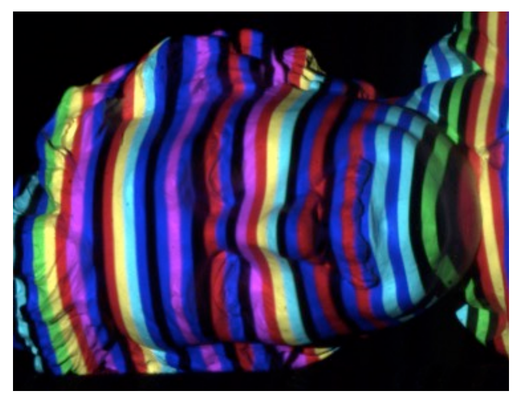


L. Zhang, B. Curless, and S. M. Seitz. Rapid Shape Acquisition Using Color Structured Light and Multi-pass Dynamic Programming. *3DPVT* 2002

Active stereo with structured light

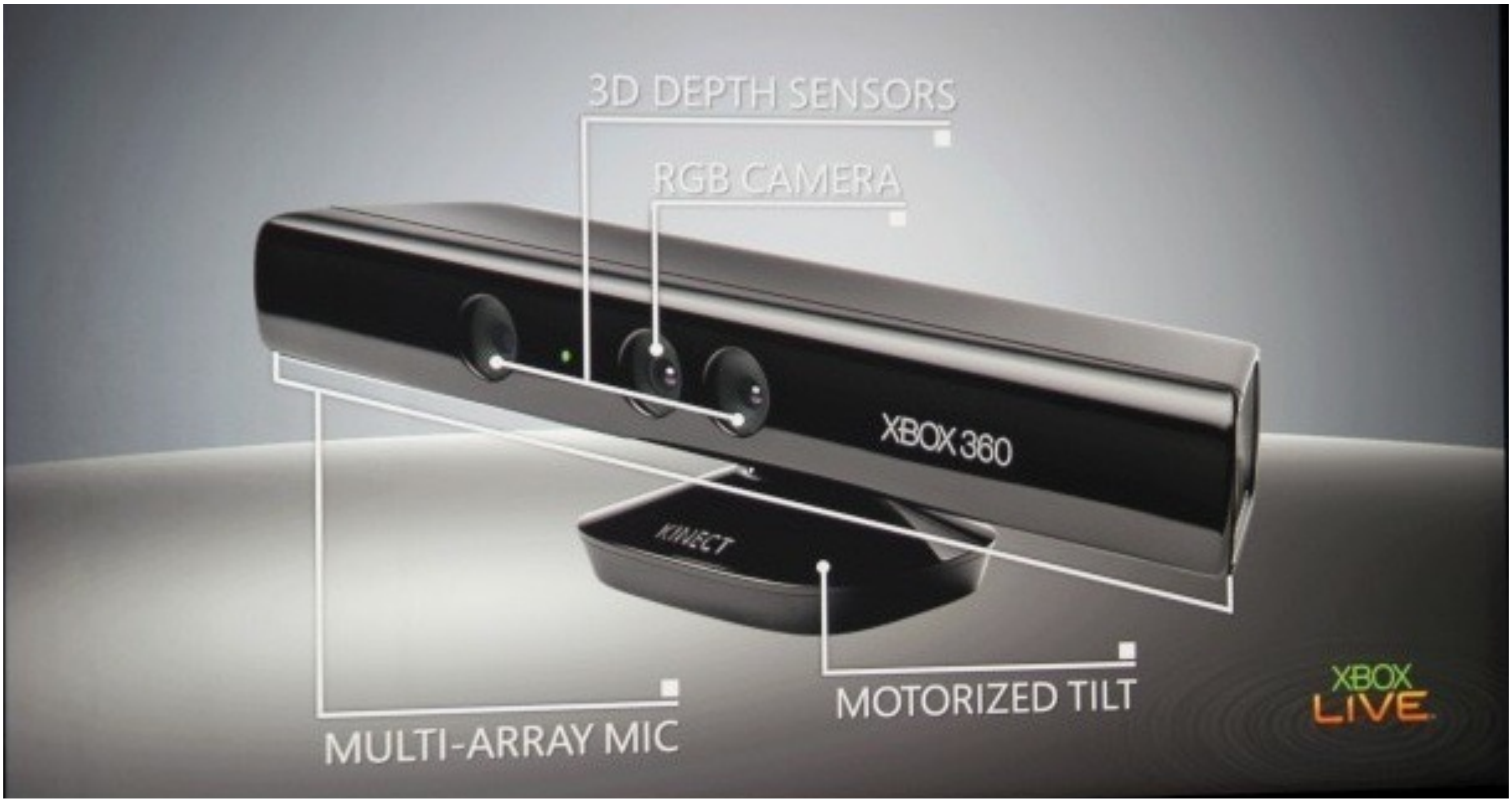




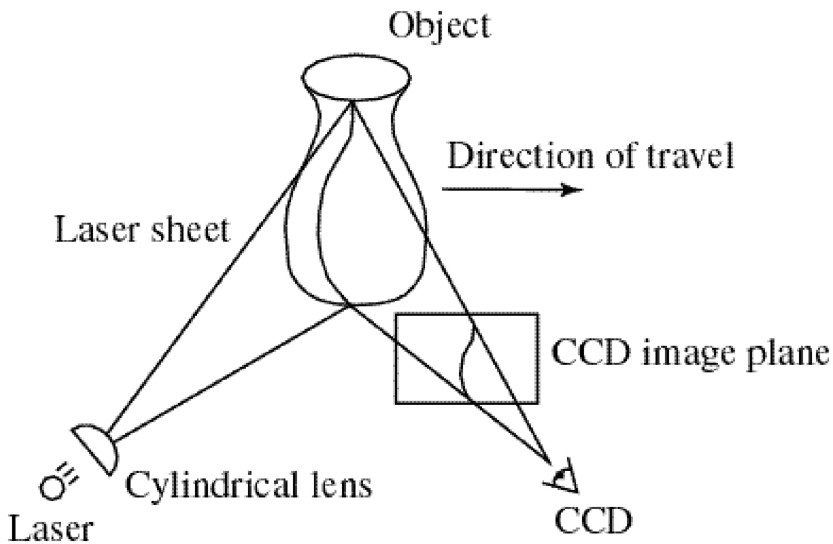


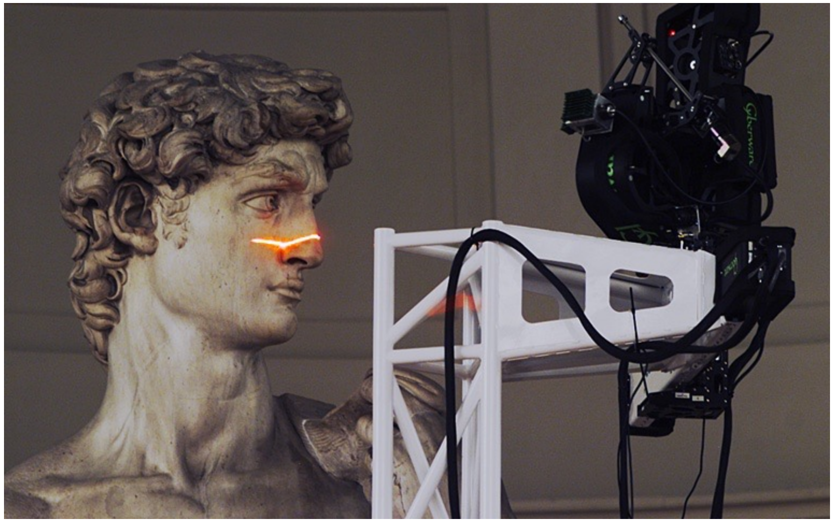
L. Zhang, B. Curless, and S. M. Seitz. Rapid Shape Acquisition Using Color Structured Light and Multi-pass Dynamic Programming. *3DPVT* 2002

Microsoft Kinect



Laser scanning





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

- Optical triangulation
  - Project a single stripe of laser light
  - Scan it across the surface of the object
  - This is a very precise version of structured light scanning

Source: S. Seitz

Laser scanned models

The Digital Michelangelo Project, Levoy et al.

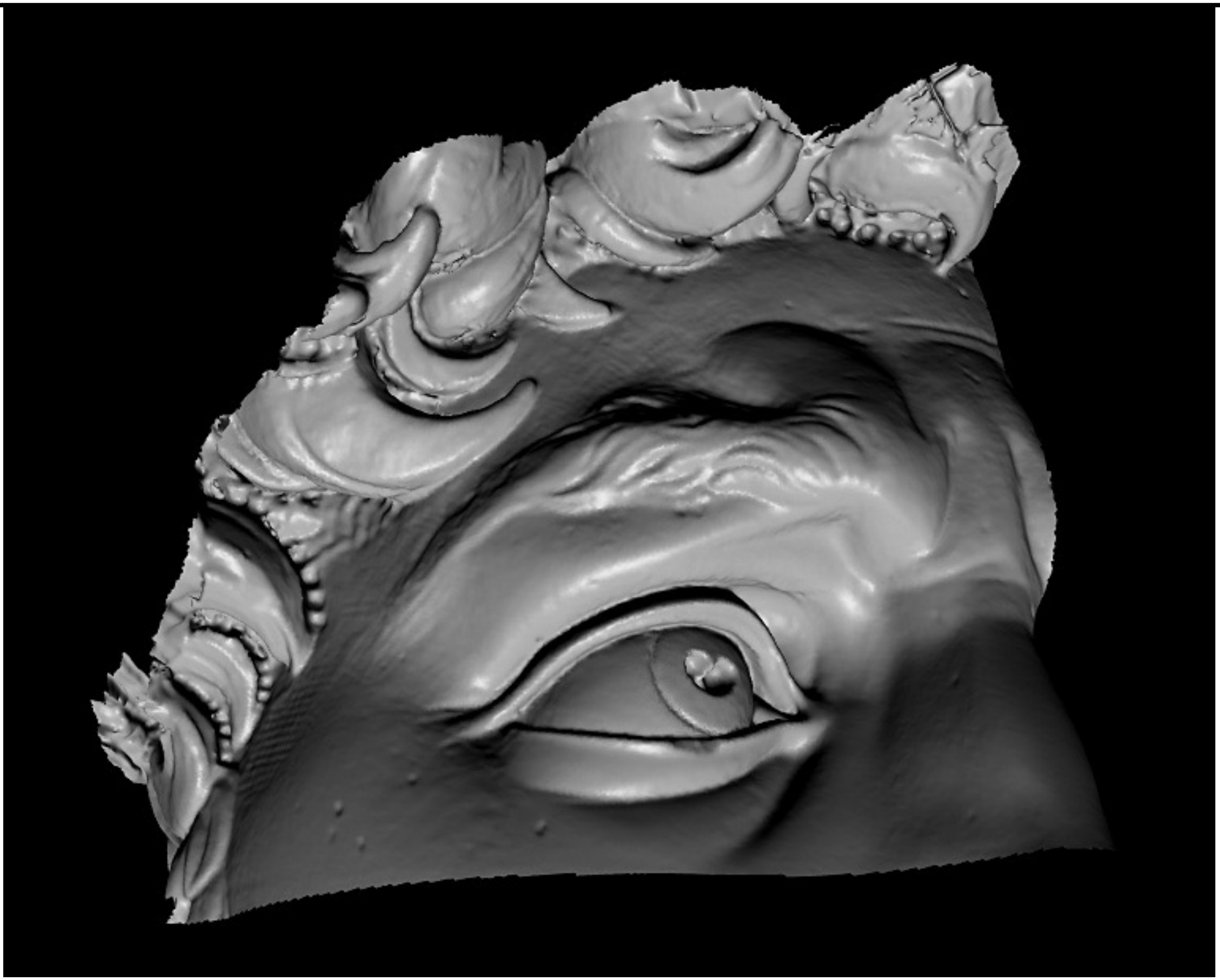
Source: S. Seitz

Laser scanned models

The Digital Michelangelo Project, Levoy et al.

Source: S. Seitz

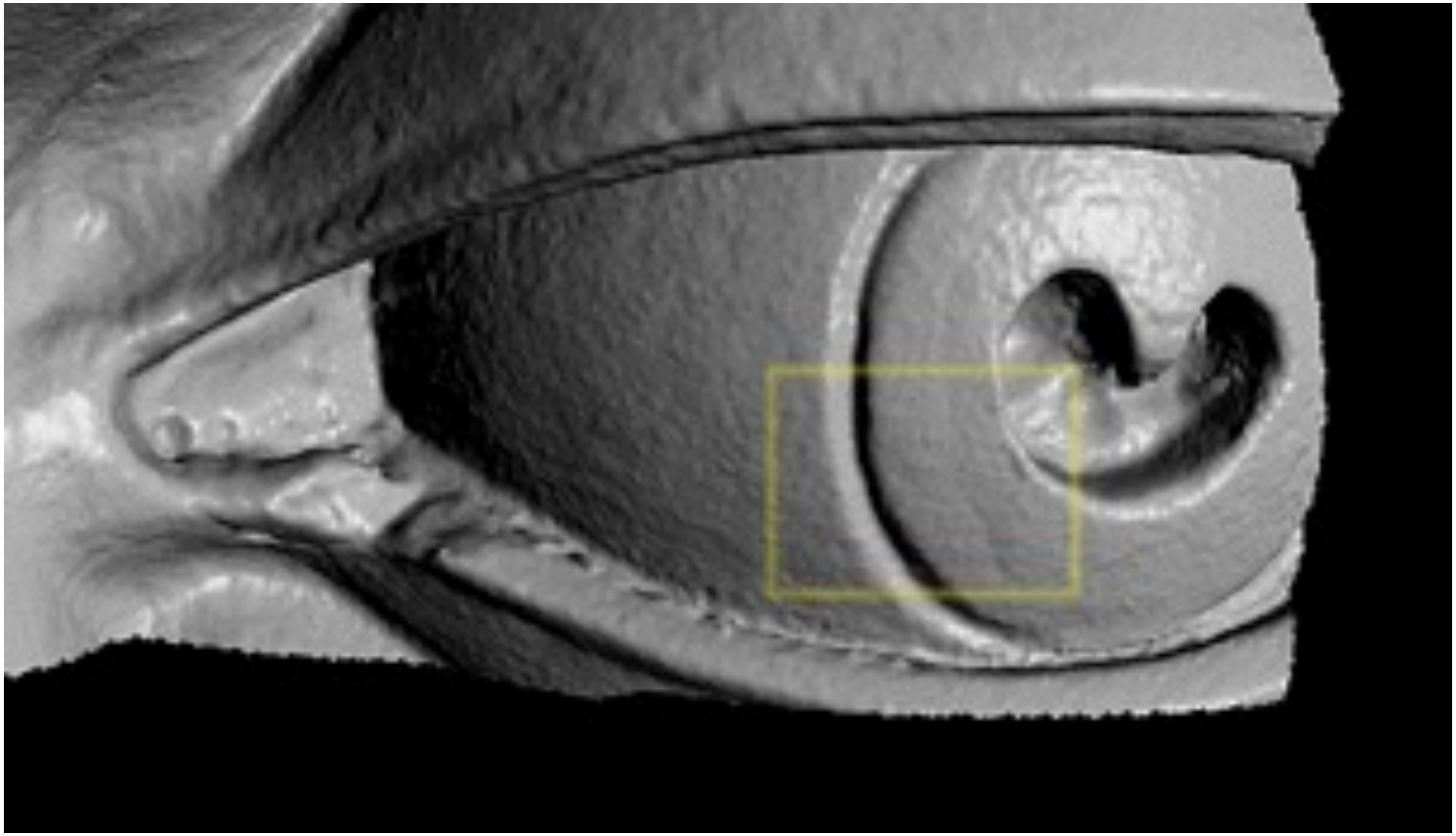
Laser scanned models



The Digital Michelangelo Project, Levoy et al.

Source: S. Seitz

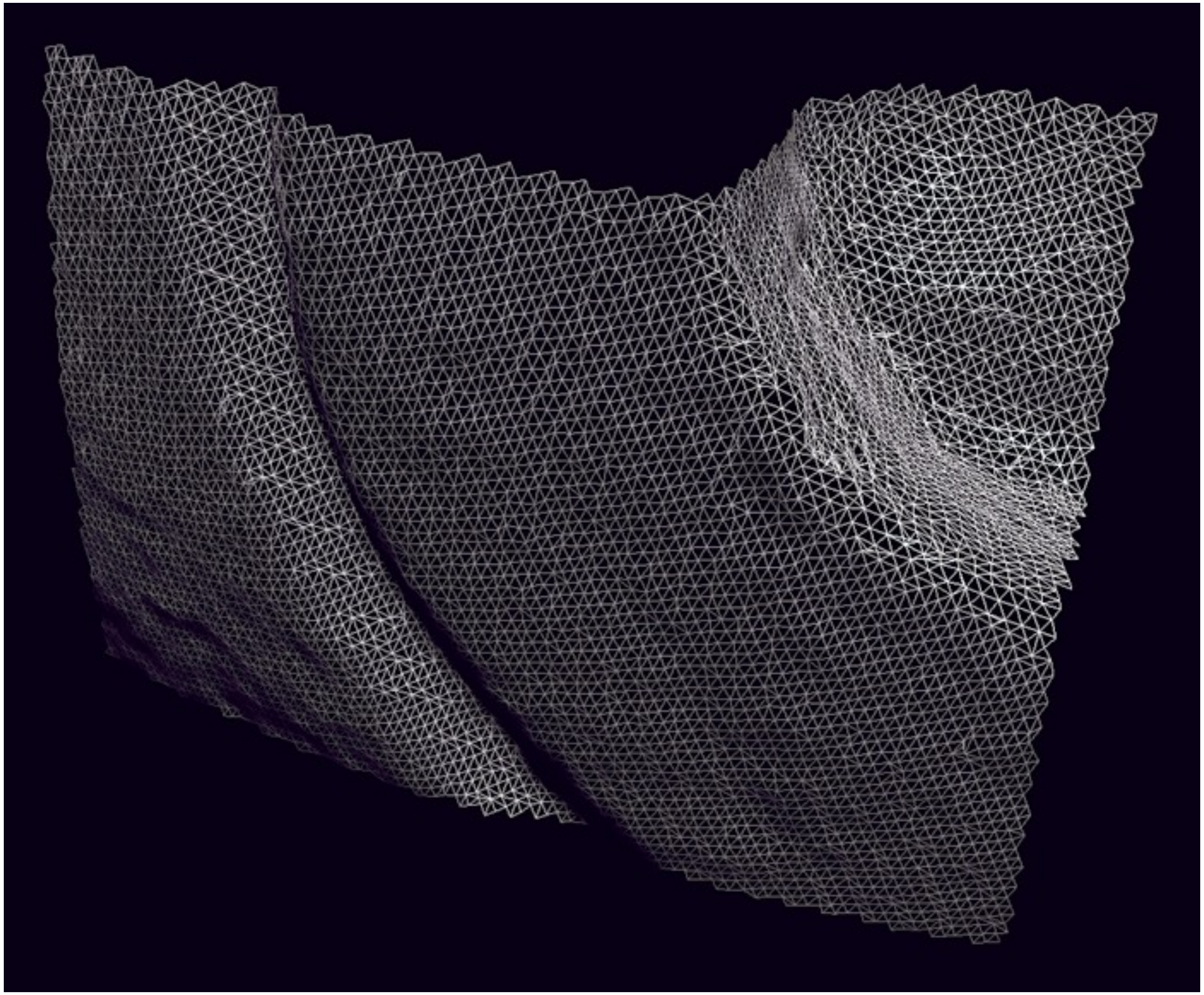
Laser scanned models



The Digital Michelangelo Project, Levoy et al.

Source: S. Seitz

Laser scanned models

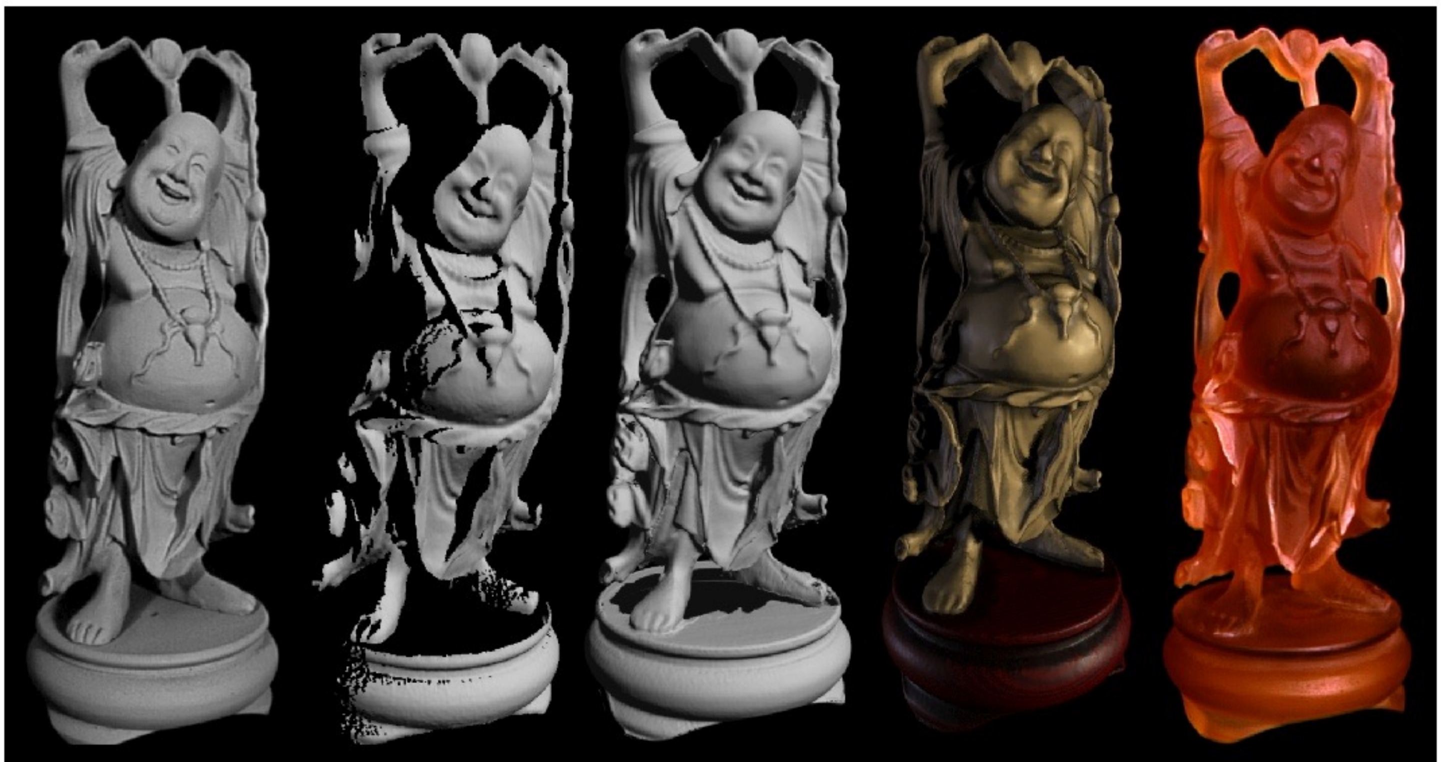


The Digital Michelangelo Project, Levoy et al.

Source: S. Seitz

Aligning range images

- A single range scan is not sufficient to describe a complex surface
- Need techniques to register multiple range images



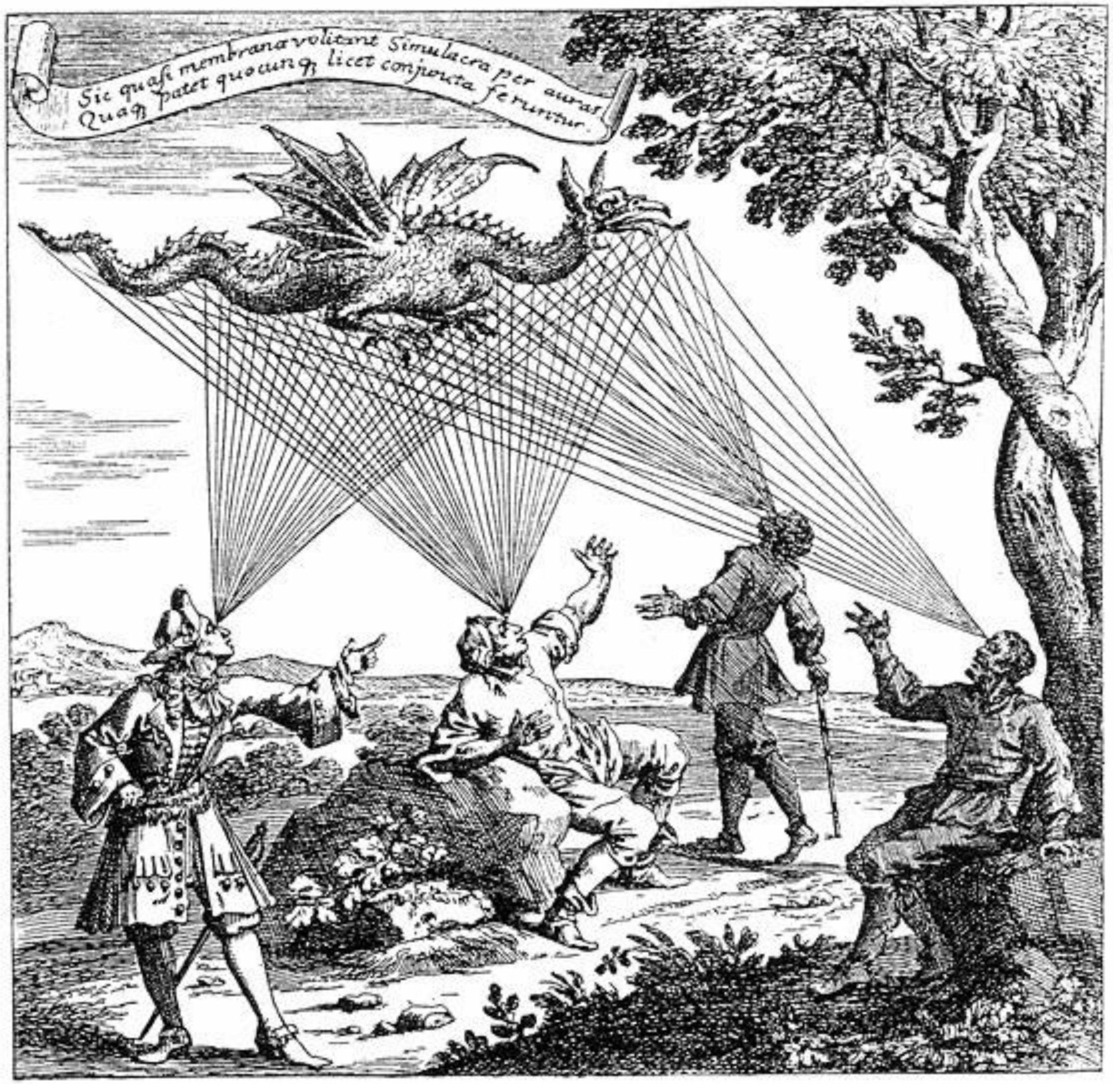
B. Curless and M. Levoy, A Volumetric Method for Building Complex Models from Range Images, SIGGRAPH 1996

Aligning range images

- A single range scan is not sufficient to describe a complex surface
- Need techniques to register multiple range images

    - … which brings us to *multi-view stereo*

Structure from motion



Драконъ, видимый подъ различными углами зрѣнія
По гравюрѣ на мѣди изъ „Oculus artificialis telediopricus" Цана. 1702 года.

Multiple-view geometry questions

- **Scene geometry (structure):** Given 2D point matches in two or more images, where are the corresponding points in 3D?

- **Correspondence (stereo matching):** Given a point in just one image, how does it constrain the position of the corresponding point in another image?

- **Camera geometry (motion):** Given a set of corresponding points in two or more images, what are the camera matrices for these views?

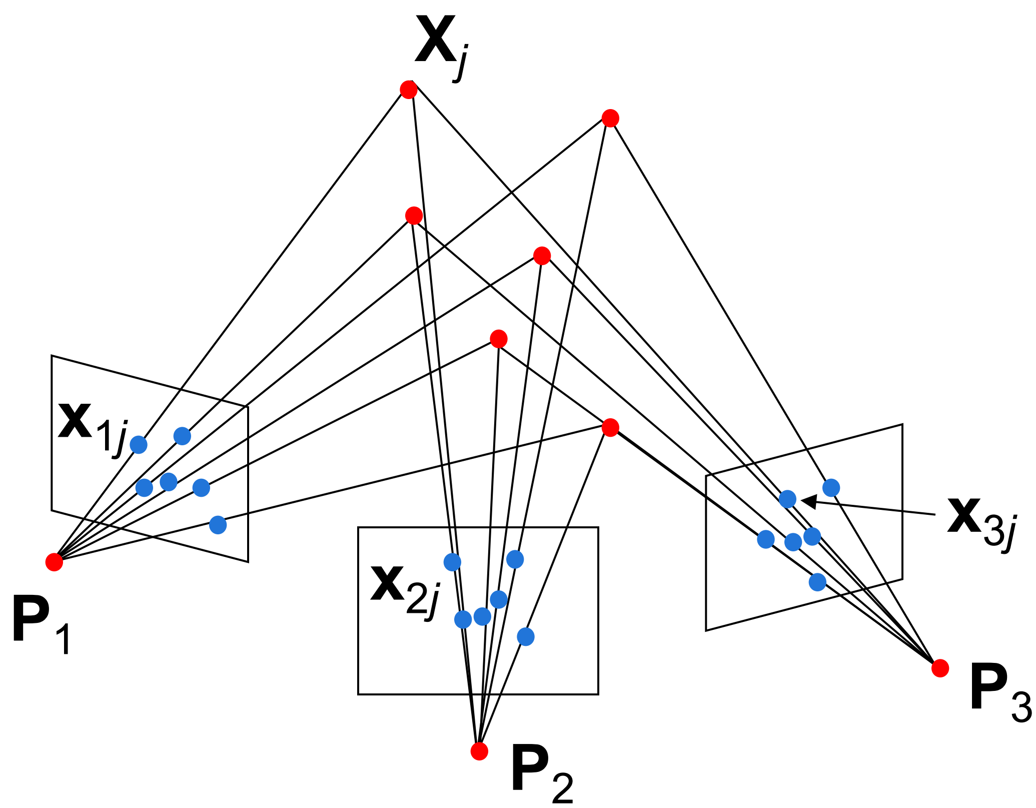
Slide: S. Lazebnik

Structure from motion

- Given: *m* images of *n* fixed 3D points

$$\mathbf{x}_{ij} = \mathbf{P}_i \mathbf{X}_j, \qquad i = 1, \ldots, m, \quad j = 1, \ldots, n$$

- Problem: estimate *m* projection matrices $\mathbf{P}_i$ and *n* 3D points $\mathbf{X}_j$ from the *mn* correspondences $\mathbf{x}_{ij}$



Slide: S. Lazebnik

Structure from motion ambiguity

- If we scale the entire scene by some factor *k* and, at the same time, scale the camera matrices by the factor of 1/*k*, the projections of the scene points in the image remain exactly the same:

$$\mathbf{x} = \mathbf{P}\mathbf{X} = \left(\frac{1}{k}\mathbf{P}\right)(k\,\mathbf{X})$$

It is impossible to recover the absolute scale of the scene!

Slide: S. Lazebnik

Structure from motion ambiguity

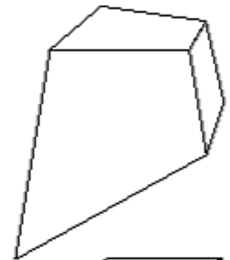
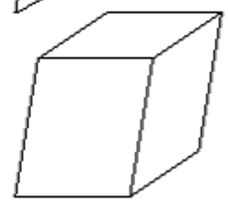
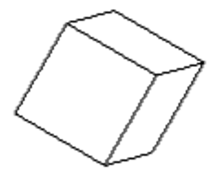
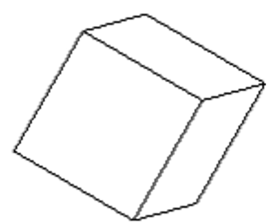
- If we scale the entire scene by some factor *k* and, at the same time, scale the camera matrices by the factor of 1/*k*, the projections of the scene points in the image remain exactly the same

- More generally: if we transform the scene using a transformation **Q** and apply the inverse transformation to the camera matrices, then the images do not change

$$\mathbf{x} = \mathbf{P}\mathbf{X} = \left(\mathbf{P}\mathbf{Q}^{-1}\right)\left(\mathbf{Q}\mathbf{X}\right)$$

Slide: S. Lazebnik

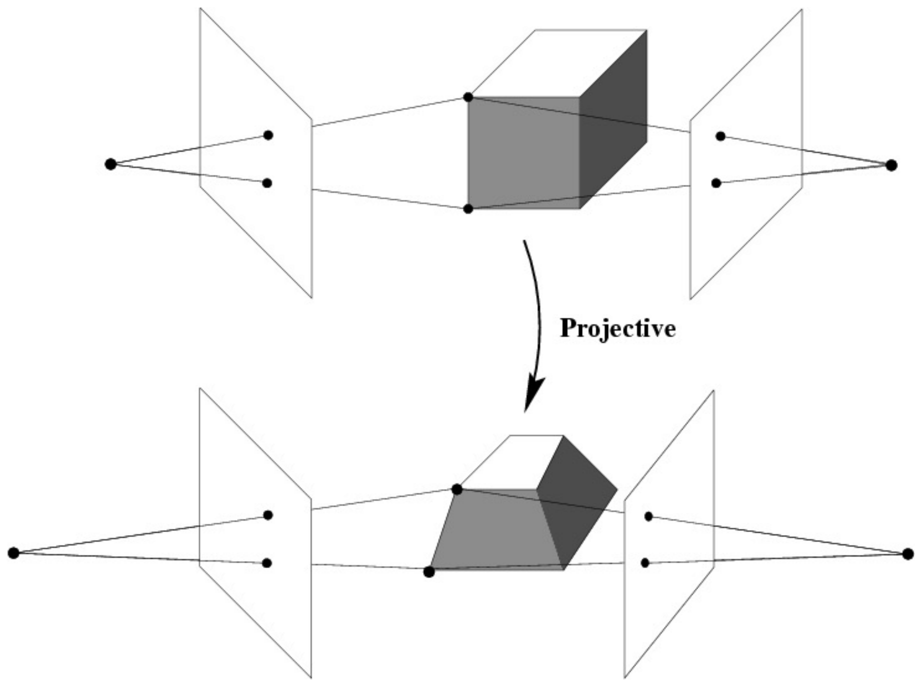
Types of ambiguity

Projective 15dof	$\begin{bmatrix} A & t \\ v^T & v \end{bmatrix}$		Preserves intersection and tangency
Affine 12dof	$\begin{bmatrix} A & t \\ 0^T & 1 \end{bmatrix}$		Preserves parallellism, volume ratios
Similarity 7dof	$\begin{bmatrix} s\,R & t \\ 0^T & 1 \end{bmatrix}$		Preserves angles, ratios of length
Euclidean 6dof	$\begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix}$		Preserves angles, lengths

- With no constraints on the camera calibration matrix or on the scene, we get a *projective* reconstruction
- Need additional information to *upgrade* the reconstruction to affine, similarity, or Euclidean

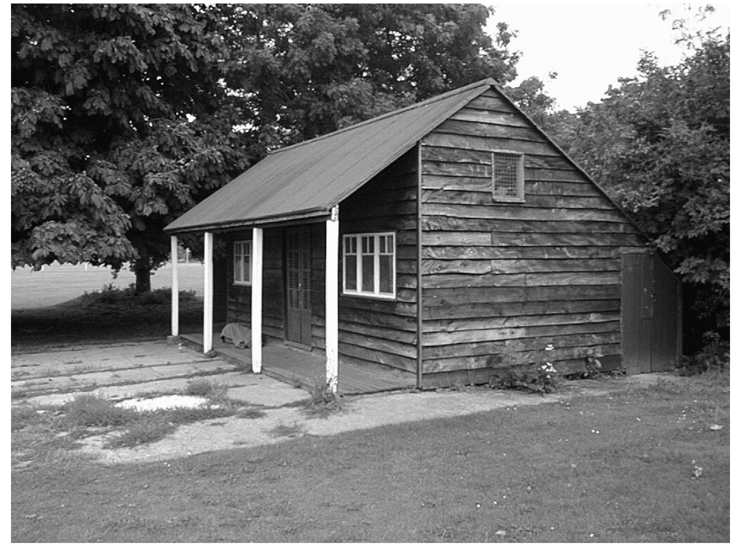
Slide: S. Lazebnik

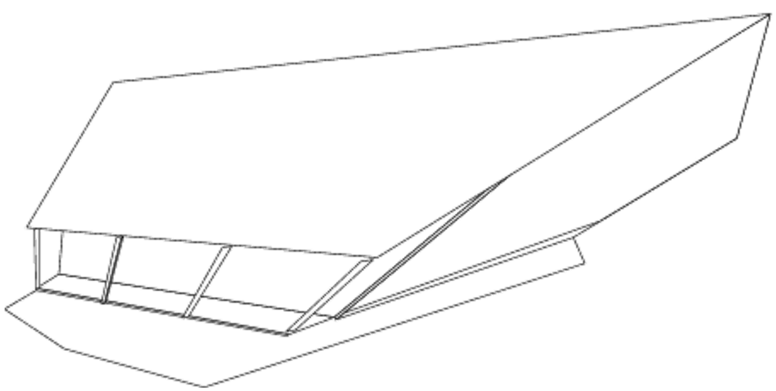
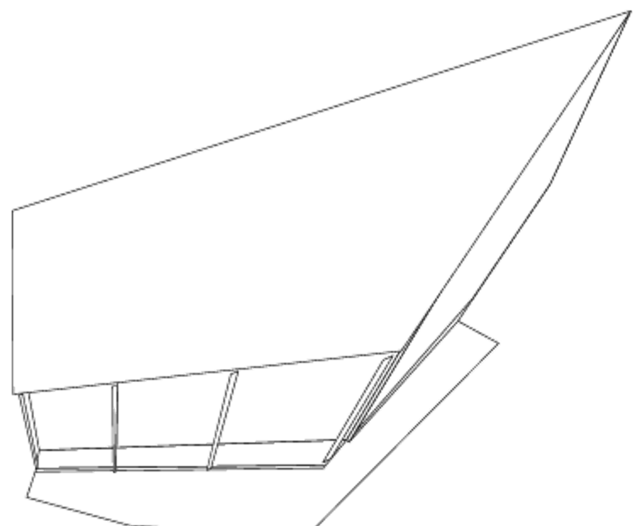
Projective ambiguity



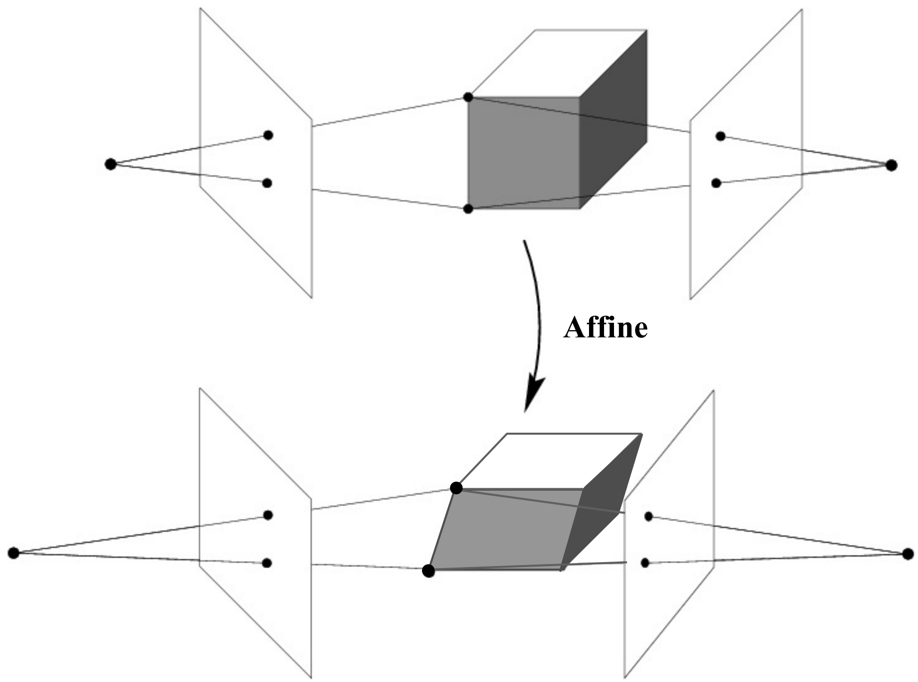
$$\mathbf{x} = \mathbf{P}\mathbf{X} = \left(\mathbf{P}\mathbf{Q}_{\mathbf{P}}^{-1}\right)\left(\mathbf{Q}_{\mathbf{P}}\mathbf{X}\right)$$

Projective ambiguity

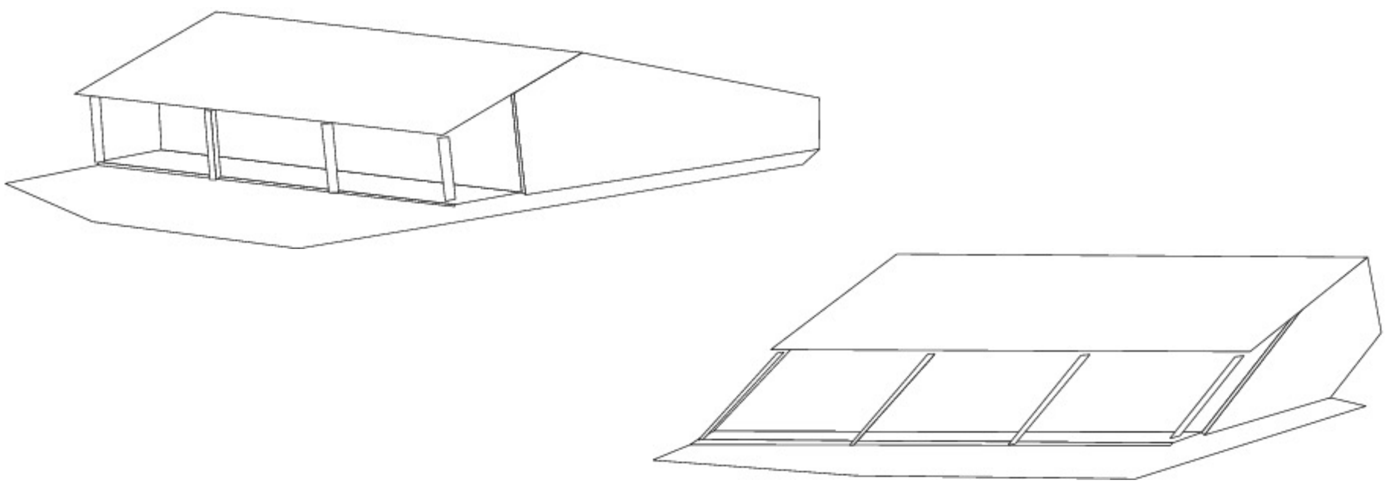
 

Affine ambiguity

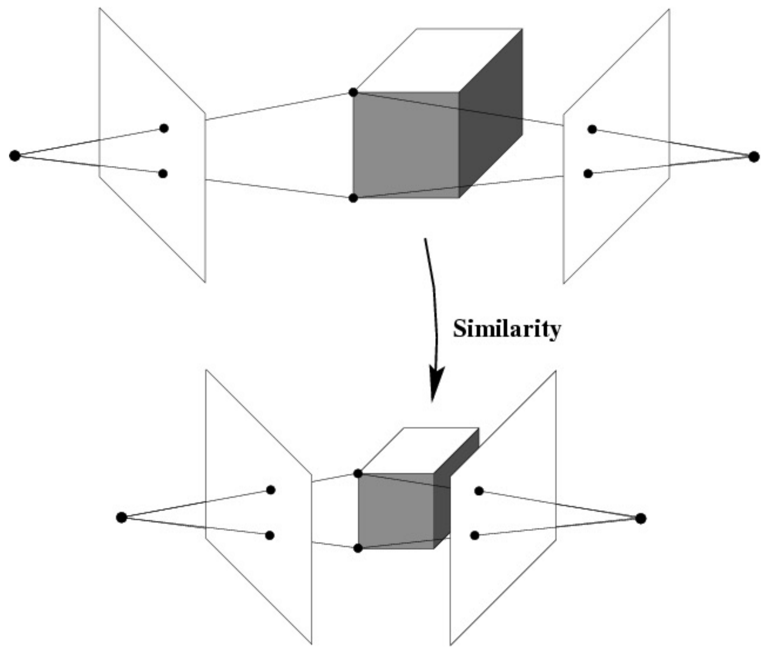


$$\mathbf{x} = \mathbf{P}\mathbf{X} = \left(\mathbf{P}\mathbf{Q}_A^{-1}\right)\left(\mathbf{Q}_A\,\mathbf{X}\right)$$

Affine ambiguity

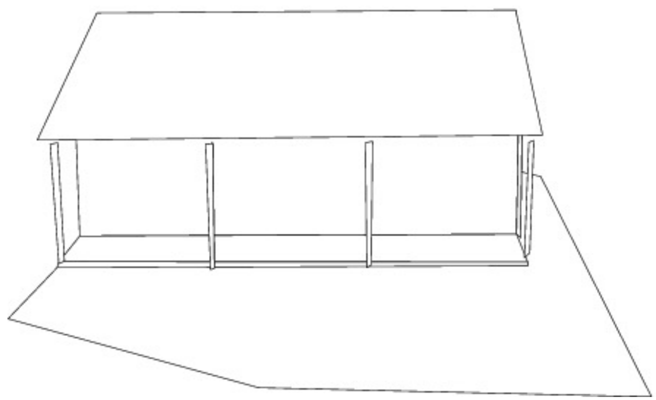


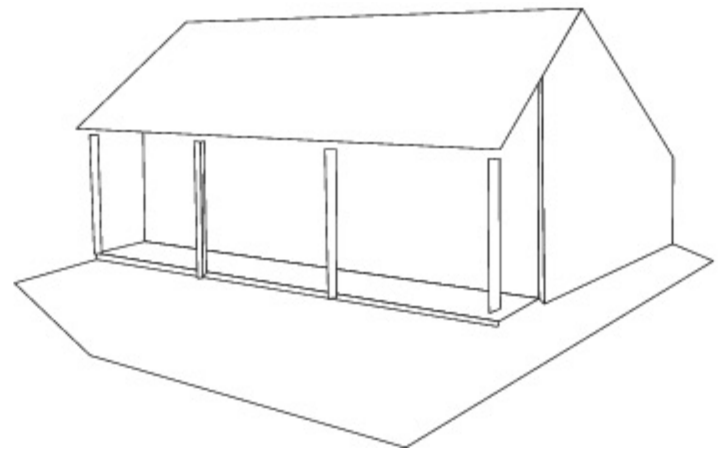
Similarity ambiguity

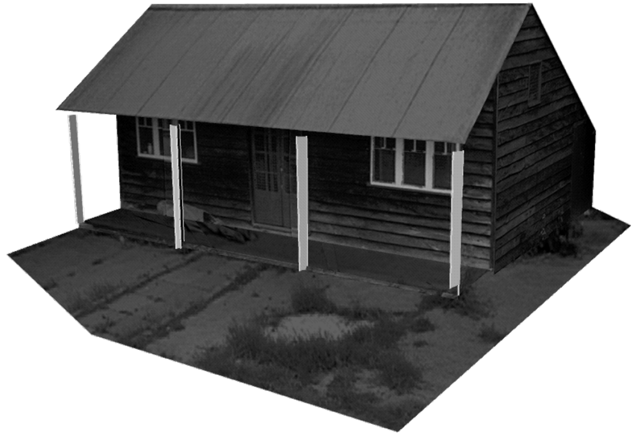


$$\mathbf{x} = \mathbf{P}\mathbf{X} = \left(\mathbf{P}\mathbf{Q}_S^{-1}\right)\left(\mathbf{Q}_S\mathbf{X}\right)$$

Similarity ambiguity

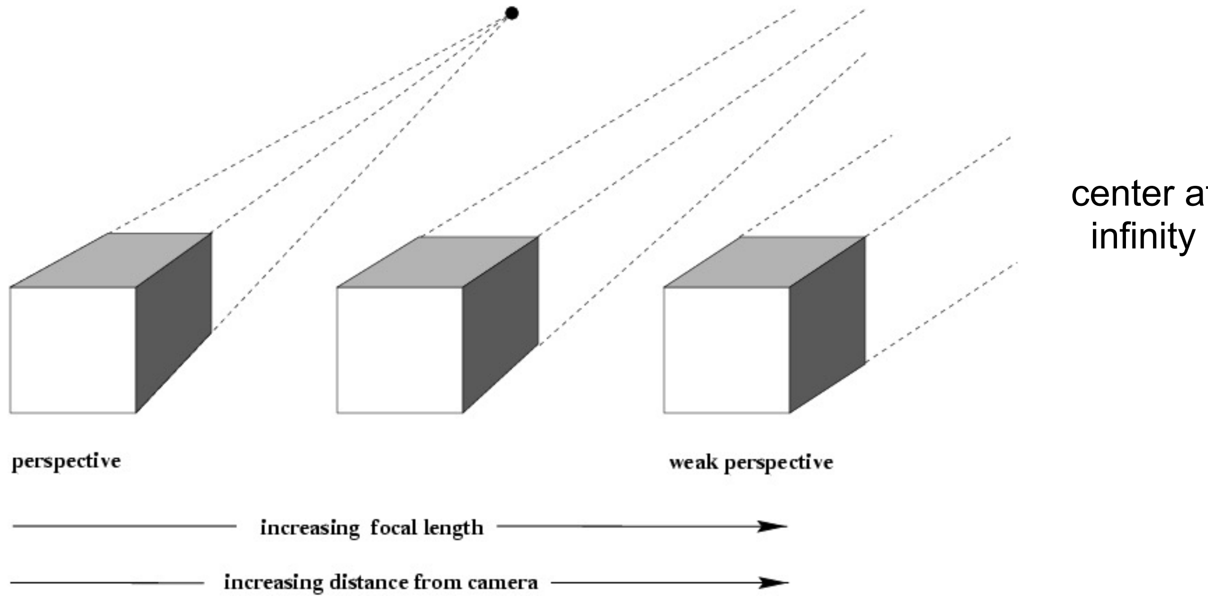






Structure from motion

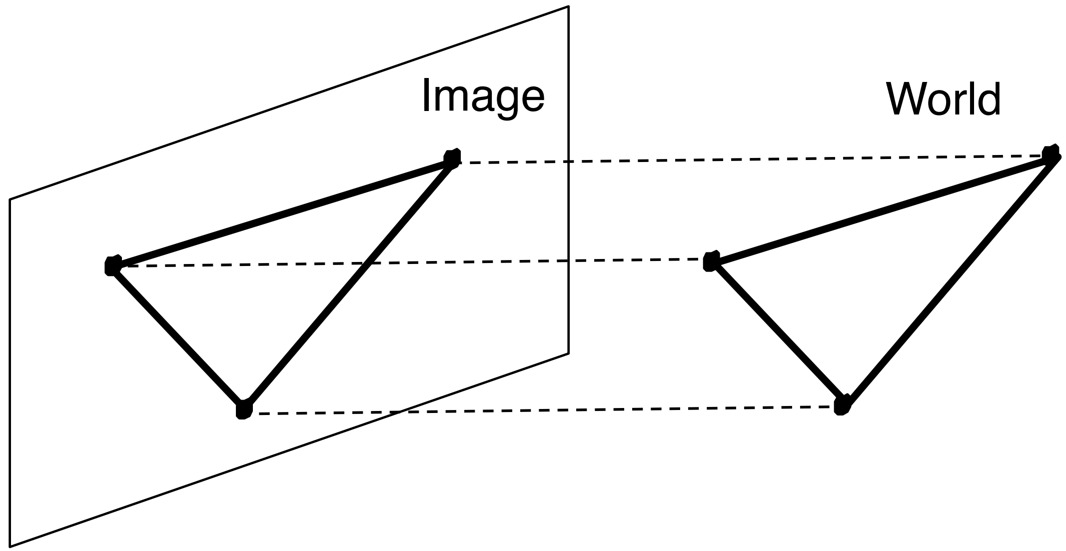
- Let's start with *affine cameras* (the math is easier)



Recall: Orthographic Projection

Special case of perspective projection

- Distance from center of projection to image plane is infinite



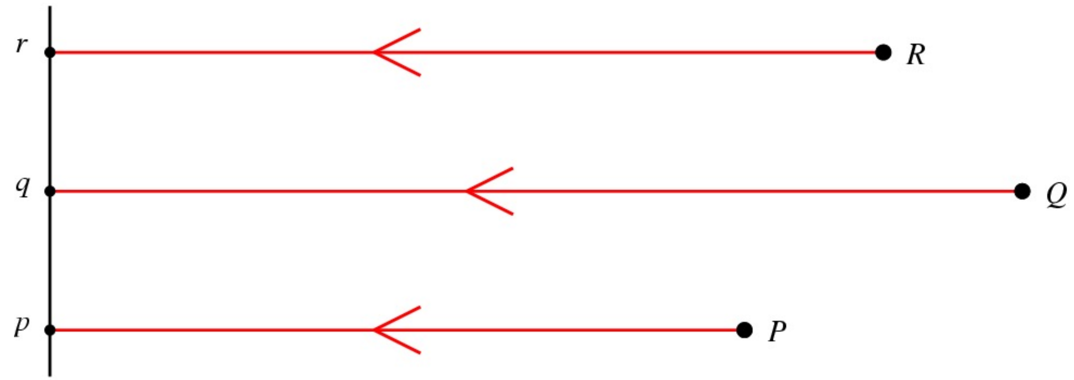
- Projection matrix:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Rightarrow (x, y)$$

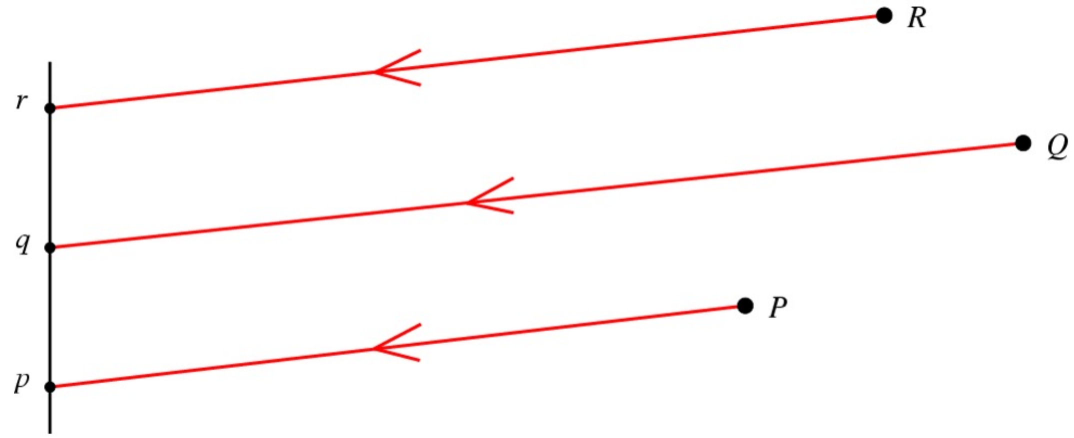
Slide by Steve Seitz

Affine cameras

Orthographic Projection



Parallel Projection

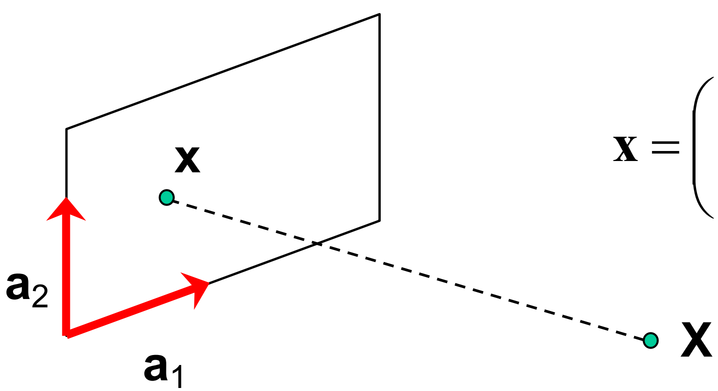


Affine cameras

- A general affine camera combines the effects of an affine transformation of the 3D space, orthographic projection, and an affine transformation of the image:

$$\mathbf{P} = [3\times 3\,\text{affine}]\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}[4\times 4\,\text{affine}] = \begin{bmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0} & \mathbf{1} \end{bmatrix}$$

- Affine projection is a linear mapping + translation in inhomogeneous coordinates



$$\mathbf{x} = \begin{pmatrix} x \\ y \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix}\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = \mathbf{AX} + \mathbf{b}$$

Projection of world origin

Affine structure from motion

- Given: *m* images of *n* fixed 3D points:

$$\mathbf{x}_{ij} = \mathbf{A}_i \mathbf{X}_j + \mathbf{b}_i, \quad i = 1, \dots, m, \; j = 1, \dots, n$$

- Problem: use the *mn* correspondences $\mathbf{x}_{ij}$ to estimate *m* projection matrices $\mathbf{A}_i$ and translation vectors $\mathbf{b}_i$, and *n* points $\mathbf{X}_j$
- The reconstruction is defined up to an arbitrary *affine* transformation **Q** (12 degrees of freedom):

$$\begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \to \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \mathbf{Q}^{-1}, \qquad \begin{pmatrix} \mathbf{X} \\ \mathbf{1} \end{pmatrix} \to \mathbf{Q} \begin{pmatrix} \mathbf{X} \\ \mathbf{1} \end{pmatrix}$$

- We have $2mn$ knowns and $8m + 3n$ unknowns (minus 12 dof for affine ambiguity)
- Thus, we must have $2mn >= 8m + 3n – 12$
- For two views, we need four point correspondences

Affine structure from motion

- Centering: subtract the centroid of the image points

$$\hat{\mathbf{x}}_{ij} = \mathbf{x}_{ij} - \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_{ik} = \mathbf{A}_i\mathbf{X}_j + \mathbf{b}_i - \frac{1}{n}\sum_{k=1}^{n}\left(\mathbf{A}_i\mathbf{X}_k + \mathbf{b}_i\right)$$

$$= \mathbf{A}_i\left(\mathbf{X}_j - \frac{1}{n}\sum_{k=1}^{n}\mathbf{X}_k\right) = \mathbf{A}_i\hat{\mathbf{X}}_j$$

- For simplicity, assume that the origin of the world coordinate system is at the centroid of the 3D points
- After centering, each normalized point $\mathbf{x}_{ij}$ is related to the 3D point $\mathbf{X}_i$ by

$$\hat{\mathbf{x}}_{ij} = \mathbf{A}_i\mathbf{X}_j$$

Affine structure from motion

- Let's create a $2m \times n$ data (measurement) matrix:

$$\mathbf{D} = \begin{bmatrix} \hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\ \hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\ & & \ddots & \\ \hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn} \end{bmatrix}$$

cameras ($2m$)

points (n)

C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *IJCV*, 9(2):137-154, November 1992.

Affine structure from motion

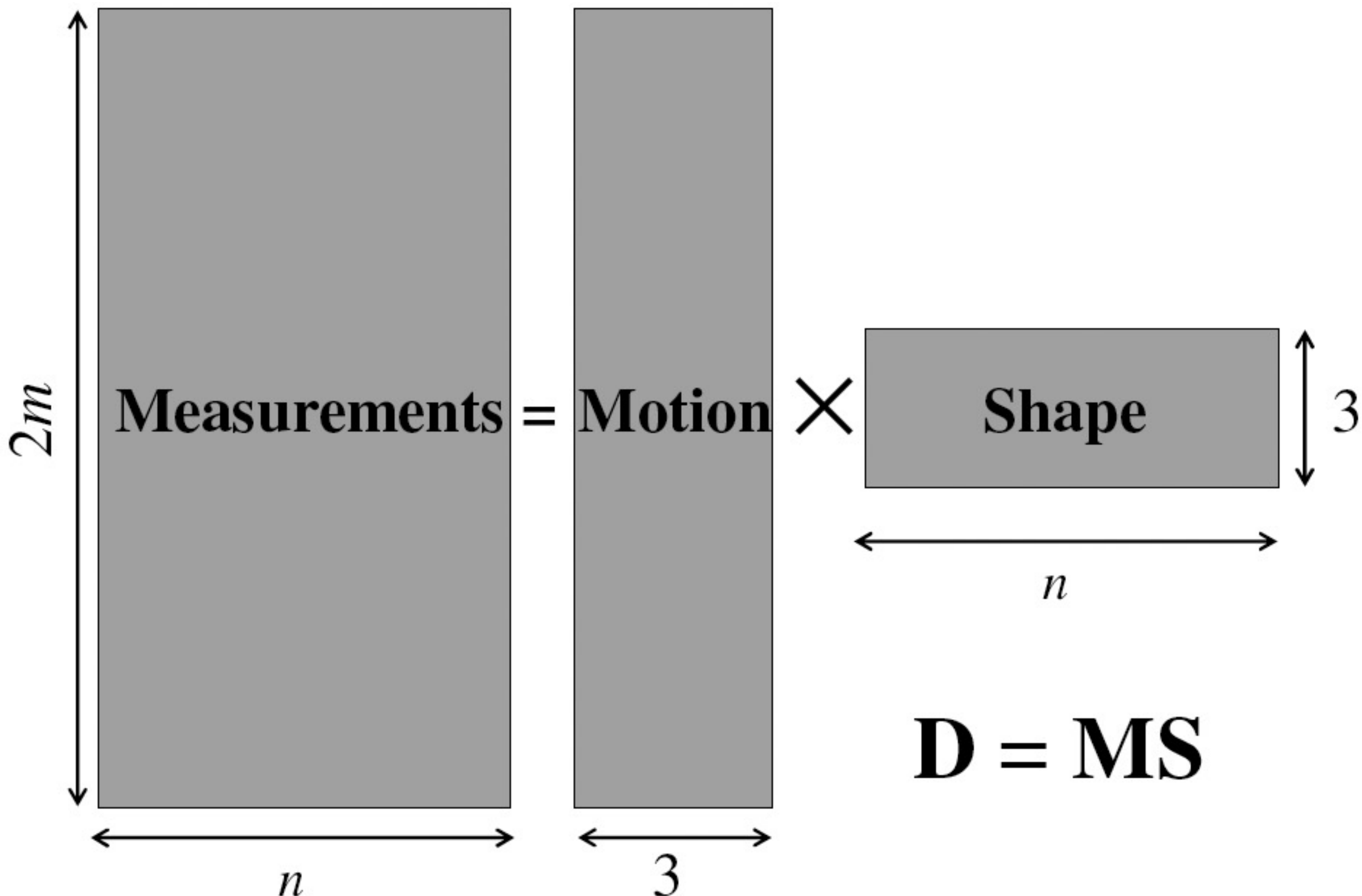
- Let's create a $2m \times n$ data (measurement) matrix:

$$\mathbf{D} = \begin{bmatrix} \hat{\mathbf{x}}_{11} & \hat{\mathbf{x}}_{12} & \cdots & \hat{\mathbf{x}}_{1n} \\ \hat{\mathbf{x}}_{21} & \hat{\mathbf{x}}_{22} & \cdots & \hat{\mathbf{x}}_{2n} \\ & & \ddots & \\ \hat{\mathbf{x}}_{m1} & \hat{\mathbf{x}}_{m2} & \cdots & \hat{\mathbf{x}}_{mn} \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \\ \vdots \\ \mathbf{A}_m \end{bmatrix}}_{\text{cameras } (2m \times 3)} \underbrace{\begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \cdots & \mathbf{X}_n \end{bmatrix}}_{\text{points } (3 \times n)}$$

The measurement matrix $\mathbf{D} = \mathbf{MS}$ must have rank 3!

C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *IJCV*, 9(2):137-154, November 1992.

Factorizing the measurement matrix

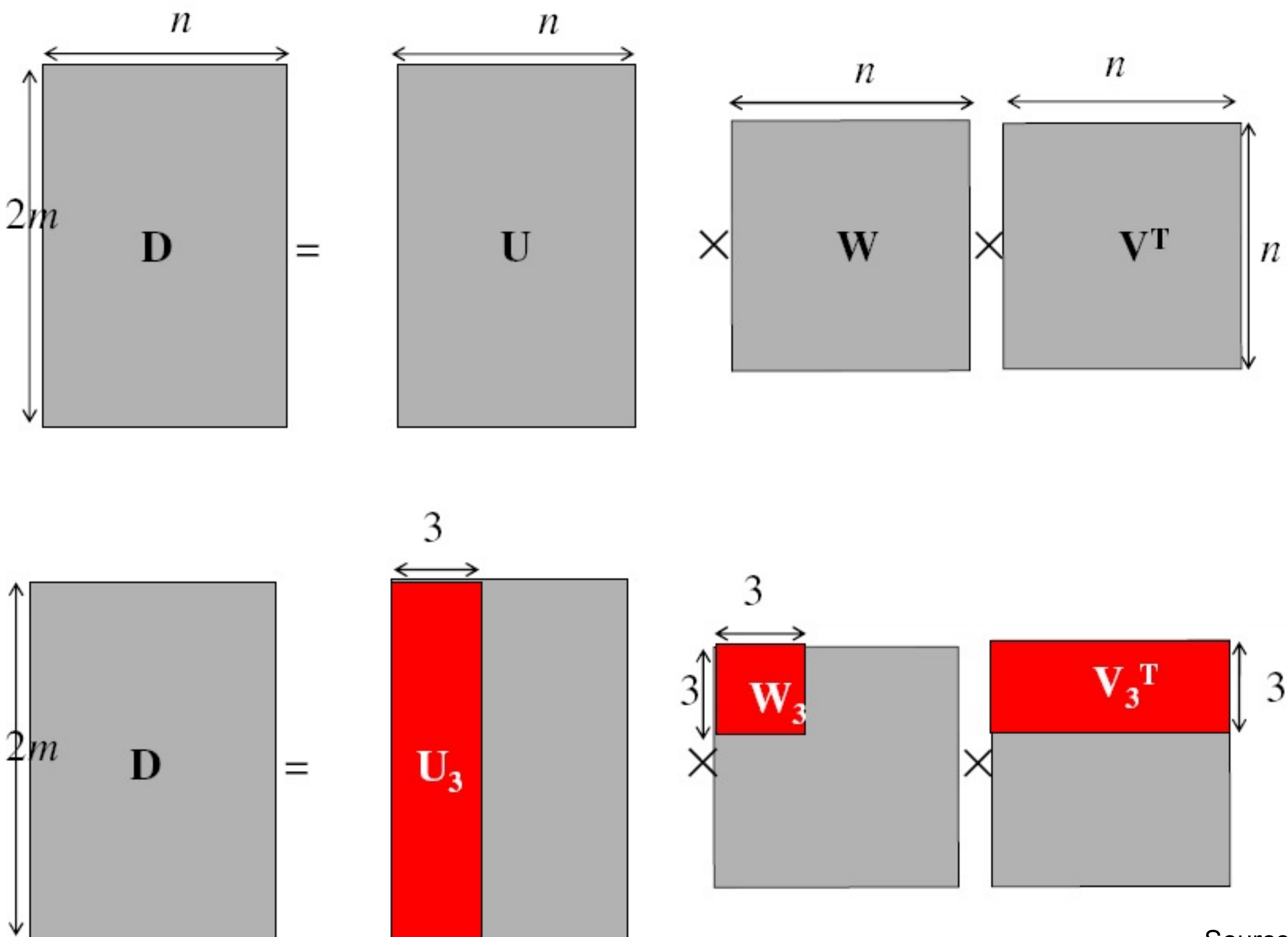


$$\mathbf{D} = \mathbf{MS}$$

Source: M. Hebert

Factorizing the measurement matrix

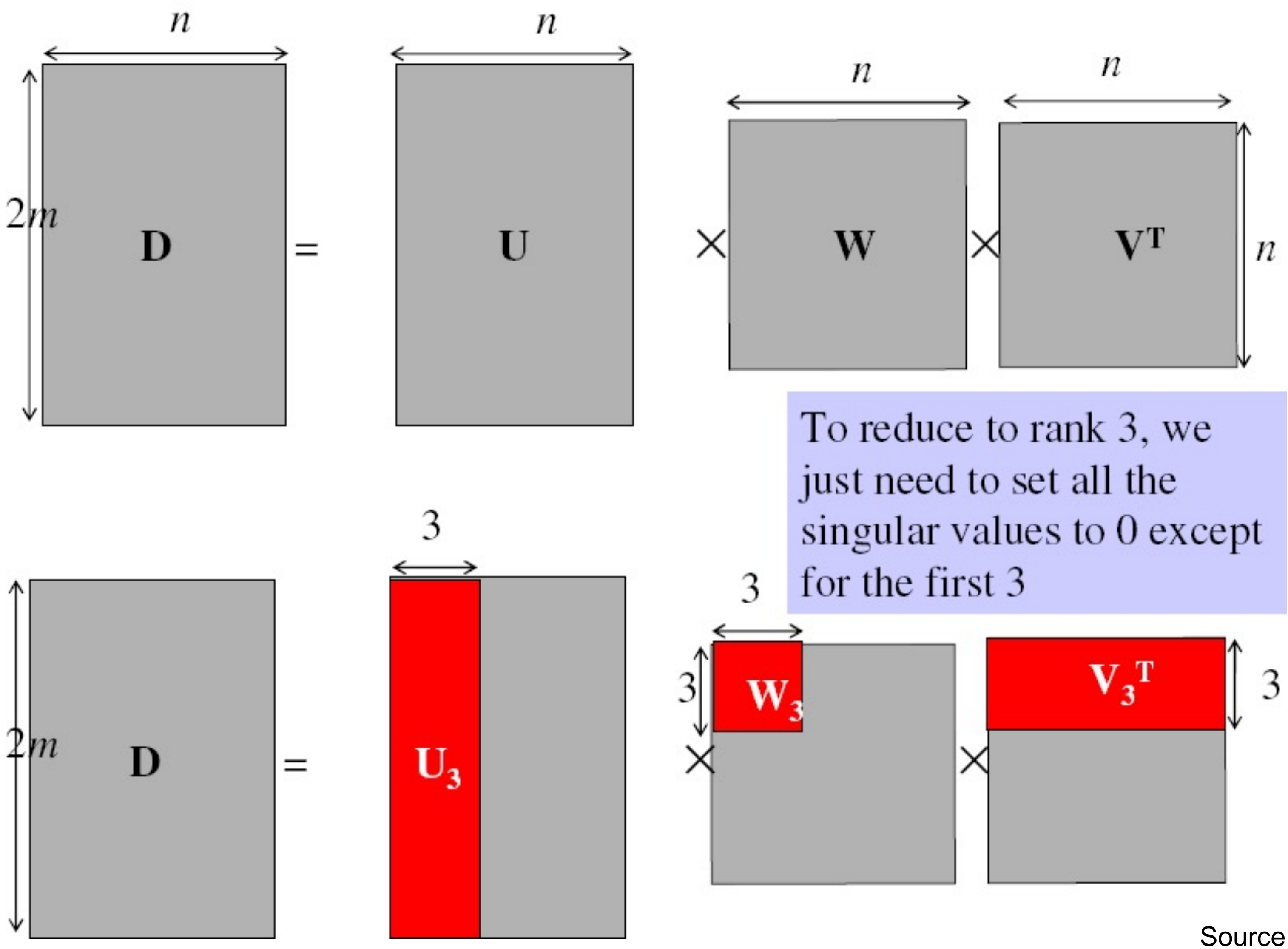
- Singular value decomposition of D:



Source: M. Hebert

Factorizing the measurement matrix

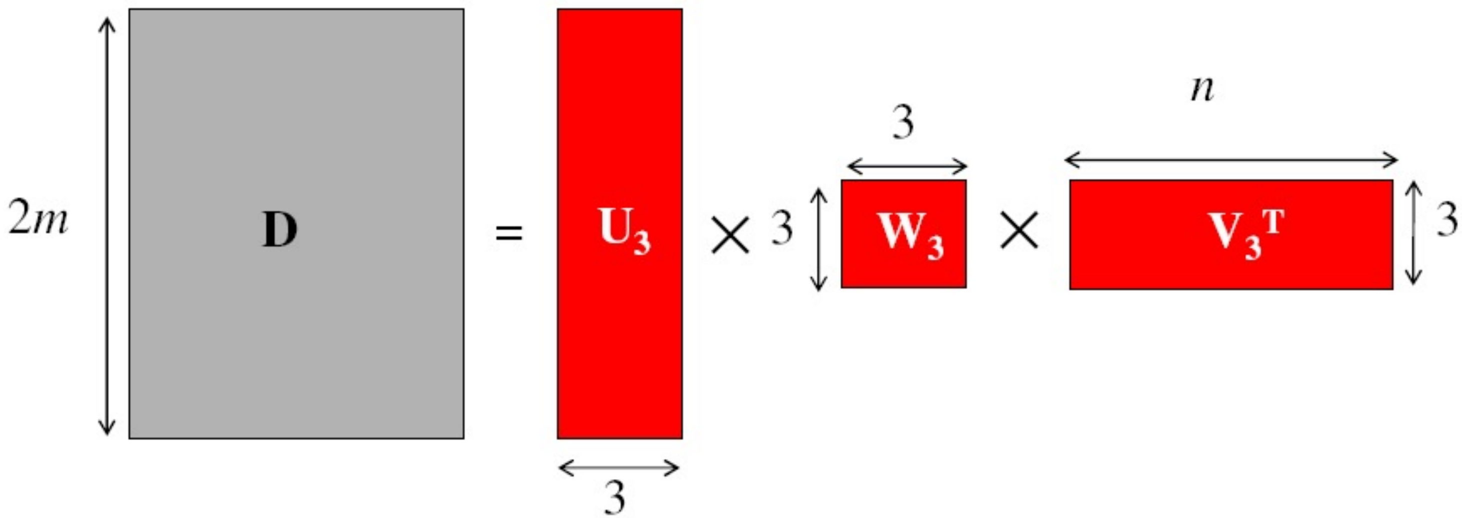
- Singular value decomposition of D:



Source: M. Hebert

Factorizing the measurement matrix

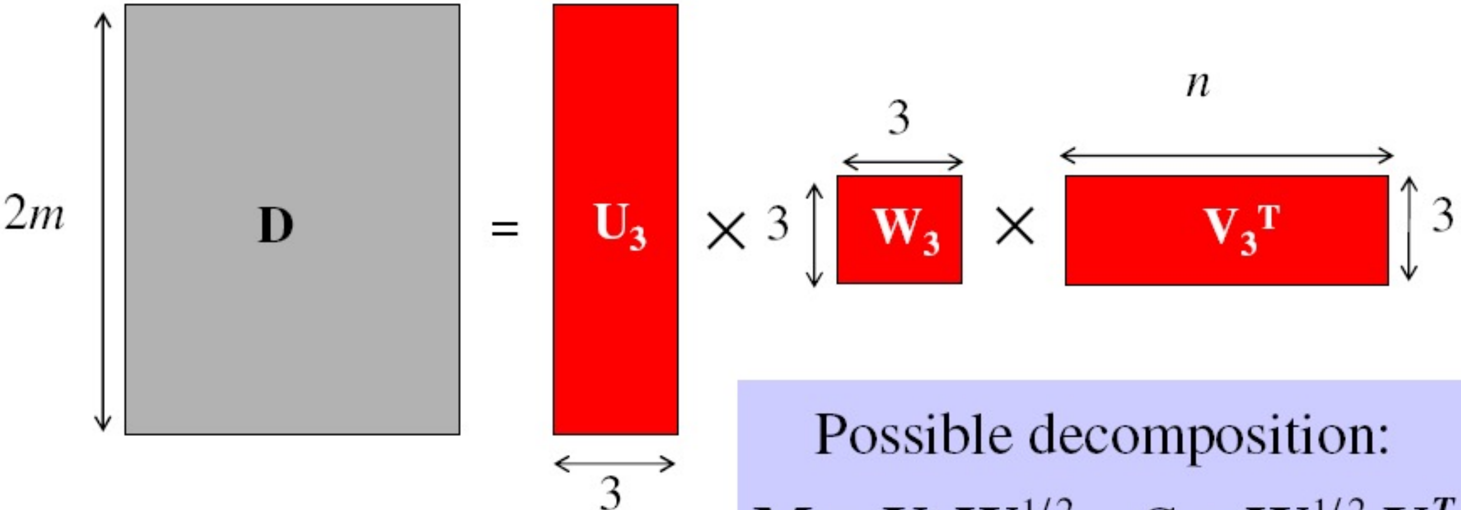
- Obtaining a factorization from SVD:



Source: M. Hebert

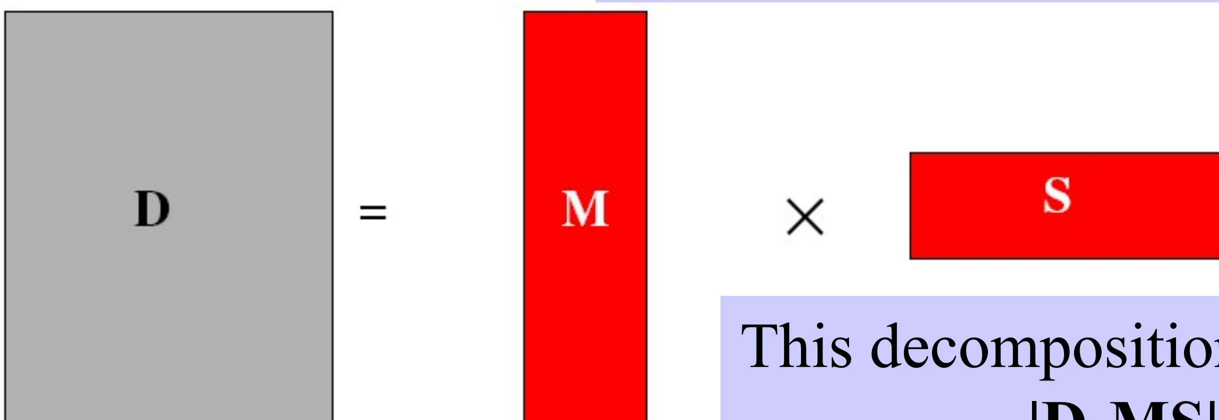
Factorizing the measurement matrix

- Obtaining a factorization from SVD:



Possible decomposition:

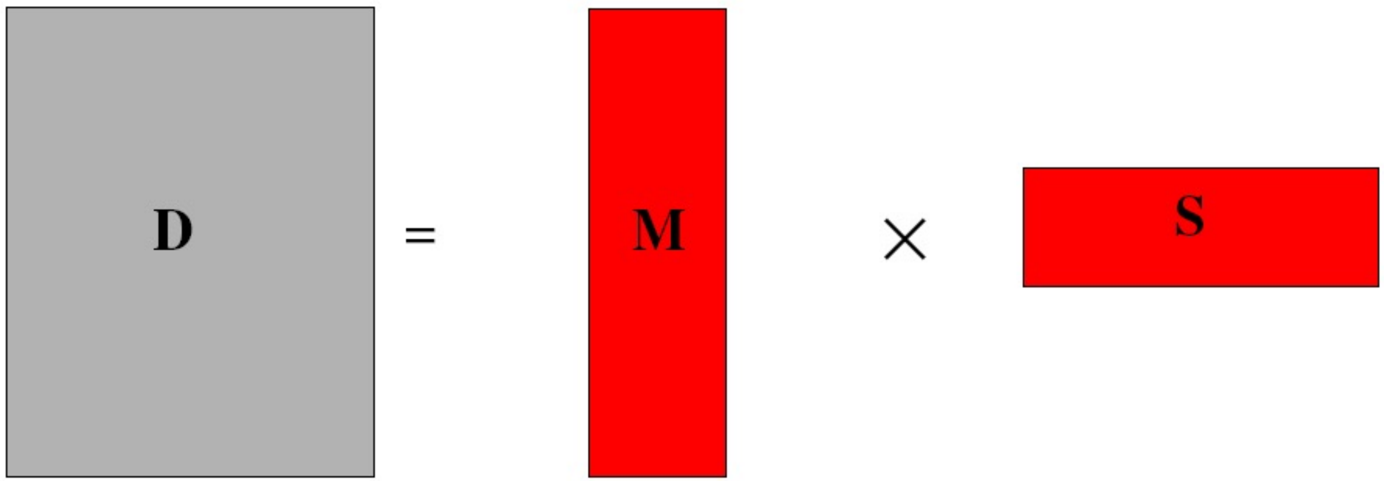
$$\mathbf{M} = \mathbf{U}_3 \mathbf{W}_3^{1/2} \qquad \mathbf{S} = \mathbf{W}_3^{1/2}\, \mathbf{V}_3^{T}$$



This decomposition minimizes
$|\mathbf{D}\text{-}\mathbf{MS}|^2$

Source: M. Hebert

Affine ambiguity

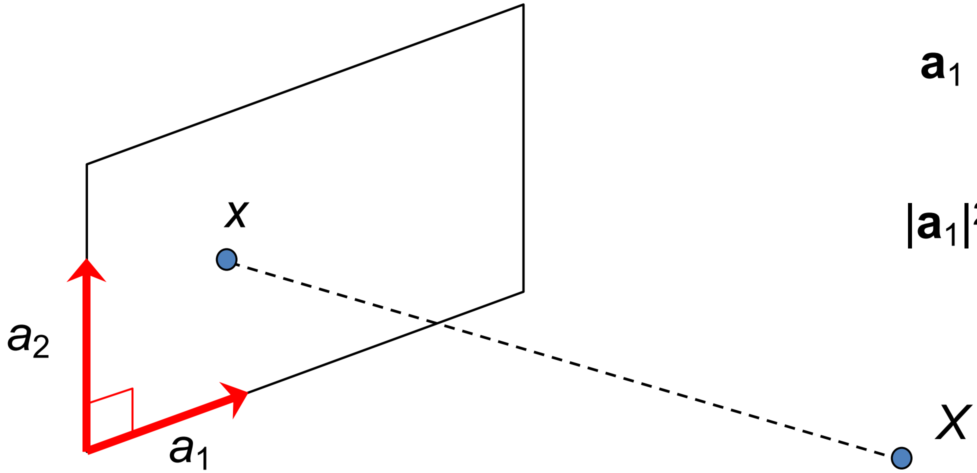


- The decomposition is not unique. We get the same **D** by using any 3 × 3 matrix **C** and applying the transformations **M** → **MC**, **S** →$\mathbf{C}^{-1}\mathbf{S}$
- That is because we have only an affine transformation and we have not enforced any Euclidean constraints (like forcing the image axes to be perpendicular, for example)

Source: M. Hebert

Eliminating the affine ambiguity

- Orthographic: image axes are perpendicular and of unit length



$\mathbf{a}_1 \cdot \mathbf{a}_2 = 0$

$|\mathbf{a}_1|^2 = |\mathbf{a}_2|^2 = 1$

Source: M. Hebert

Solve for orthographic constraints

Three equations for each image i

$$\tilde{\mathbf{a}}_{i1}^T \mathbf{C}\mathbf{C}^T \tilde{\mathbf{a}}_{i1}^T = 1$$
$$\tilde{\mathbf{a}}_{i2}^T \mathbf{C}\mathbf{C}^T \tilde{\mathbf{a}}_{i2}^T = 1 \quad \text{where} \quad \tilde{\mathbf{A}}_i = \begin{bmatrix} \tilde{\mathbf{a}}_{i1}^T \\ \tilde{\mathbf{a}}_{i2}^T \end{bmatrix}$$
$$\tilde{\mathbf{a}}_{i1}^T \mathbf{C}\mathbf{C}^T \tilde{\mathbf{a}}_{i2}^T = 0$$

- Solve for $\mathbf{L} = \mathbf{C}\mathbf{C}^\mathbf{T}$
- Recover **C** from **L** by Cholesky decomposition: $\mathbf{L} = \mathbf{C}\mathbf{C}^\mathbf{T}$
- Update **A** and **X**: $\mathbf{A} = \tilde{\mathbf{A}}\mathbf{C}$, $\mathbf{X} = \mathbf{C}^{-1}\tilde{\mathbf{X}}$

Slide: D. Hoiem

Algorithm summary

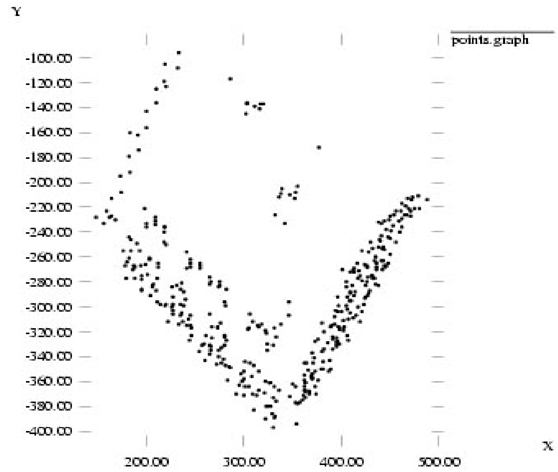
- Given: m images and n features $\mathbf{x}_{ij}$
- For each image i, center the feature coordinates
- Construct a $2m \times n$ measurement matrix $\mathbf{D}$:
  - Column j contains the projection of point j in all views
  - Row i contains one coordinate of the projections of all the n points in image i
- Factorize $\mathbf{D}$:
  - Compute SVD: $\mathbf{D} = \mathbf{U}\ \mathbf{W}\ \mathbf{V}^T$
  - Create $\mathbf{U}_3$ by taking the first 3 columns of $\mathbf{U}$
  - Create $\mathbf{V}_3$ by taking the first 3 columns of $\mathbf{V}$
  - Create $\mathbf{W}_3$ by taking the upper left 3×3 block of $\mathbf{W}$
- Create the motion and shape matrices:
  - $\mathbf{M} = \mathbf{U}_3\mathbf{W}_3^{1/2}$ and $\mathbf{S} = \mathbf{W}_3^{1/2}\,\mathbf{V}_3^T$ (**or** $\mathbf{M} = \mathbf{U}_3$ and $\mathbf{S} = \mathbf{W}_3\mathbf{V}_3^T$)
- Eliminate affine ambiguity

Source: M. Hebert

Reconstruction results

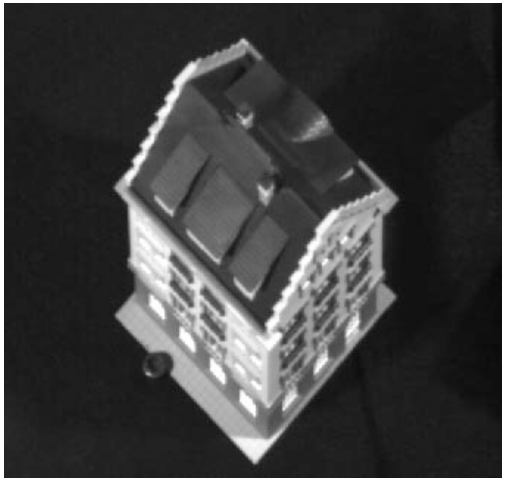
1

60



120

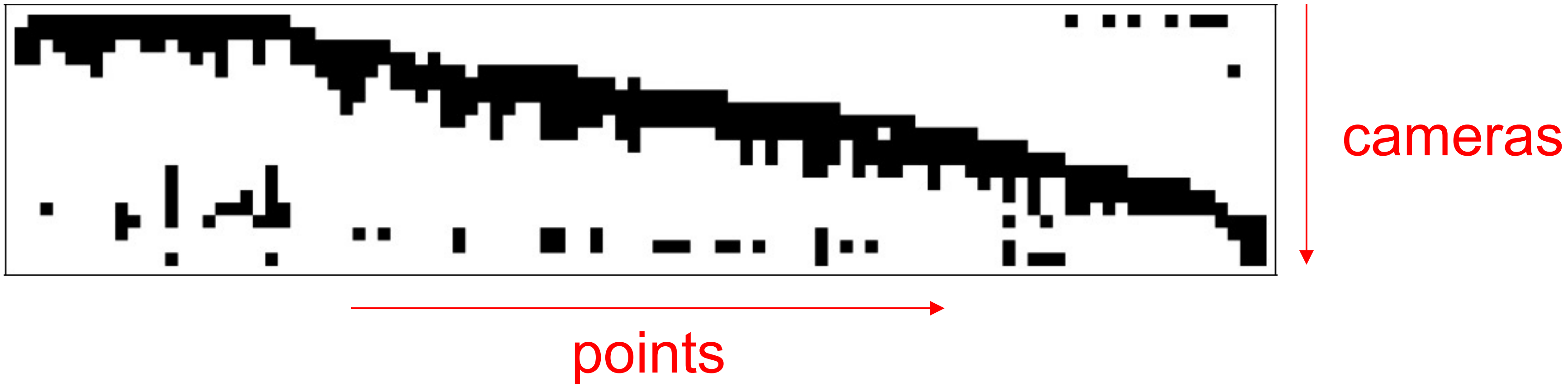
150



C. Tomasi and T. Kanade. [Shape and motion from image streams under orthography: A factorization method.](#) *IJCV*, 9(2):137-154, November 1992.

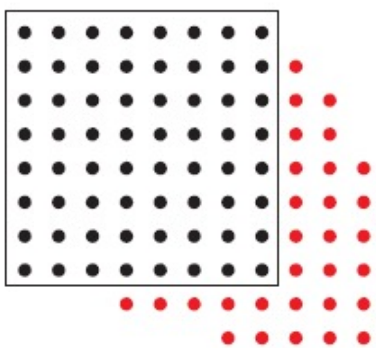
Dealing with missing data

- So far, we have assumed that all points are visible in all views
- In reality, the measurement matrix typically looks something like this:

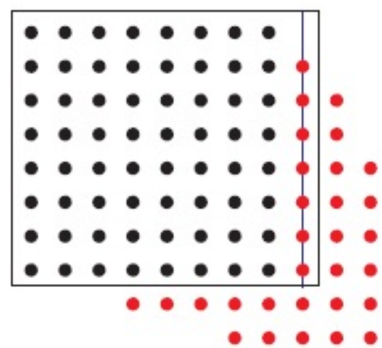


Dealing with missing data

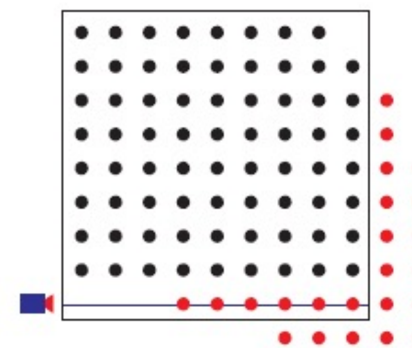
- Possible solution: decompose matrix into dense sub-blocks, factorize each sub-block, and fuse the results
  - Finding dense maximal sub-blocks of the matrix is NP-complete (equivalent to finding maximal cliques in a graph)
- Incremental bilinear refinement



(1) Perform factorization on a dense sub-block



(2) Solve for a new 3D point visible by at least two known cameras (linear least squares)

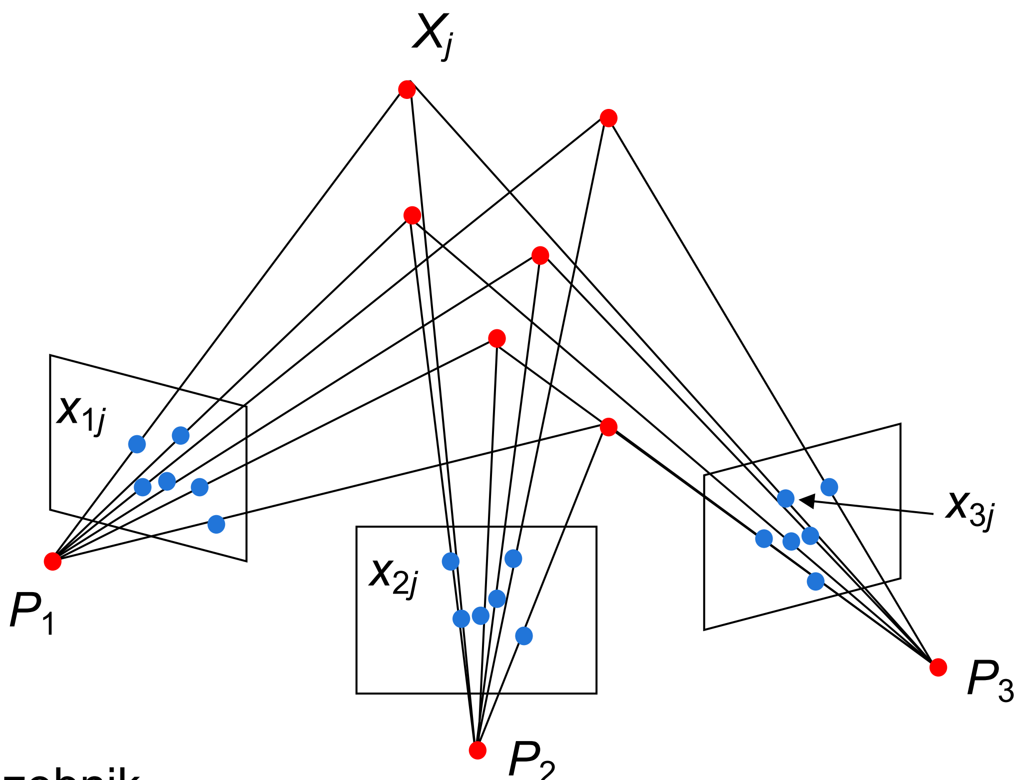


(3) Solve for a new camera that sees at least three known 3D points (linear least squares)

F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. [Segmenting, Modeling, and Matching Video Clips Containing Multiple Moving Objects.]() PAMI 2007.

Projective structure from motion

- Given: *m* images of *n* fixed 3D points
  - $\mathbf{x}_{ij} = \mathbf{P}_i \mathbf{X}_j, \; i = 1, \dots, m, \quad j = 1, \dots, n$
- Problem: estimate *m* projection matrices $\mathbf{P}_i$ and *n* 3D points $\mathbf{X}_j$ from the *mn* corresponding points $\mathbf{x}_{ij}$



Slides from Lana Lazebnik

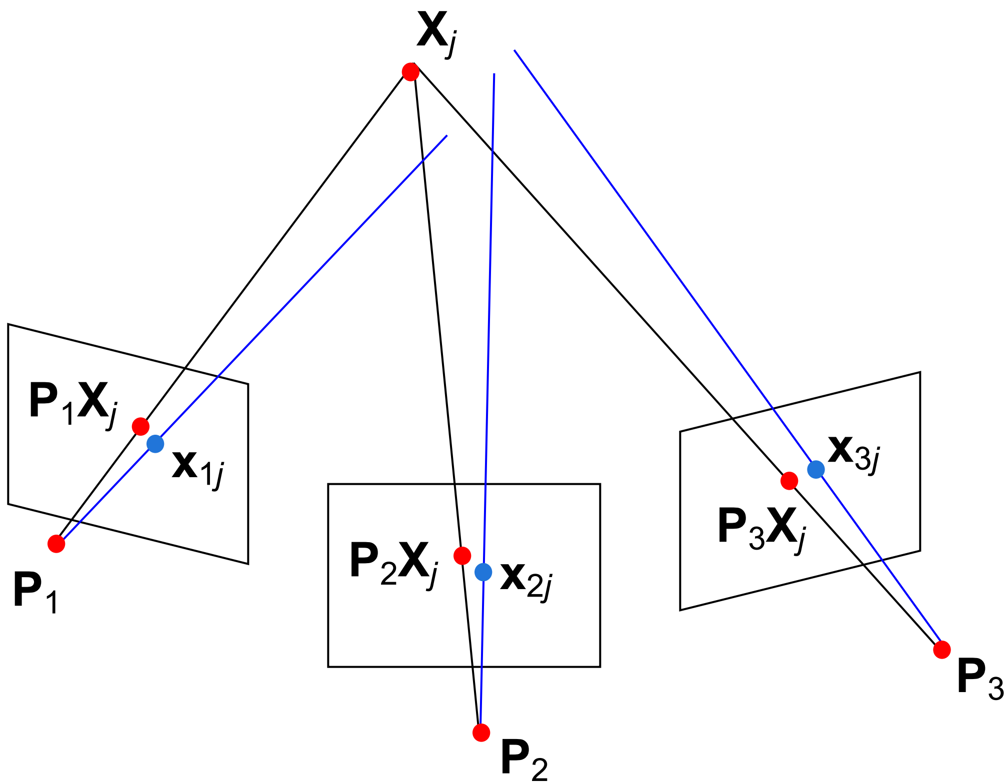
Projective structure from motion

- Given: m images of n fixed 3D points
  - $\mathbf{x}_{ij} = \mathbf{P}_i \mathbf{X}_j, \qquad i = 1, \ldots, m, \quad j = 1, \ldots, n$
- Problem: estimate m projection matrices $\mathbf{P}_i$ and n 3D points $\mathbf{X}_j$ from the mn corresponding points $\mathbf{x}_{ij}$
- With no calibration info, cameras and points can only be recovered up to a 4x4 projective transformation **Q**:
  - $\mathbf{X} \rightarrow \mathbf{QX}, \mathbf{P} \rightarrow \mathbf{PQ}^{-1}$
- We can solve for structure and motion when
  - $2mn >= 11m + 3n - 15$
- For two cameras, at least 7 points are needed

Bundle adjustment

- Non-linear method for refining structure and motion
- Minimizing reprojection error

$$E(\mathbf{P},\mathbf{X}) = \sum_{i=1}^{m}\sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, \mathbf{P}_i\mathbf{X}_j\right)^2$$



Self-calibration

- Self-calibration (auto-calibration) is the process of determining intrinsic camera parameters directly from uncalibrated images
- For example, when the images are acquired by a single moving camera, we can use the constraint that the intrinsic parameter matrix remains fixed for all the images
  - Compute initial projective reconstruction and find 3D projective transformation matrix $\mathbf{Q}$ such that all camera matrices are in the form $\mathbf{P}_i = \mathbf{K}\,[\mathbf{R}_i \mid \mathbf{t}_i]$
- Can use constraints on the form of the calibration matrix: zero skew

Review: Structure from motion

- Ambiguity
- Affine structure from motion
  - Factorization
- Dealing with missing data
  - Incremental structure from motion
- Projective structure from motion
  - Bundle adjustment
  - Self-calibration

Summary: 3D geometric vision

- Single-view geometry
  - The pinhole camera model
    - Variation: orthographic projection
  - The perspective projection matrix
  - Intrinsic parameters
  - Extrinsic parameters
  - Calibration
- Multiple-view geometry
  - Triangulation
  - The epipolar constraint
    - Essential matrix and fundamental matrix
  - Stereo
    - Binocular, multi-view
  - Structure from motion
    - Reconstruction ambiguity
    - Affine SFM
    - Projective SFM