— CM50248 — 2017/2018 —

Visual Understanding 1

Dr Christian Richardt

Today's lecture

- Recap:
  - 2D transformations
  - RANSAC
  - Image warping, interpolation & stitching
- Camera geometry
- Camera calibration
- Epipolar geometry
- Fundamental matrix estimation
- Triangulation of 3D points

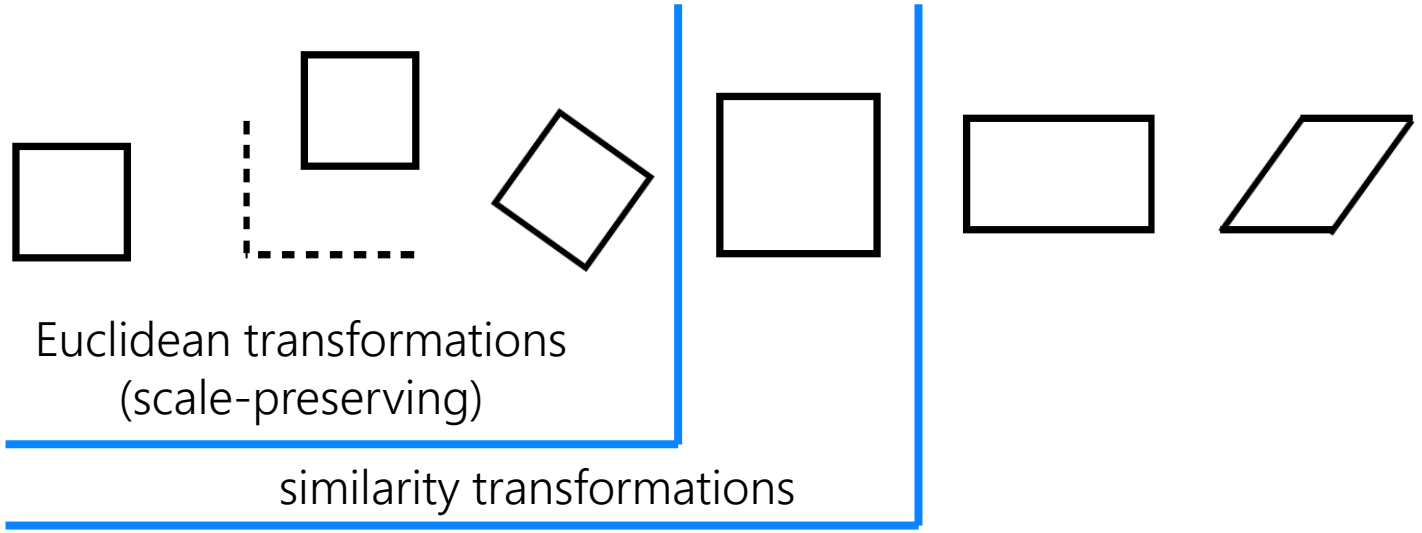
- Recommended reading:
  - CM50248 course notes:
    - 4.1 Projection
    - 4.2 Epipolar geometry
    - 4.3.2 Reconstruction
  - CM20219 course notes:
    - 4.3 Projection – 3D on a 2D display

Affine transformations

- What range of transformations can we represent using this?

$$\begin{bmatrix} p'_1 \\ p'_2 \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & t_1 \\ a_{21} & a_{22} & t_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ 1 \end{bmatrix}$$

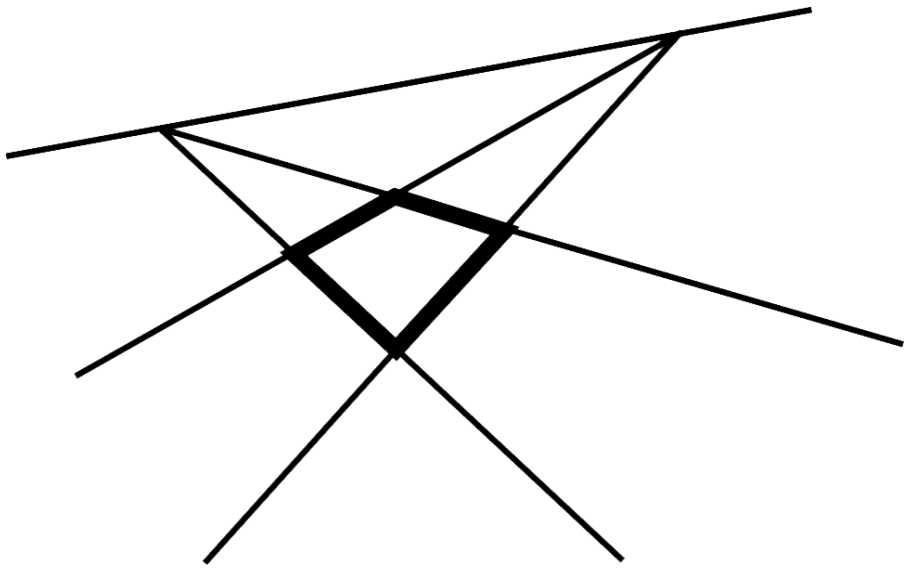
- 6 degrees of freedom: translation (2), rotation, scale, stretch, shear



Homographies

- Linear transform in homogeneous coordinates
- Represented by 3x3 matrix:

$$\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$



- 8 degrees of freedom (scale of **H** arbitrary)
- 4 points can map to any 4 points
- Includes translation (2), rotation, scale, stretch, shear

RANSAC for homography estimation

RANSAC loop:

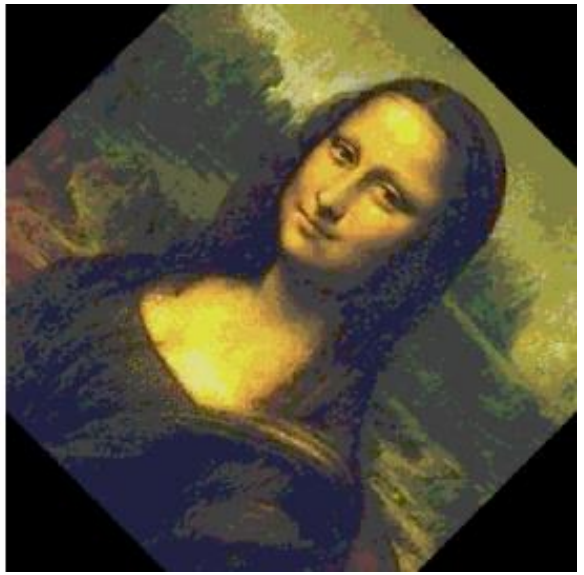
1. Select four feature pairs (at random)
2. Compute homography H (exact)
3. Compute *inliers* where $\|\mathbf{p}' - \mathbf{H}\mathbf{p}\| < \varepsilon$
4. Keep largest set of inliers
5. Re-compute least-squares **H** estimate on all of the inliers

Slide by Alexei Efros, Rick Szeliski and/or Steve Seitz

Image warping

- How do we warp images given a known affine transform?

Forward mapping



Backward mapping

$$\tilde{\mathbf{p}}' = \mathbf{T}\tilde{\mathbf{p}} \qquad \tilde{\mathbf{p}} = \mathbf{T}^{-1}\tilde{\mathbf{p}}'$$

(if we round coordinates, image will contain gaps!)

[CM20219 Notes, Section 4.5]

Bundle adjustment

New images initialised with rotation,
focal length of best matching image

Slide by Matthew Brown and David Lowe

Bundle adjustment

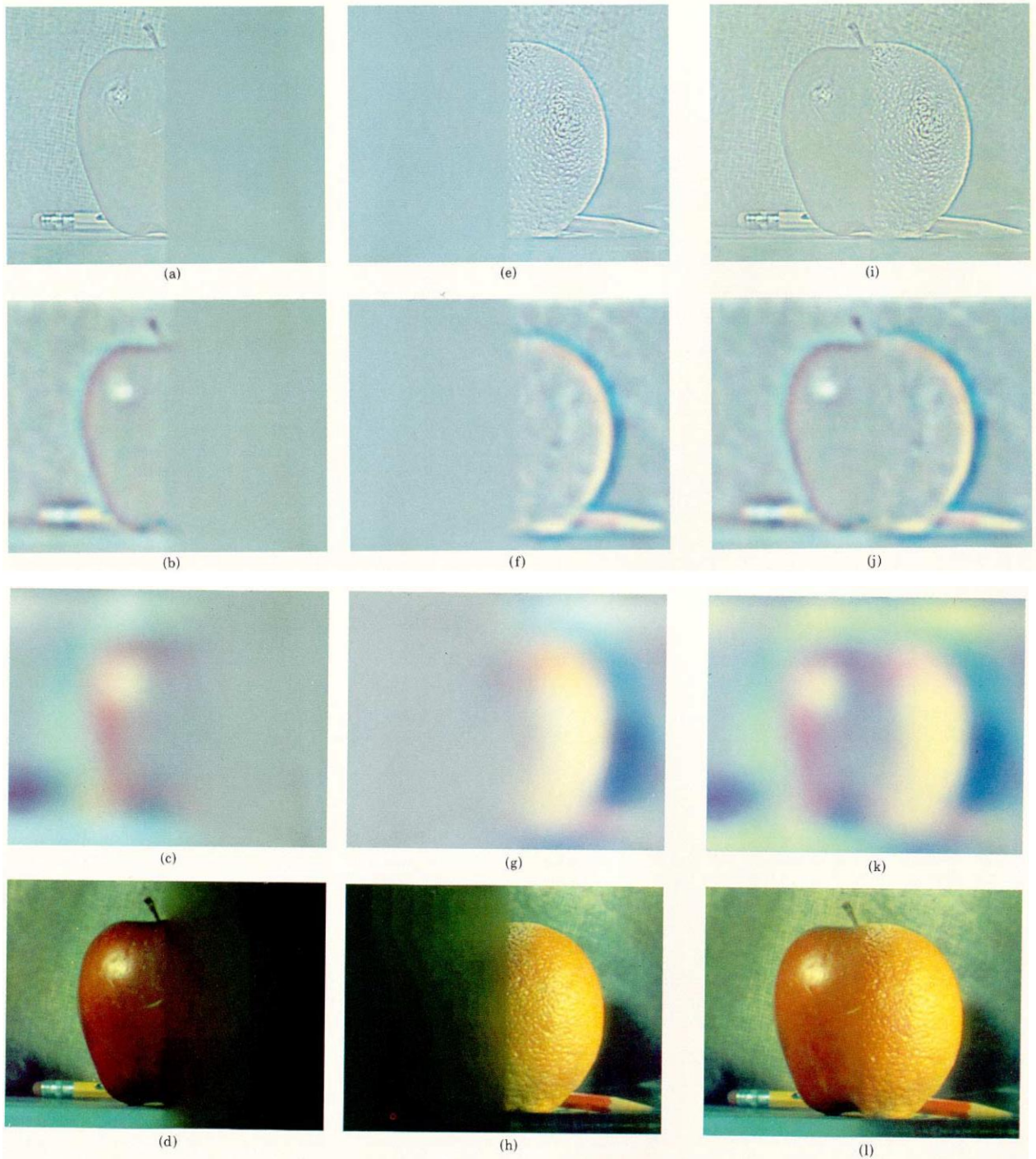
New images initialised with rotation,
focal length of best matching image

Slide by Matthew Brown and David Lowe

Panorama stitching

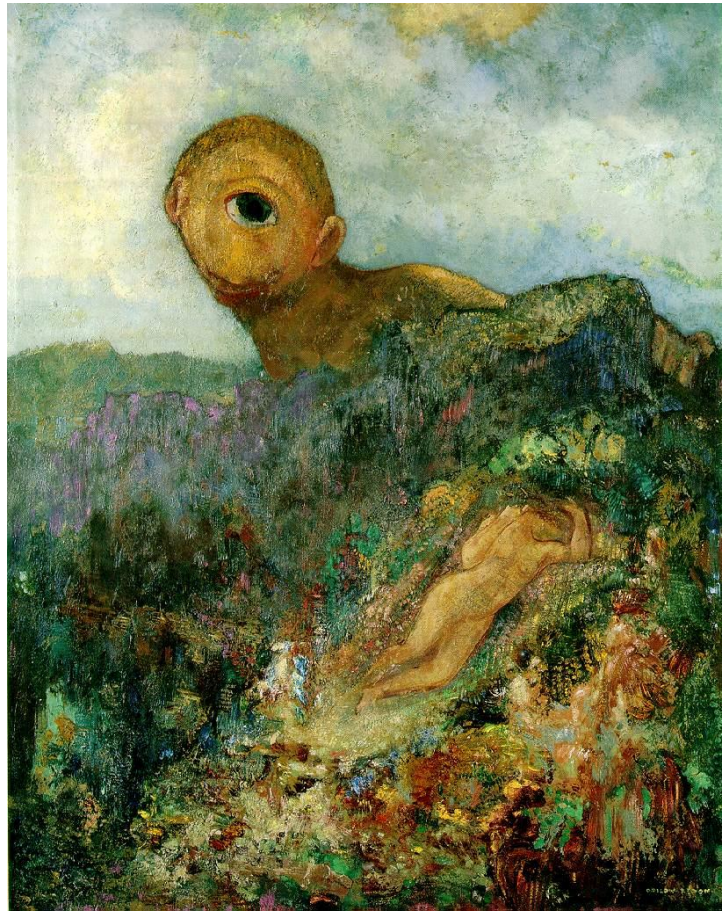
- Multi-band blending [Burt & Adelson 1983]
  - Blend frequency bands over different scales

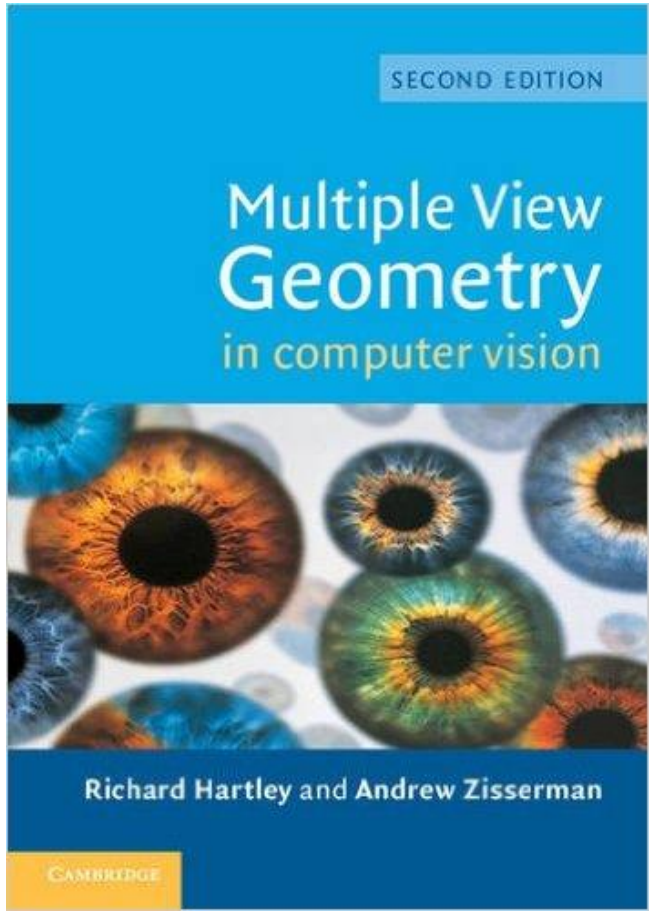
Slide by Matthew Brown and David Lowe



Burt & Adelson [ACM ToG 1983]

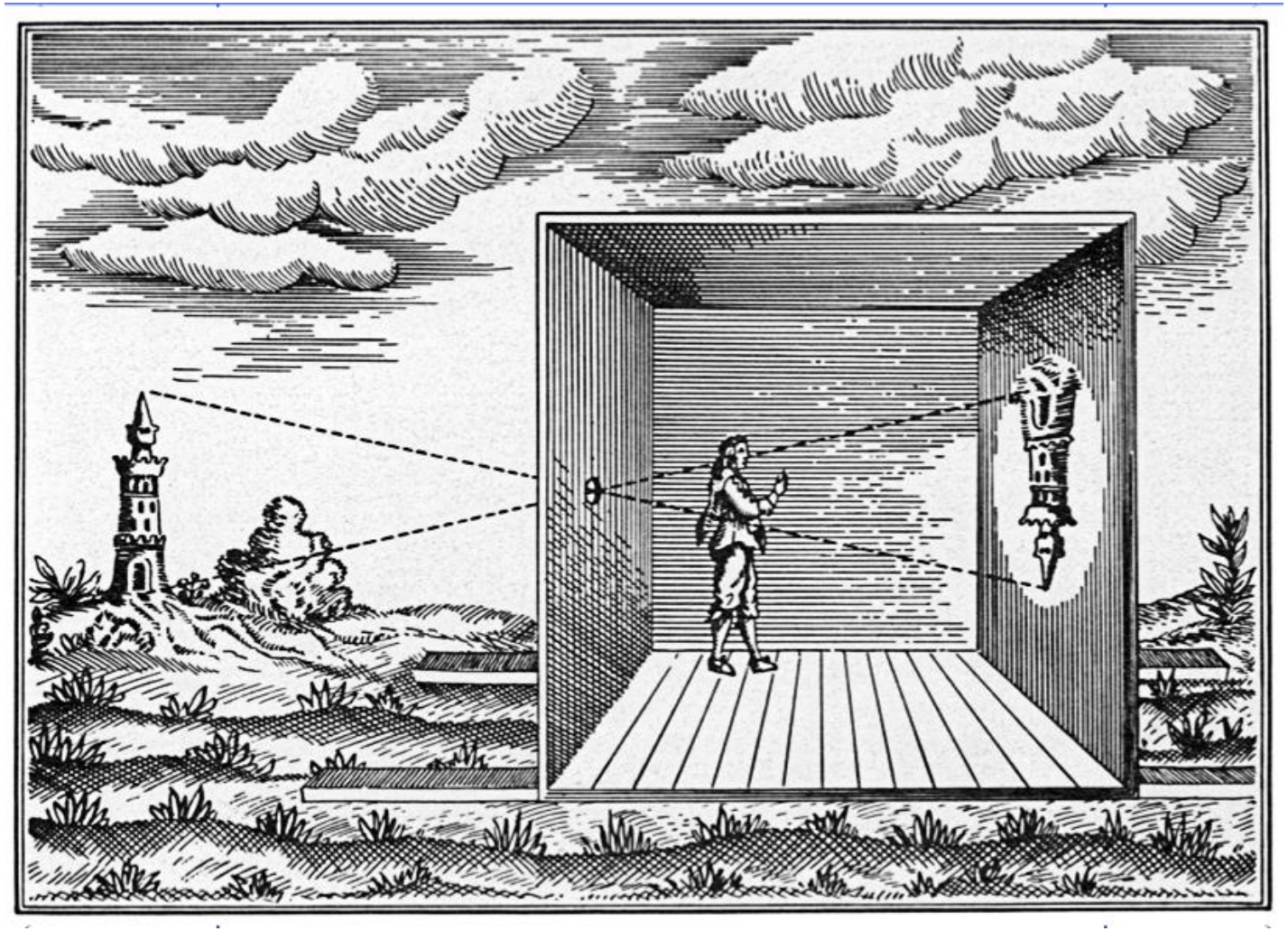
Camera geometry



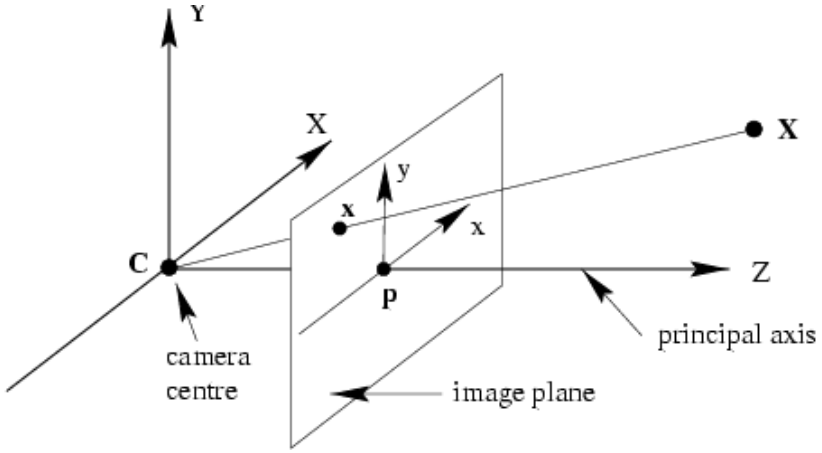


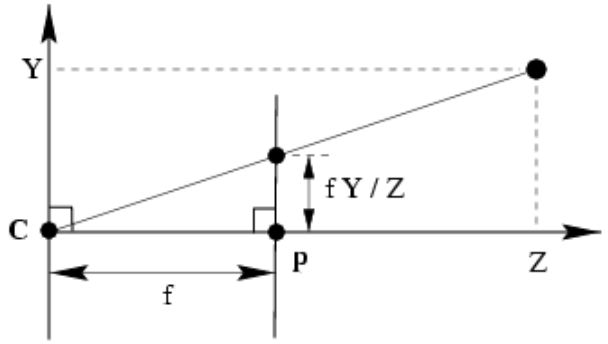
Multiple View Geometry in Computer Vision
Hartley & Zisserman, 2004

Camera obscura



Pinhole camera model





$$(X, Y, Z)^T \mapsto (fX / Z,\ fY / Z)^T$$

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & \\ & f & \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

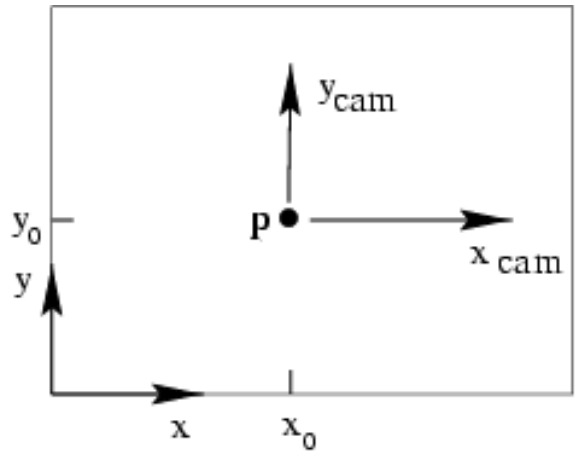
$$\mathrm{P} = \mathrm{diag}(f, f, 1)[\mathrm{I} \mid 0] \qquad \mathrm{x} = \mathrm{PX}$$

Slide by Marc Pollefeys

Principal point offset

$$(X,Y,Z)^T \mapsto (fX/Z + p_x, fY/Z + p_y)^T$$

$(p_x, p_y)^T$ principal point



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

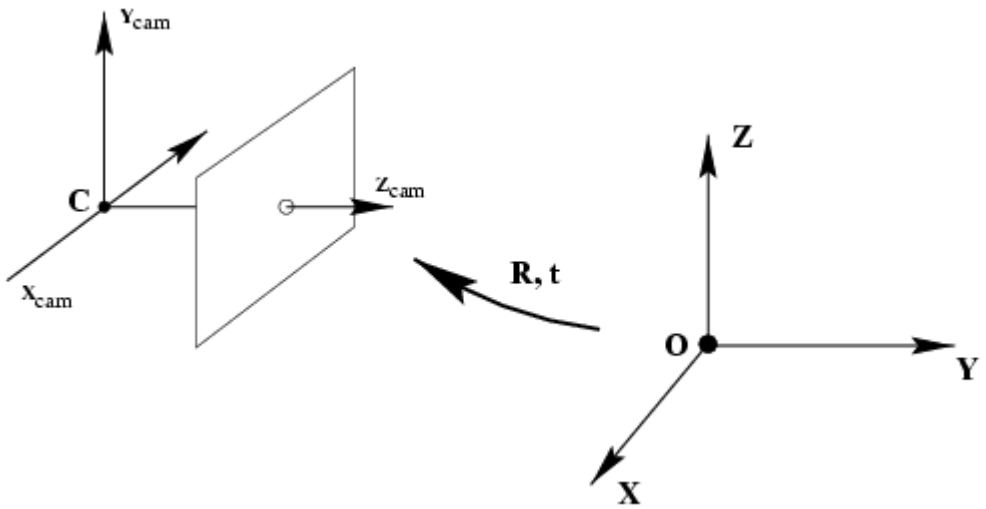
$$\mathrm{x} = \mathrm{K}[\mathrm{I} \mid 0]\mathrm{X}_{\mathrm{cam}}$$

$$K = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}$$

calibration matrix

Slide by Marc Pollefeys

Camera rotation and translation



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

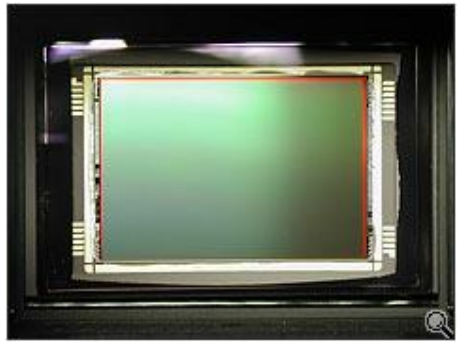
$$\mathrm{X}_{\mathrm{cam}} = \begin{bmatrix} \mathrm{R} & -\mathrm{R}\tilde{\mathrm{C}} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} \mathrm{R} & -\mathrm{RC} \\ 0 & 1 \end{bmatrix} \mathrm{X}$$

$$\mathrm{x} = \mathrm{K}\left[\mathrm{I} \mid 0\right]\mathrm{X}_{\mathrm{cam}} \qquad \mathrm{x} = \mathrm{KR}\left[\mathrm{I} \mid -\tilde{\mathrm{C}}\right]\mathrm{X}$$

$$\mathrm{x} = \mathrm{PX} \qquad \mathrm{P} = \mathrm{K}\left[\mathrm{R} \mid \mathrm{t}\right] \qquad \mathrm{t} = -\mathrm{R}\tilde{\mathrm{C}}$$

Slide by Marc Pollefeys

Pixel coordinates



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

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

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

Finite projective camera

$$K = \begin{bmatrix} \alpha_x & s & p_x \\ & \alpha_x & p_y \\ & & 1 \end{bmatrix}$$

$$\mathrm{P} = \underbrace{\mathrm{KR}}\left[\mathrm{I} \mid -\tilde{\mathrm{C}}\right] \qquad \text{11 dof (5+3+3)}$$

non-singular

decompose P in K,R,C?

$$\mathrm{P} = \left[\mathrm{M} \mid \mathrm{p}_4\right] \qquad \left[\mathrm{K},\mathrm{R}\right] = RQ\left(\mathrm{M}\right) \qquad \tilde{\mathrm{C}} = -\mathrm{M}^{-1}\mathrm{p}_4$$

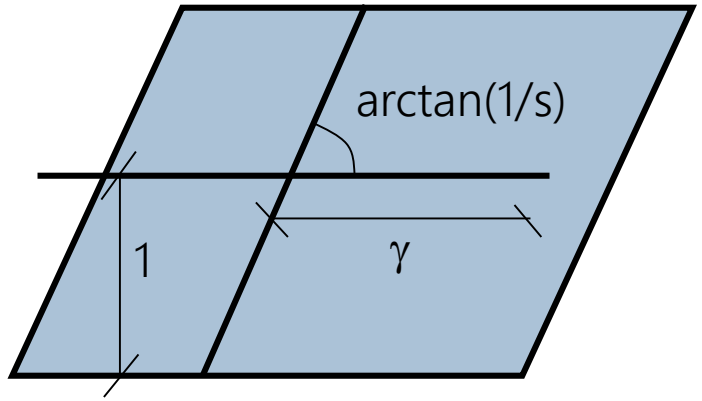
{finite cameras}={P_{4x3} | det M≠0}

If rank P=3, but rank M<3, then cam at infinity

Slide by Marc Pollefeys

When is skew non-zero?

$$K = \begin{bmatrix} \alpha_x & s & p_x \\ & \alpha_x & p_y \\ & & 1 \end{bmatrix}$$



for CCD/CMOS, always s=0

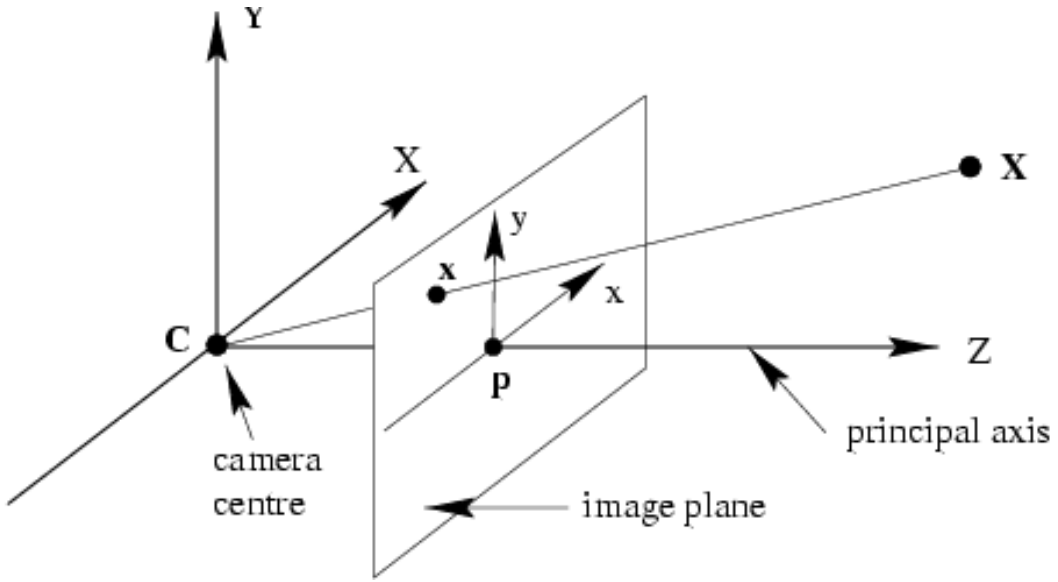
Image from image, s≠0 possible
(non coinciding principal axis)

resulting camera: $\mathbf{HP}$

Slide by Marc Pollefeys

Camera anatomy

- Camera center
- Column points
- Principal plane
- Axis plane
- Principal point
- Principal ray



Slide by Marc Pollefeys

Camera centre

null-space camera projection matrix

$$\mathbf{PC} = \mathbf{0}$$

$$\mathbf{X} = \lambda\mathbf{A} + (1-\lambda)\mathbf{C}$$

$$\mathbf{x} = \mathbf{PX} = \lambda\mathbf{PA} + (1-\lambda)\mathbf{PC}$$

For all A, all points on AC project on image of A,

therefore C is camera centre

Image of camera center is $(0,0,0)^{\mathsf{T}}$, i.e. undefined

Finite cameras: $C = \begin{pmatrix} -\mathbf{M}^{-1}\mathbf{p}_4 \\ 1 \end{pmatrix}$

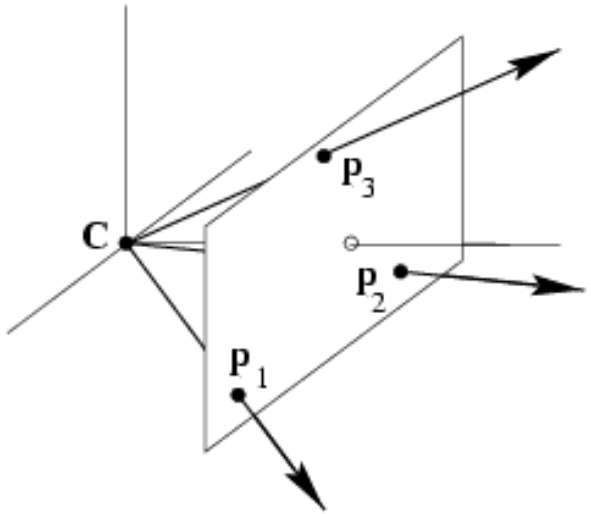
Infinite cameras: $C = \begin{pmatrix} \mathbf{d} \\ 0 \end{pmatrix}, \mathbf{Md} = 0$

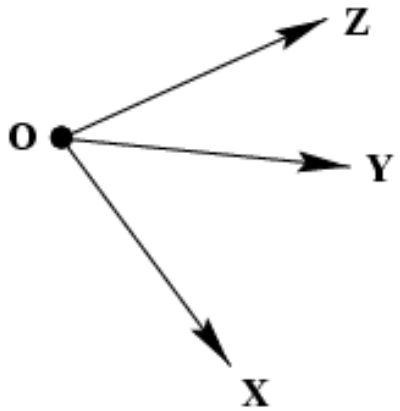
Slide by Marc Pollefeys

Column vectors of P

$$[p_2] = [P_1 P_2 P_3 P_4] \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

Image points corresponding to x-, y-, z- directions and origin

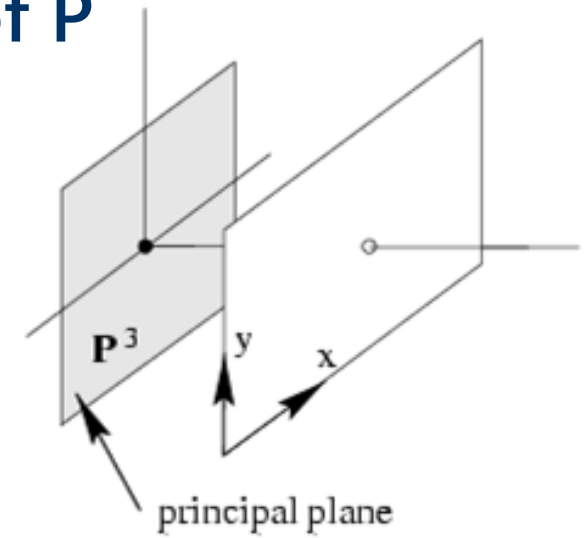




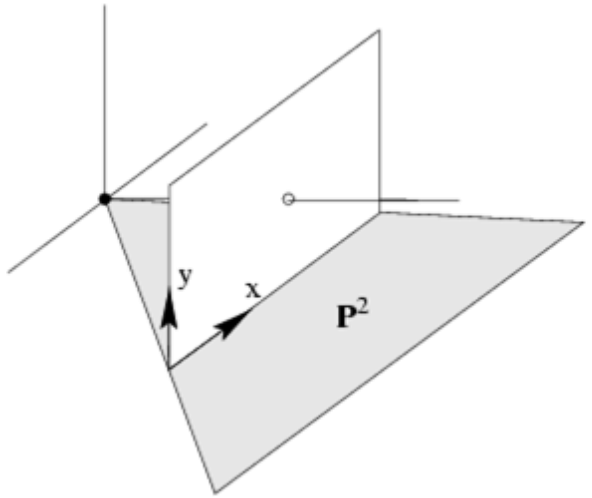
Slide by Marc Pollefeys

Row vectors of P

$$\begin{bmatrix} x \\ y \\ 0 \end{bmatrix} = \begin{bmatrix} \mathrm{p}^{1\mathsf{T}} \\ \mathrm{p}^{2\mathsf{T}} \\ \mathrm{p}^{3\mathsf{T}} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$



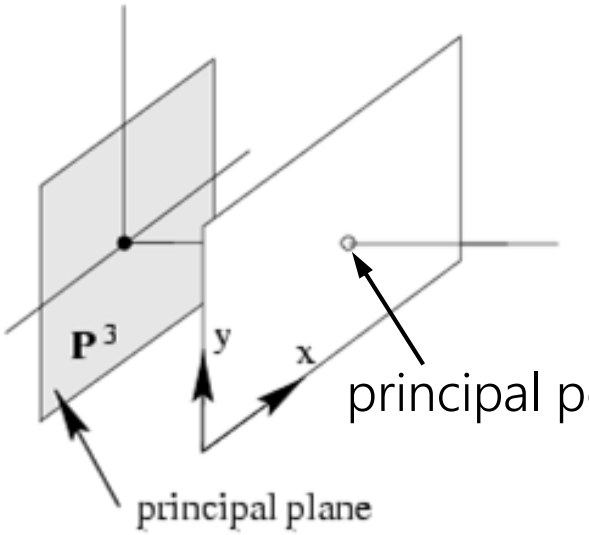
$$\begin{bmatrix} 0 \\ y \\ w \end{bmatrix} = \begin{bmatrix} \mathrm{p}^{1\mathsf{T}} \\ \mathrm{p}^{2\mathsf{T}} \\ \mathrm{p}^{3\mathsf{T}} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

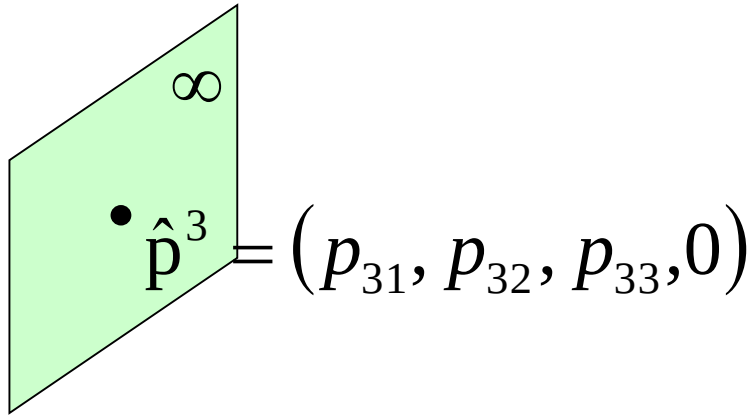


note: p^1, p^2 dependent on image parametrisation

Slide by Marc Pollefeys

The principal point



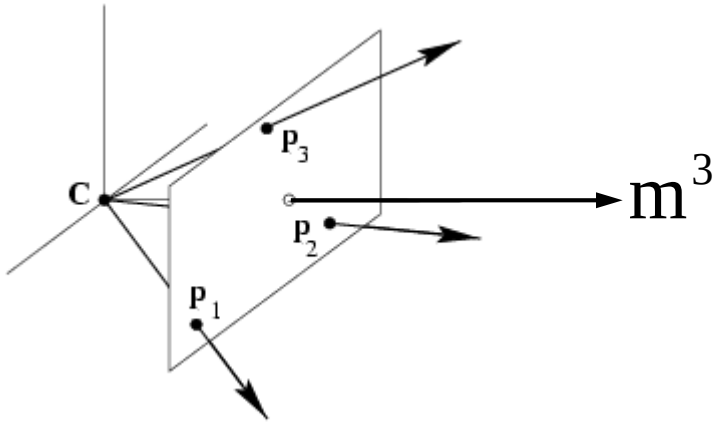


$$x_0 = P\hat{p}^3 = Mm^3$$

Slide by Marc Pollefeys

The principal axis vector

vector defining front side of camera



$$x = P_{cam} X_{cam} = K[I \mid 0] X_{cam}$$

$$P_{cam} \mapsto k P_{cam}$$

$$P = kKR[I \mid -\tilde{C}] = [M \mid p_4]$$

$$P_{cam} \mapsto k P_{cam}$$

$$v = \det(M) m^3 = (0,0,1)^T$$

$$v \mapsto k^4 v \quad \text{(direction unaffected)}$$

$$v \mapsto \det(kM) k m^3 = k^4 v$$

because $\det(R) > 0$

Slide by Marc Pollefeys

Action of projective camera on a point

Forward projection

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

$$\mathbf{x} = \mathbf{P}\mathbf{D} = \left[\mathbf{M} \mid \mathbf{p}_4\right]\mathbf{D} = \mathbf{M}\mathbf{d}$$

Back-projection

$$\mathbf{P}\mathbf{C} = \mathbf{0}$$

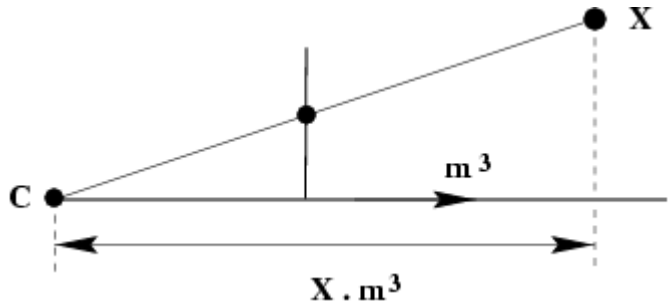
$$\mathbf{X} = \mathbf{P}^{+}\mathbf{x} \qquad \mathbf{P}^{+} = \mathbf{P}^{\mathsf{T}}\left(\mathbf{P}\mathbf{P}^{\mathsf{T}}\right)^{-1} \qquad \mathbf{P}\mathbf{P}^{+} = \mathbf{I}$$

(pseudo-inverse)

$$\mathbf{X}(\lambda) = \mathbf{P}^{+}\mathbf{x} + \lambda\mathbf{C}$$

Slide by Marc Pollefeys

Depth of points



$$w = \mathrm{P}^{3\mathrm{T}}\mathrm{X} = \mathrm{P}^{3\mathrm{T}}\left(\mathrm{X}-\mathrm{C}\right) = \mathrm{m}^{3\mathrm{T}}\left(\tilde{\mathrm{X}}-\tilde{\mathrm{C}}\right)$$

(PC=0) (dot product)

If $\det M > 0; \left\|\mathrm{m}^3\right\| = 1$,
then m^3 unit vector in positive direction

$$\mathrm{depth}\left(\mathrm{X};\mathrm{P}\right) = \frac{\mathrm{sign}(\det \mathrm{M})w}{T\left\|\mathrm{m}^3\right\|}$$

$$\mathrm{X} = \left(X,Y,Z,T\right)^{\mathrm{T}}$$

Slide by Marc Pollefeys

Camera matrix decomposition

Finding the camera center

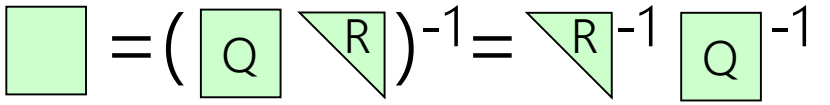
$$\mathrm{PC} = 0$$ (use SVD to find null-space)

$$X = \det([\mathrm{p}_2, \mathrm{p}_3, \mathrm{p}_4]) \quad Y = -\det([\mathrm{p}_1, \mathrm{p}_3, \mathrm{p}_4])$$
$$Z = \det([\mathrm{p}_1, \mathrm{p}_2, \mathrm{p}_4]) \quad T = -\det([\mathrm{p}_1, \mathrm{p}_2, \mathrm{p}_3])$$

Finding the camera orientation and internal parameters

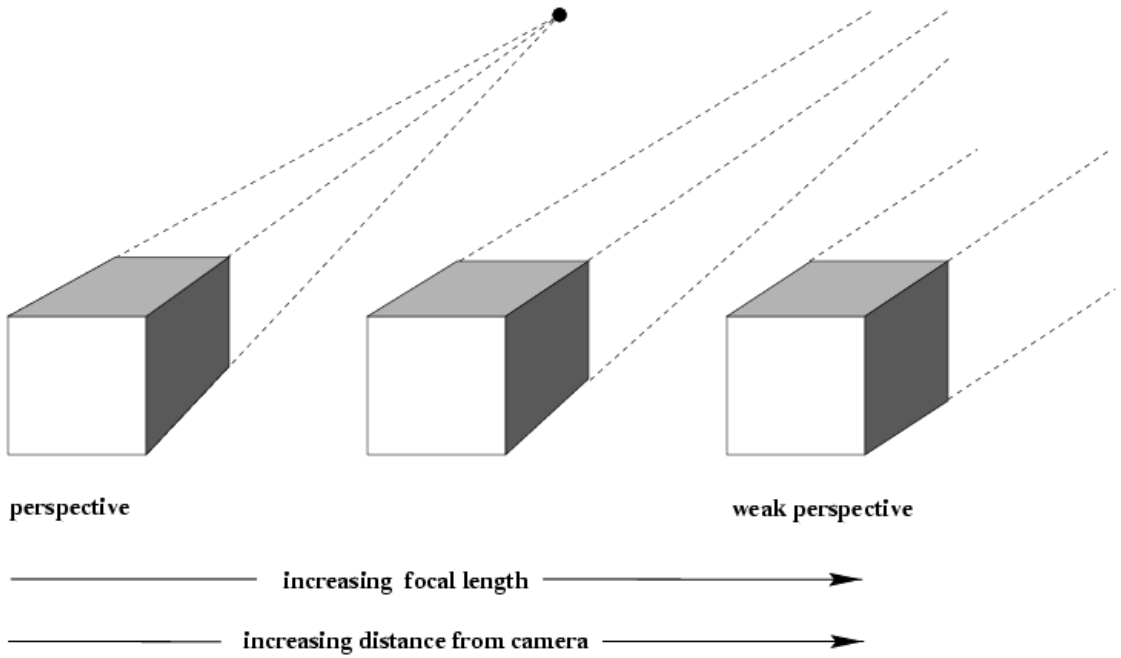
$$\mathrm{M} = \mathrm{KR}$$ (use RQ decomposition ~QR)

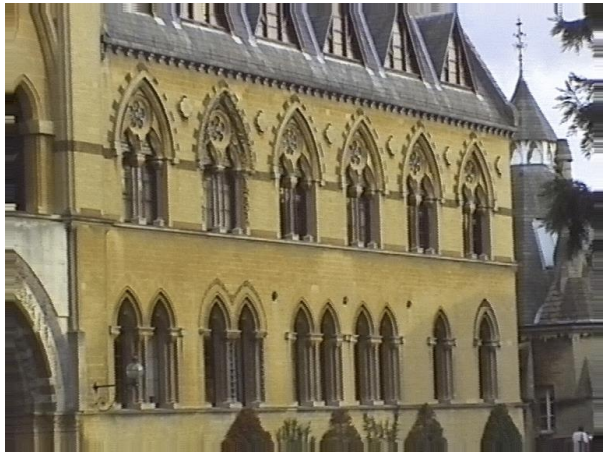
(if only QR, invert)



Slide by Marc Pollefeys

Affine cameras





Slide by Marc Pollefeys

Camera calibration

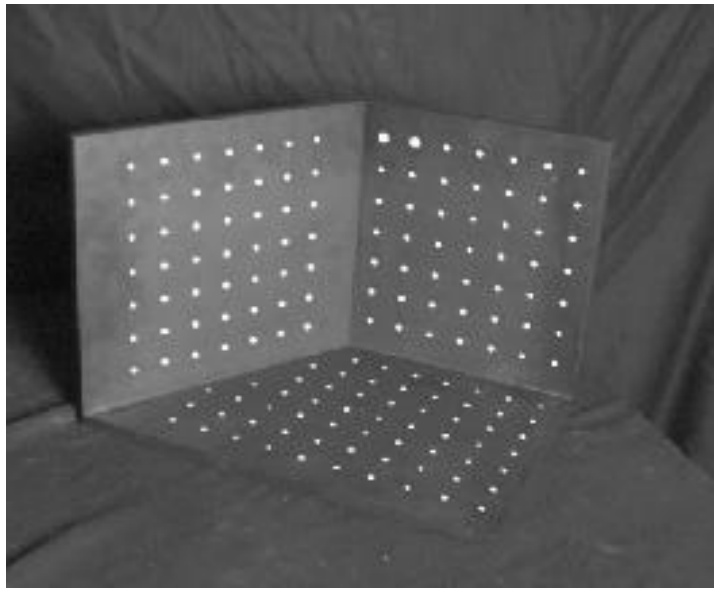
Why calibrate?

- Want to solve for 3D geometry
- Could solve for 3D shape without known cameras:
  - Structure-from-motion (unknown extrinsics)
  - Self-calibration (unknown intrinsics & extrinsics)
- So why bother pre-calibrating the camera?
  - Simplifies the 3D reconstruction problem – fewer parameters to solve for
  - Improves accuracy
  - Not too hard to do
  - Eliminates certain ambiguities (scale of scene)

Slide by Steve Seitz

Camera calibration

- Determine camera parameters from known 3D points or calibration object(s)
  - internal or intrinsic parameters such as focal length, optical center, aspect ratio: what kind of camera?
  - external or extrinsic (pose) parameters: where is the camera?
- How can we do this?



Slide by Richard Szeliski

Camera matrix

- Fold intrinsic calibration matrix **K** and extrinsic pose parameters (**R**,**t**) together into a camera matrix
  - M = K [R | t]

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

- (put 1 in lower right-hand corner for 11 degrees of freedom)

Slide by Richard Szeliski

Camera matrix calibration

- Directly estimate 11 unknowns in the **M** matrix using known 3D points (X_i, Y_i, Z_i) and measured feature positions (u_i,v_i)

$$u_i = \frac{m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1}$$

$$v_i = \frac{m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1}$$

Slide by Richard Szeliski

Camera matrix calibration

- Linear regression:
  - Bring denominator over, solve set of (over-determined) linear equations.

    How?

$$u_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1) = m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}$$

$$v_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + 1) = m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}$$

  - Least squares (pseudo-inverse)
  - Is this good enough?

Slide by Richard Szeliski

Camera matrix calibration

- Advantages:
  - very simple to formulate and solve
  - can recover $\mathbf{K}\,[\mathbf{R} \mid \mathbf{t}]$ from $\mathbf{M}$ using QR decomposition [Golub & VanLoan 96]
- Disadvantages:
  - doesn't compute internal parameters
  - more unknowns than true degrees of freedom
  - need a separate camera matrix for each new view

Slide by Richard Szeliski

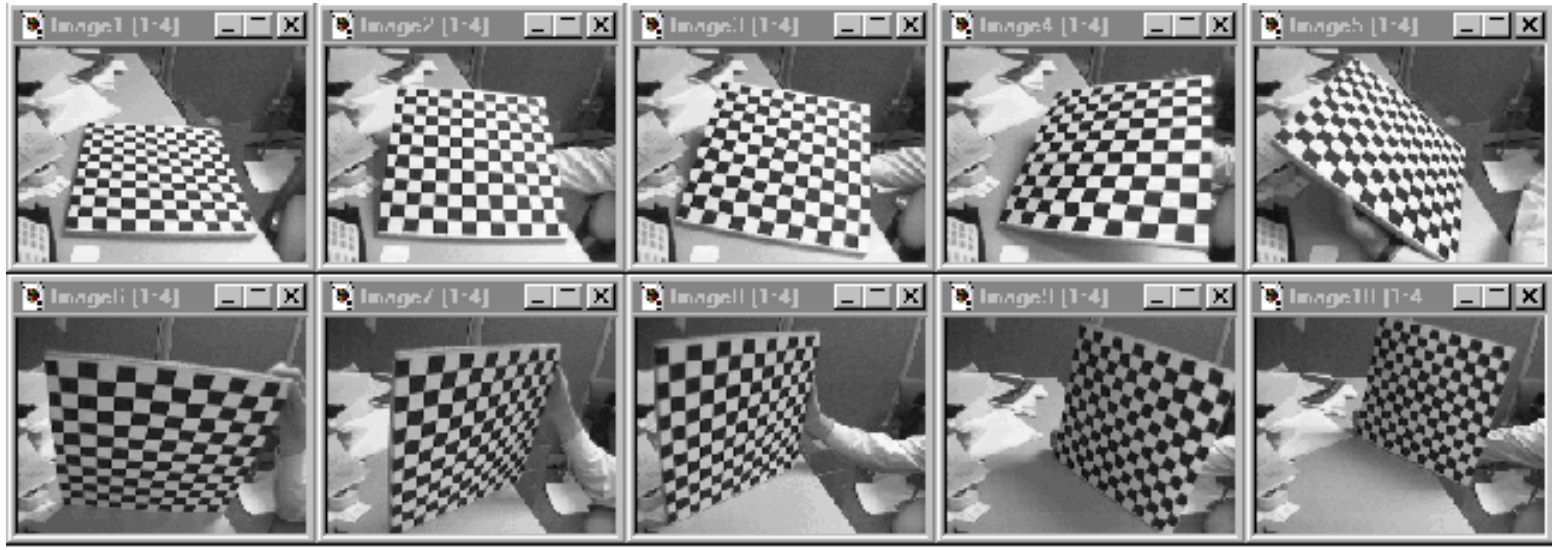
Separate intrinsic and extrinsic calibration

- Advantages:
  - can solve for more than one camera pose at a time
  - potentially fewer degrees of freedom

- Disadvantages:
  - more complex update rules
  - need a good initialization (recover $\mathbf{K}\,[\mathbf{R} \mid \mathbf{t}]$ from $\mathbf{M}$)

Slide by Richard Szeliski

Multi-plane calibration



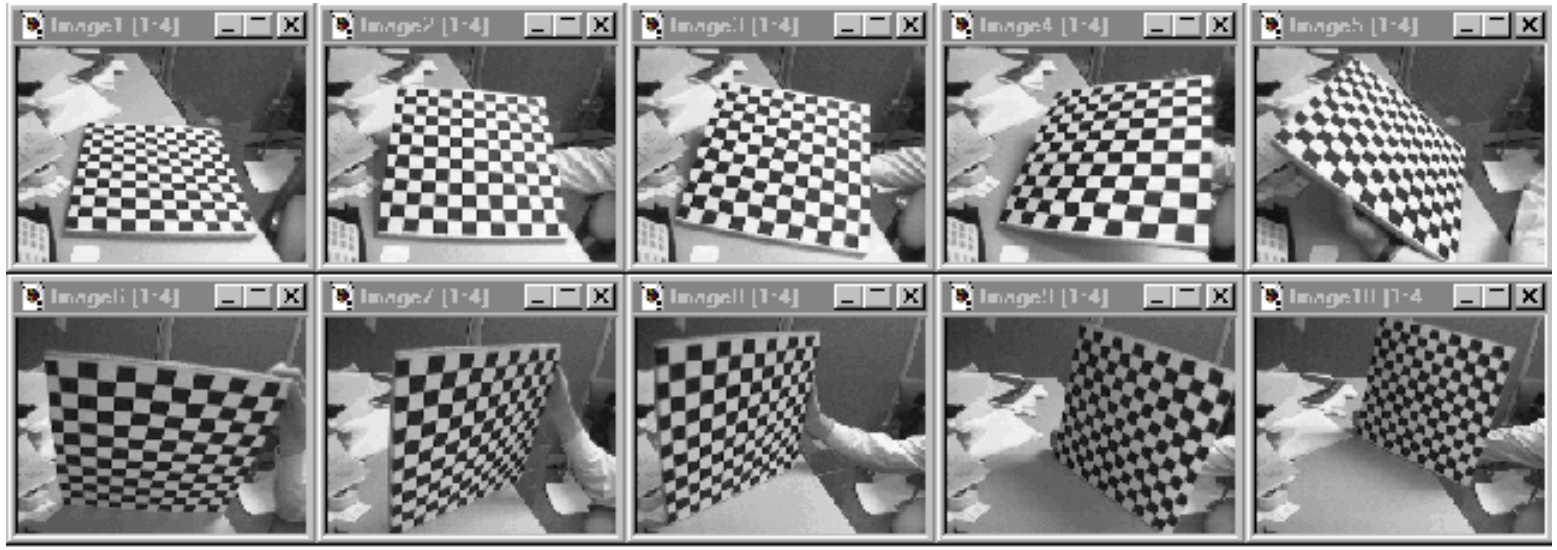
Images courtesy Jean-Yves Bouguet, Intel Corp.

Advantage

Disadvantages?

- Only requires a plane
- Don't have to know positions/orientations
- Good code available online!
  - OpenCV library: http://www.opencv.org/
  - MATLAB toolbox by Jean-Yves Bouget: http://www.vision.caltech.edu/bouguetj/calib_doc/index.html

Multi-plane calibration



Images courtesy Jean-Yves Bouguet, Intel Corp.

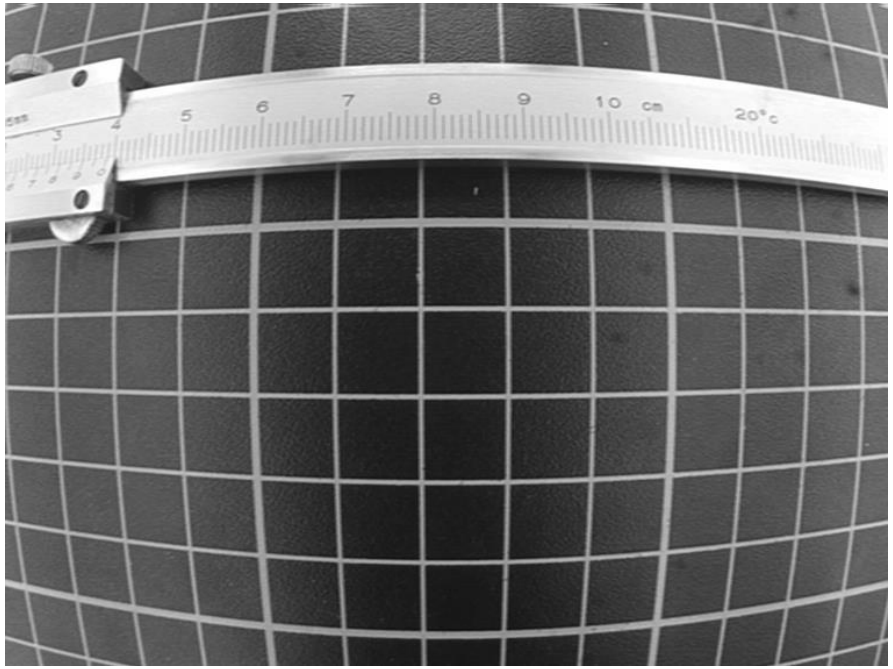
Need 3D → 2D correspondence

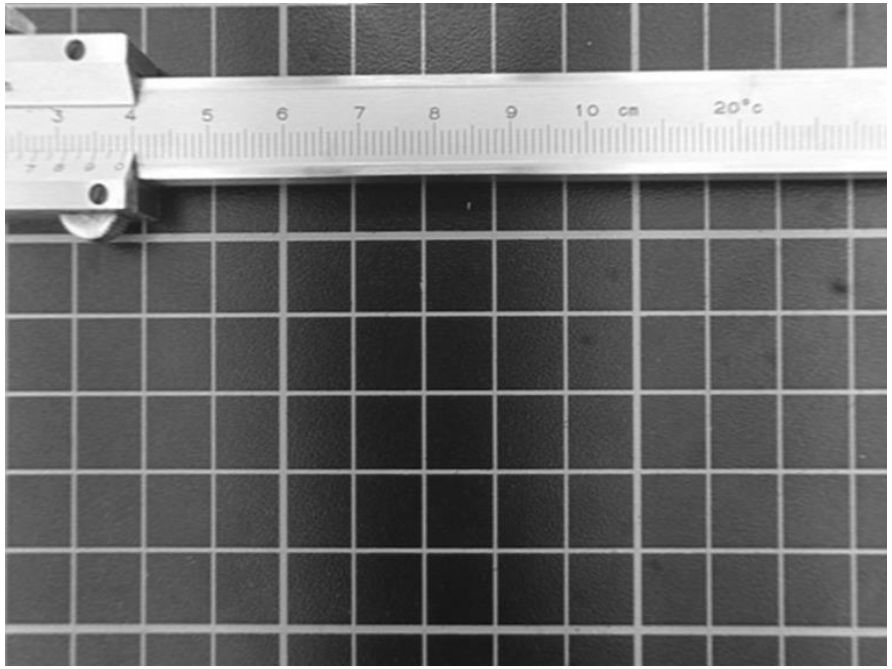
- User provided (lots of clicking)
- User seeded (some clicking)
- Fully automatic?

Lens distortion

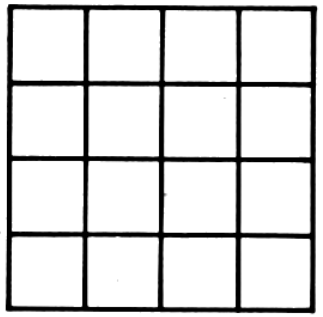
- radial & tangential distortion of the image
- caused by imperfect (read: cheap) lenses
- most noticeable near the edge of the lens

Making straight lines straight

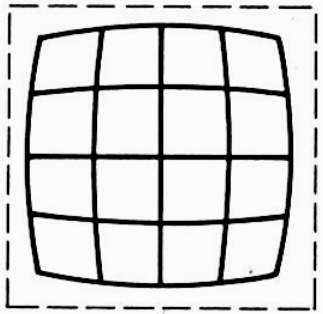




Barrel distortion



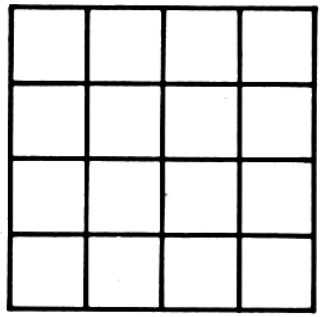
No distortion



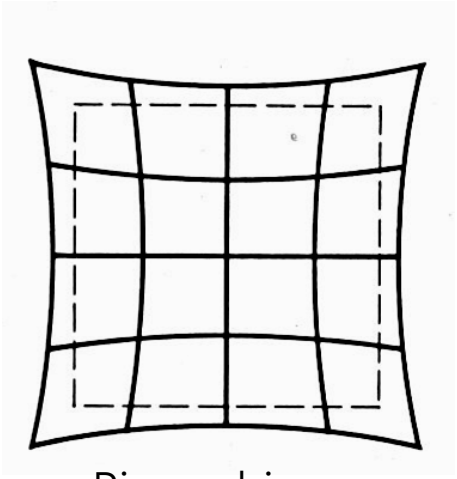
Barrel

Wide Angle Lens

Pin-cushion distortion



No distortion



Pin cushion

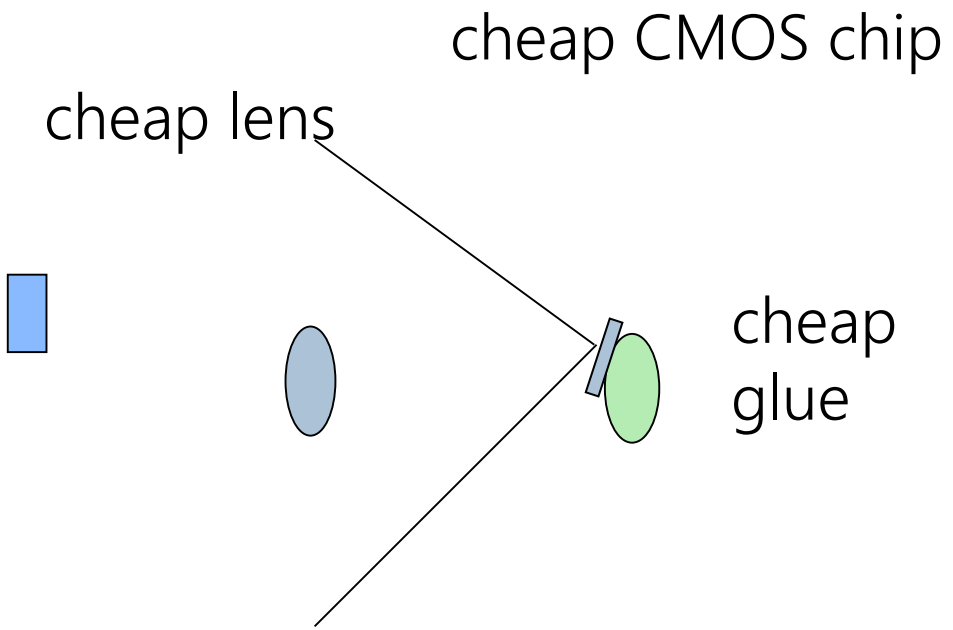
Telephoto lens

Radial distortion – k_1, k_2, k_5

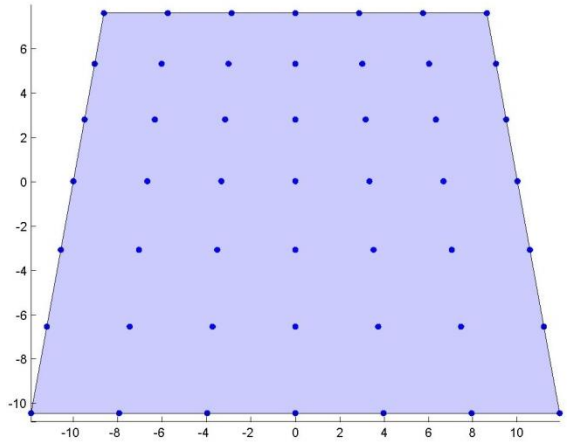
$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_d \\ y_d \end{pmatrix} \cdot \left(1 + k_1 \cdot r^2 + k_2 \cdot r^4 + k_5 \cdot r^6\right)$$

distance from center

Tangential distortion – k_3, k_4



image



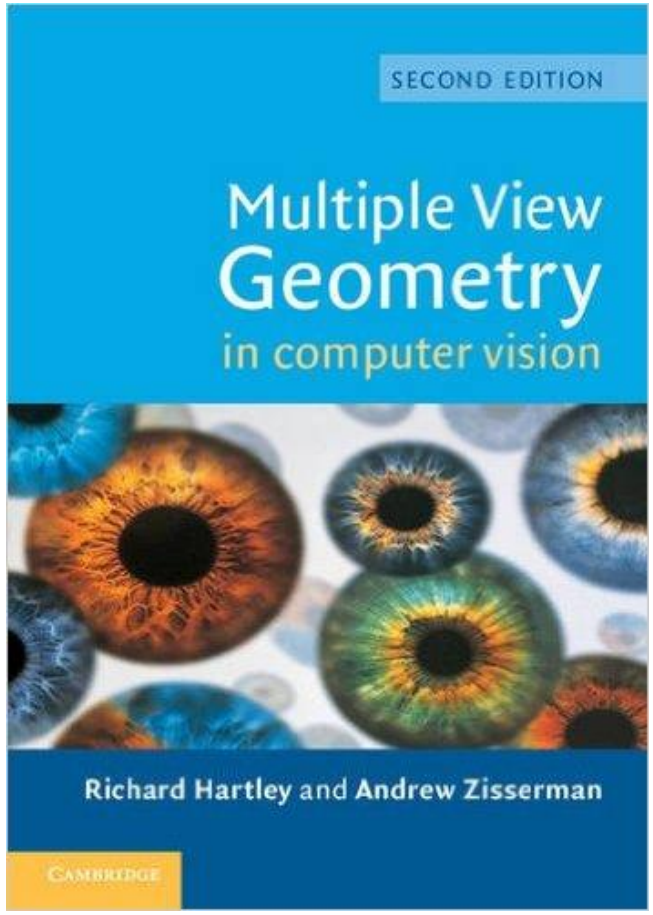
Recap of intrinsic calibration parameters

- Focal lengths f_x, f_y
- Principal point c_x, c_y
- Skew s

$$\mathbf{K} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

- Lens distortion k_1, k_2, k_3, k_4, k_5

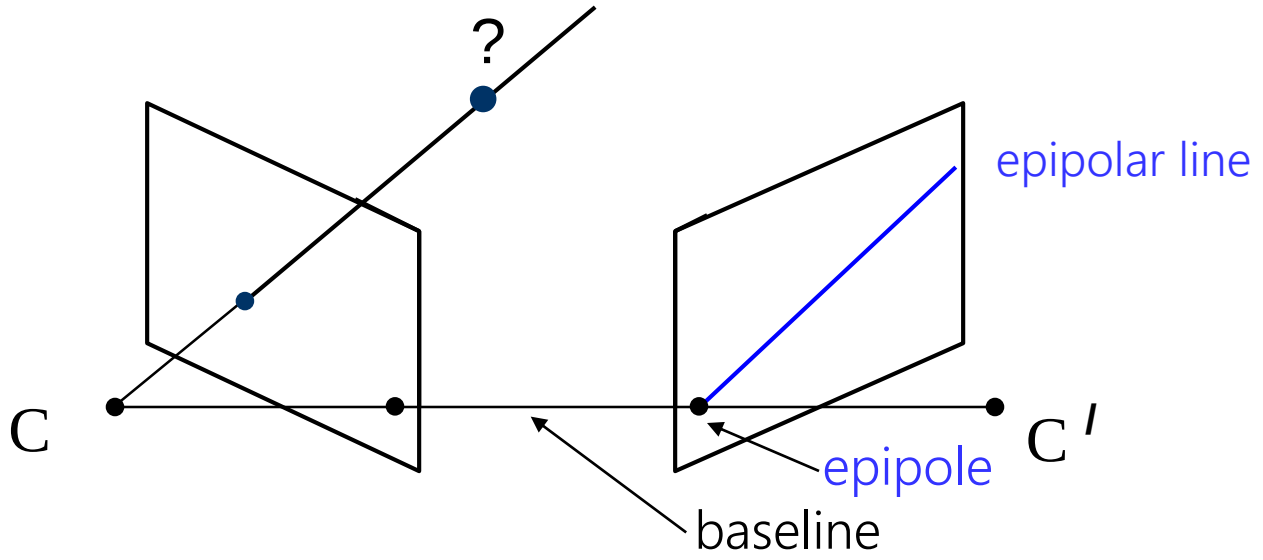
Epipolar geometry



Multiple View Geometry in Computer Vision
Hartley & Zisserman, 2004

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

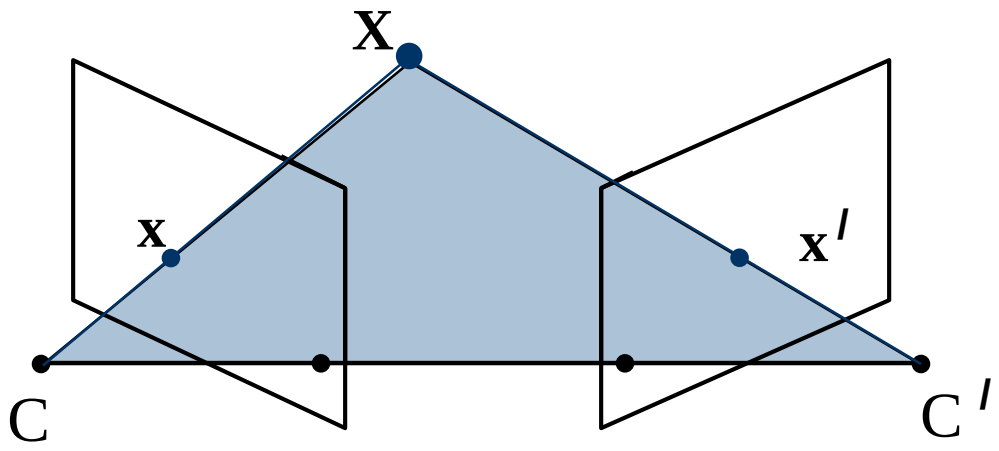
Epipolar constraint

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

Slide by Andrew Zisserman & Svetlana Lazebnik

Epipolar geometry continued

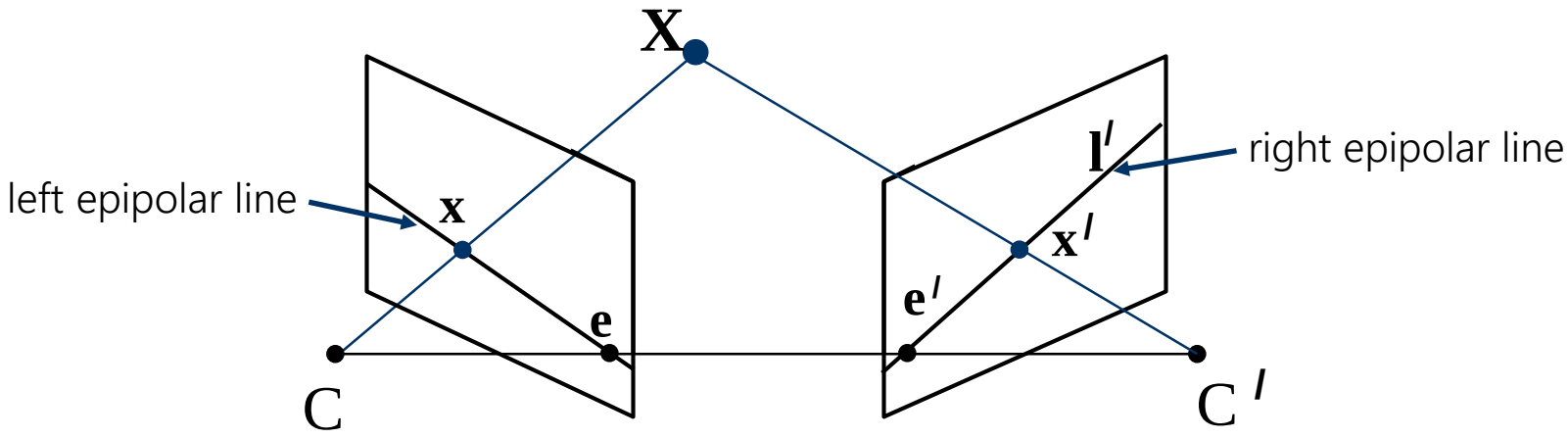
- Epipolar geometry is a consequence of the coplanarity of the camera centres and scene point



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

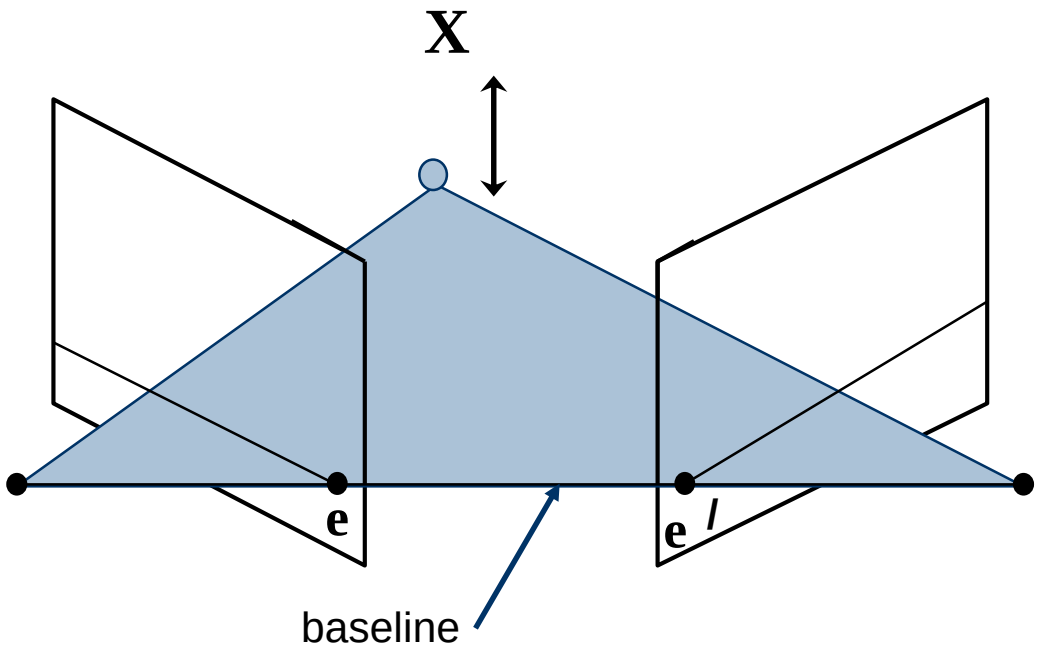
Nomenclature

- 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 other camera centre: $\mathbf{e}=\mathbf{P}\mathbf{C}'$, $e'=\mathbf{P}'\mathbf{C}$



Slide by Andrew Zisserman & Svetlana Lazebnik

The epipolar pencil

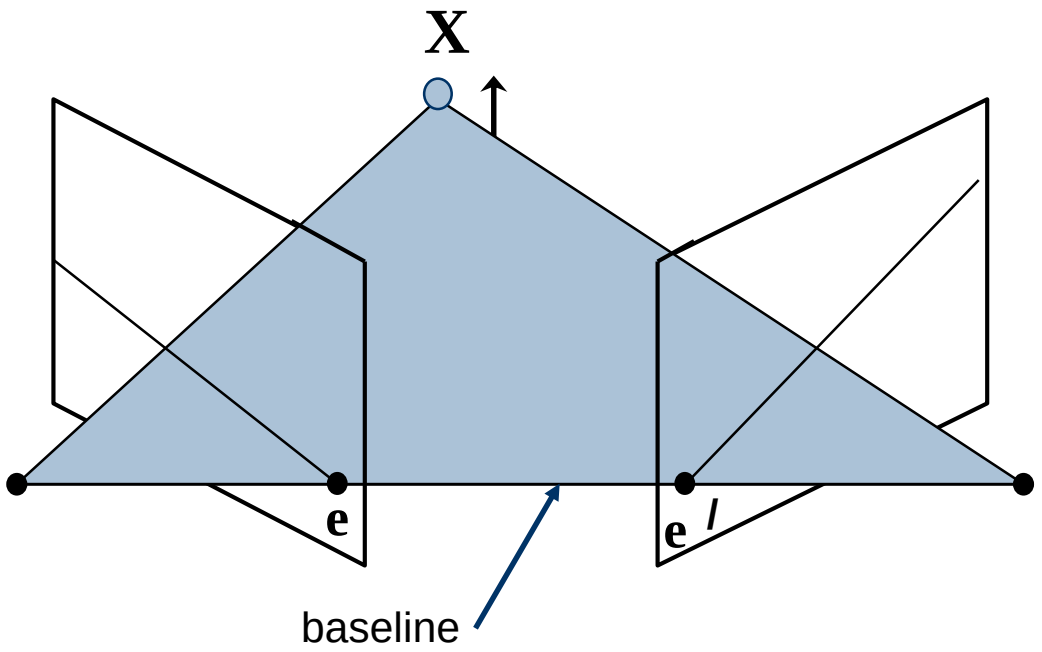


As the position of the 3D point **X** varies, the epipolar planes "rotate" about the baseline. This family of planes is known as an epipolar pencil. All epipolar lines intersect at the epipole.

(a pencil is a one parameter family)

Slide by Andrew Zisserman & Svetlana Lazebnik

The epipolar pencil

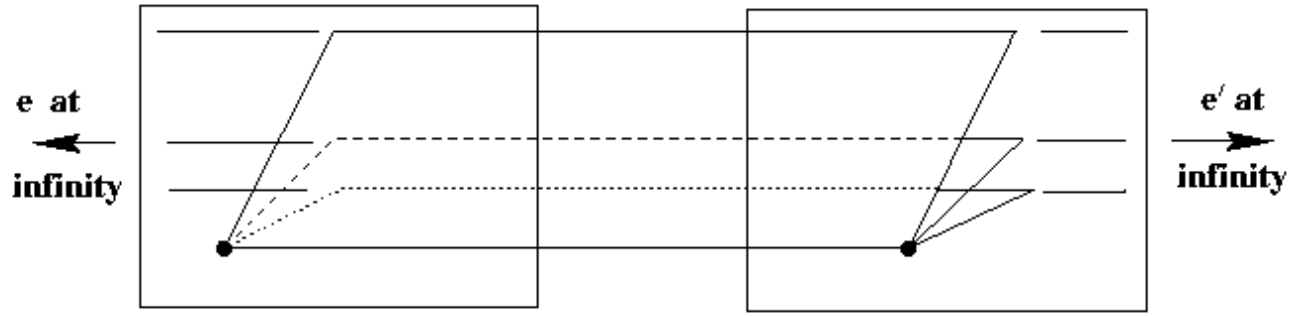


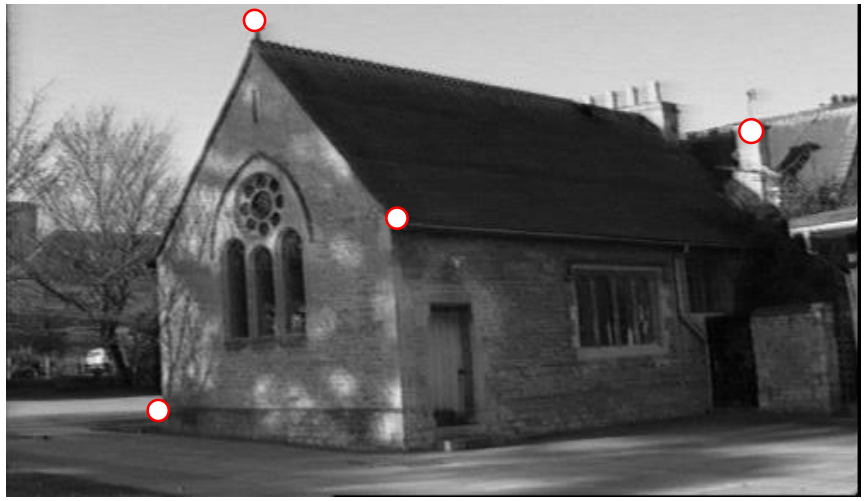
As the position of the 3D point **X** varies, the epipolar planes "rotate" about the baseline. This family of planes is known as an epipolar pencil. All epipolar lines intersect at the epipole.

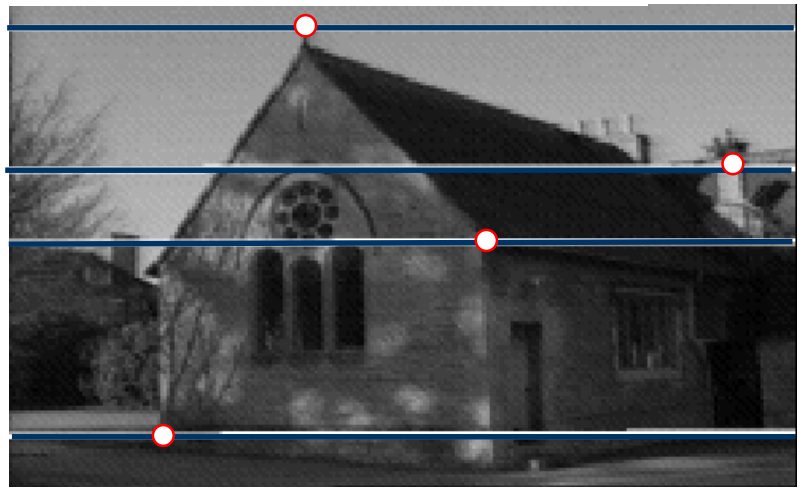
(a pencil is a one parameter family)

Slide by Andrew Zisserman & Svetlana Lazebnik

Epipolar geometry example I: parallel cameras



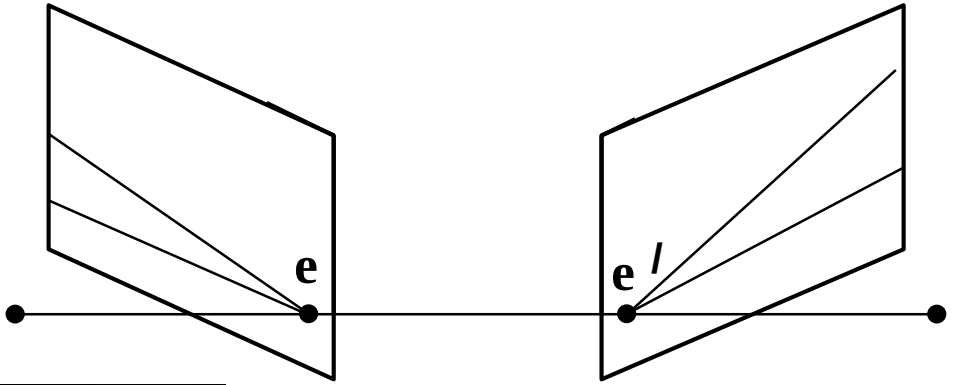


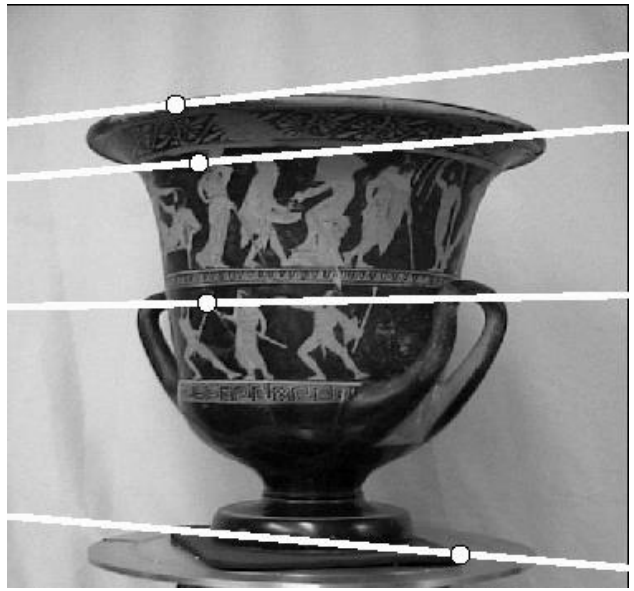


Epipolar geometry depends only on the relative pose (position and orientation) and internal parameters of the two cameras, i.e. the position of the camera centres and image planes. It does not depend on the scene structure (3D points external to the camera).

Slide by Andrew Zisserman & Svetlana Lazebnik

Epipolar geometry example II: converging cameras





Note: epipolar lines are in general not parallel

Slide by Andrew Zisserman & Svetlana Lazebnik

Homogeneous notation for lines

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.

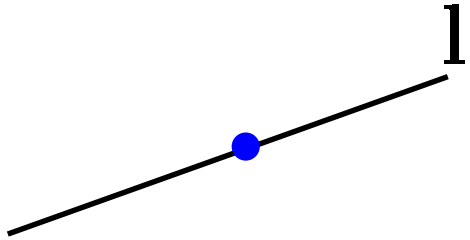
Write the line as $-y + 1 = 0$ then $l_1 = 0, l_2 = -1, l_3 = 1$, and $\mathbf{l} = (0, -1, 1)^\top$.

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).

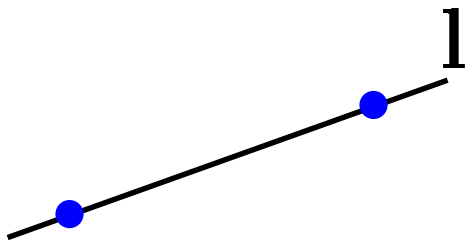
Writing both the point and line in homogeneous coordinates gives

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

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



- The line **l** through the two points **p** and **q** is **l = p x q**



Proof

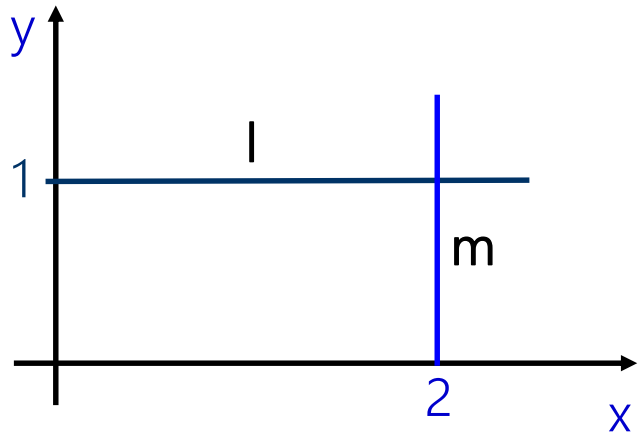
$$\mathbf{l}.\mathbf{p} = (\mathbf{p} \times \mathbf{q}).\mathbf{p} = 0 \qquad \mathbf{l}.\mathbf{q} = (\mathbf{p} \times \mathbf{q}).\mathbf{q} = 0$$

- The intersection of two lines **l** and **m** is the point **x = l x m**

Example: compute the point of intersection of the two lines **l** and **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 $\mathbf{v} \times \mathbf{v} = [\mathbf{v}]_\times \mathbf{v} = \mathbf{0}$.

Slide by Andrew Zisserman & Svetlana Lazebnik

Example

Compute the cross product of **l** and **m**

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

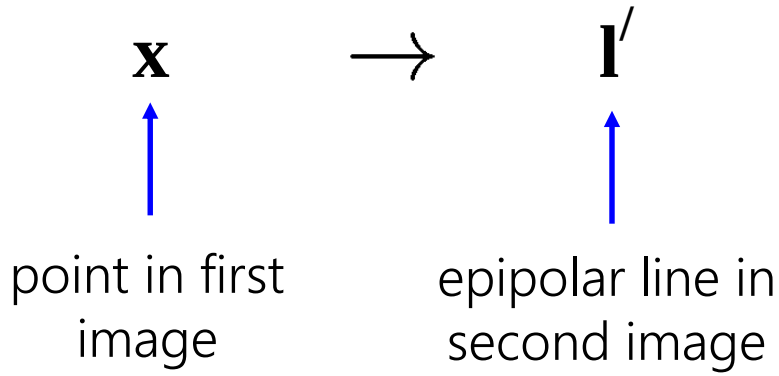
$$\mathbf{x} = \mathbf{l} \times \mathbf{m} = [\mathbf{l}]_\times \mathbf{m} = \begin{bmatrix} 0 & -1 & -1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{pmatrix} -1 \\ 0 \\ 2 \end{pmatrix} = \begin{pmatrix} -2 \\ -1 \\ -1 \end{pmatrix}$$

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



- 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

Slide by Andrew Zisserman & Svetlana Lazebnik

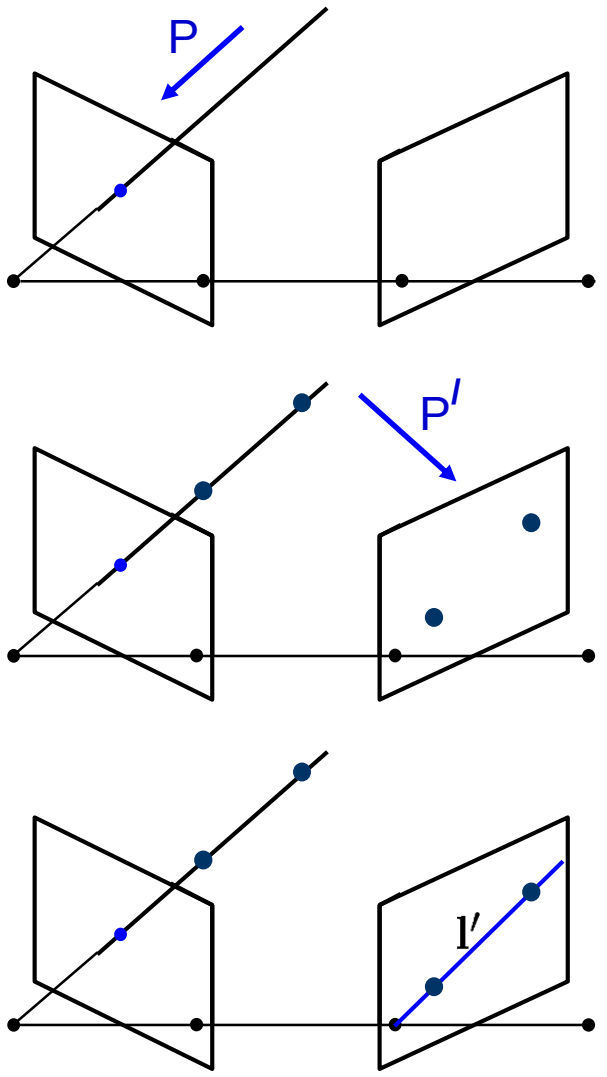
Derivation of the algebraic expression $\mathbf{l}' = \mathbf{Fx}$

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 $\mathbf{l}' = \mathbf{p} \times \mathbf{q}$



Slide by Andrew Zisserman & Svetlana Lazebnik

- choose camera matrices

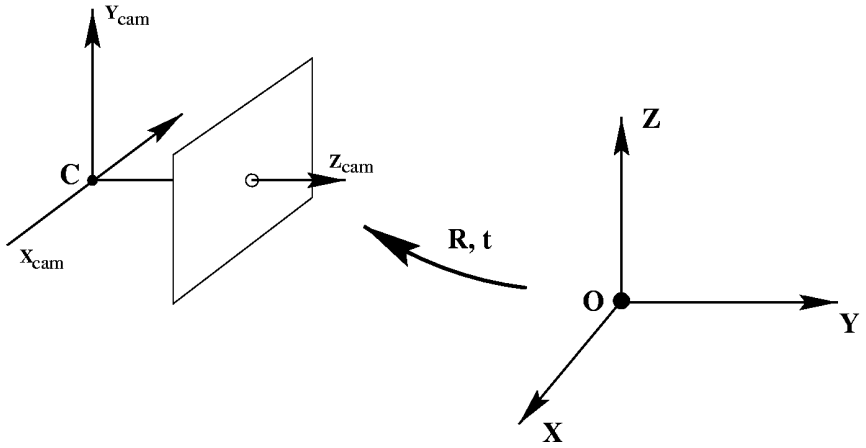
$$P = K[R \mid t]$$

internal calibration

rotation

translation

from world to camera coordinate frame



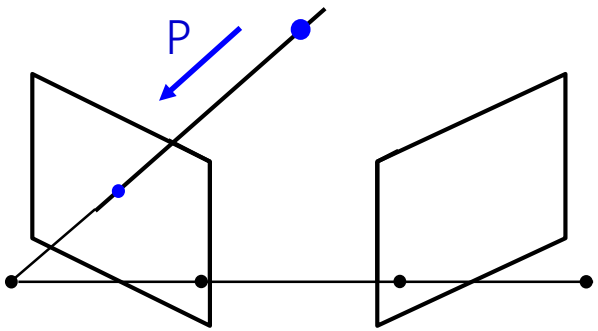
- first camera $P = K[I \mid 0]$

world coordinate frame aligned with first camera

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

Slide by Andrew Zisserman & Svetlana Lazebnik

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



A point **x** back projects to a ray

$$\begin{pmatrix} \mathsf{x} \\ \mathsf{y} \\ \mathsf{z} \end{pmatrix} = \mathsf{z}\mathbf{K}^{-1} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \mathsf{z}\mathbf{K}^{-1}\mathbf{x}$$

where **Z** is the point's depth, since

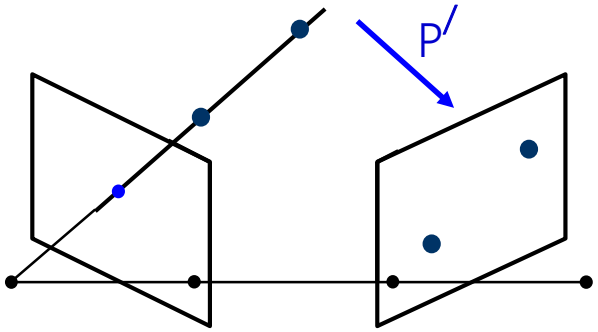
$$\mathbf{X}(\mathsf{z}) = \begin{pmatrix} \mathsf{z}\mathbf{K}^{-1}\mathbf{x} \\ 1 \end{pmatrix}$$

satisfies

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

Slide by Andrew Zisserman & Svetlana Lazebnik

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



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

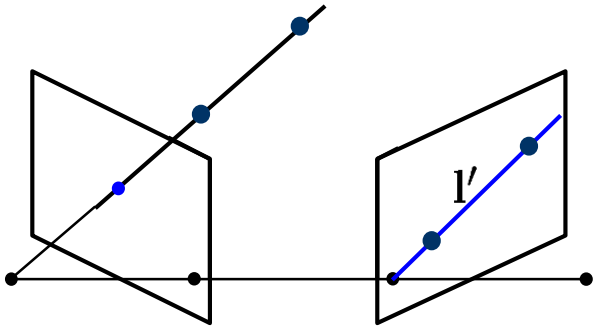
- Z = 0 is the camera centre $\begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix}$
- Z = ∞ is the point at infinity $\begin{pmatrix} K^{-1}\mathbf{x} \\ 0 \end{pmatrix}$

Project these two points into the second view

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

Slide by Andrew Zisserman & Svetlana Lazebnik

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



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 $(\mathbf{Ma}) \times (\mathbf{Mb}) = \mathtt{M}^{-\top}(\mathbf{a} \times \mathbf{b})$ where $\mathtt{M}^{-\top} = (\mathtt{M}^{-1})^{\top} = (\mathtt{M}^{\top})^{-1}$

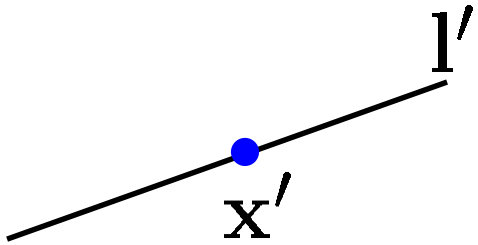
$$\mathbf{l}' = \mathtt{K}'^{-\top}\left(\mathbf{t} \times (\mathtt{R}\mathtt{K}^{-1}\mathbf{x})\right) = \underbrace{\mathtt{K}'^{-\top}[\mathbf{t}]_{\times}\mathtt{R}\mathtt{K}^{-1}}_{\mathtt{F}}\mathbf{x}$$

F is the fundamental matrix

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

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

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

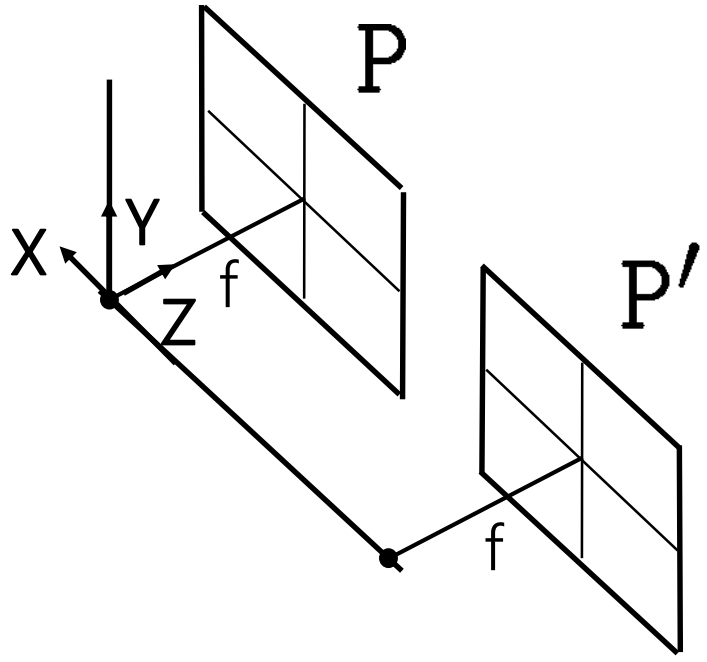


Slide by Andrew Zisserman & Svetlana Lazebnik

Example 1: compute F for parallel cameras

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

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



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

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

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

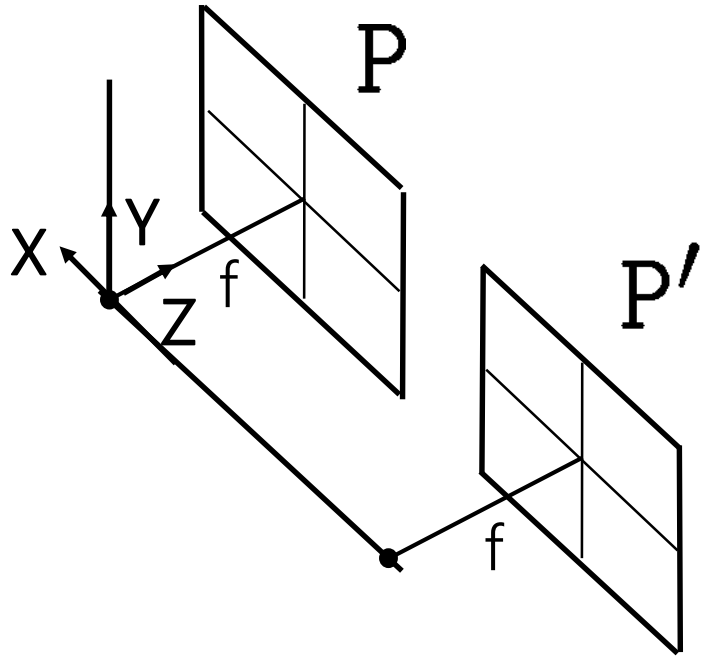
reduces to $y = y'$, i.e. raster correspondence (horizontal scan-lines)

Slide by Andrew Zisserman & Svetlana Lazebnik

Example 1: compute F for parallel cameras

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

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



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

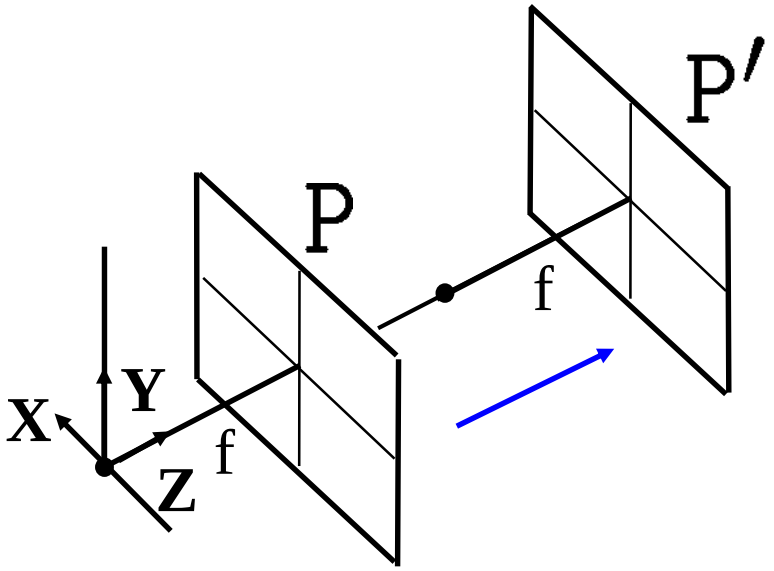
$$\mathbf{e} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

Geometric interpretation?

Example 2: compute F for forward-translating camera

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

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



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

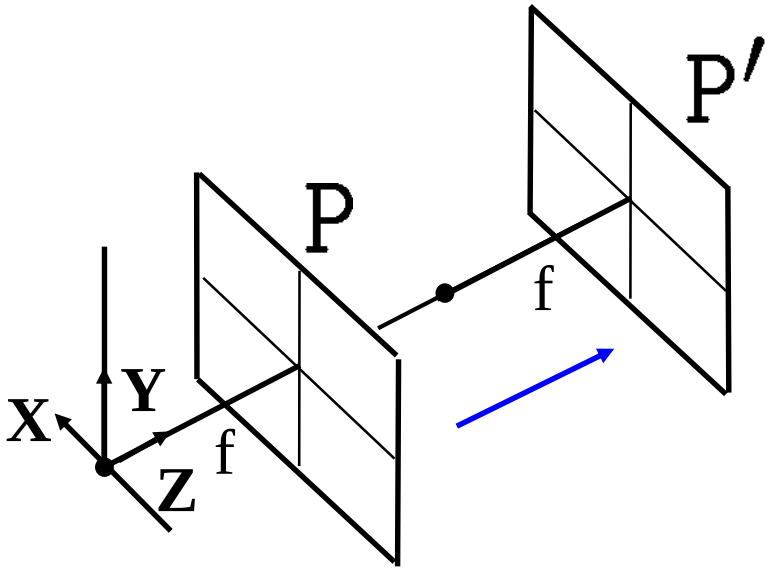
Slide by Andrew Zisserman & Svetlana Lazebnik

Example 2: compute F for forward-translating camera

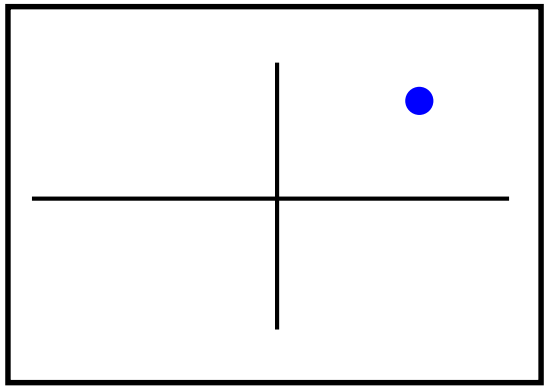
From $\mathbf{l}' = \mathbf{F}\mathbf{x}$ the epipolar line for the point $\mathbf{x} = (x, y, 1)^\top$ is

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

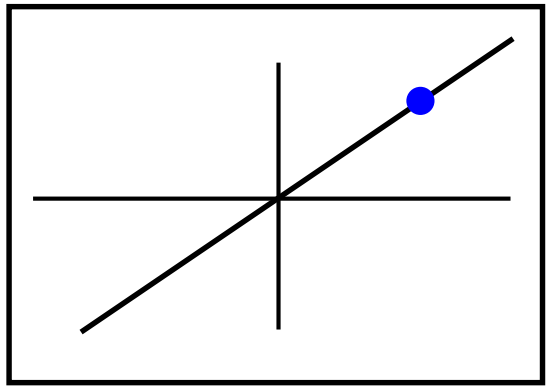
The points $(x, y, 1)^\top$ and $(0, 0, 1)^\top$ lie on this line



first image



second image



Slide by Andrew Zisserman & Svetlana Lazebnik

Slide by Andrew Zisserman & Svetlana Lazebnik

Slide by Andrew Zisserman & Svetlana Lazebnik

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 $\mathtt{P}, \mathtt{P}'$:

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

Slide by Andrew Zisserman & Svetlana Lazebnik

Fundamental matrix estimation

Automatic computation of F

(i) Interest points

(ii) Putative correspondences

(iii) RANSAC

(iv) Non-linear re-estimation of F

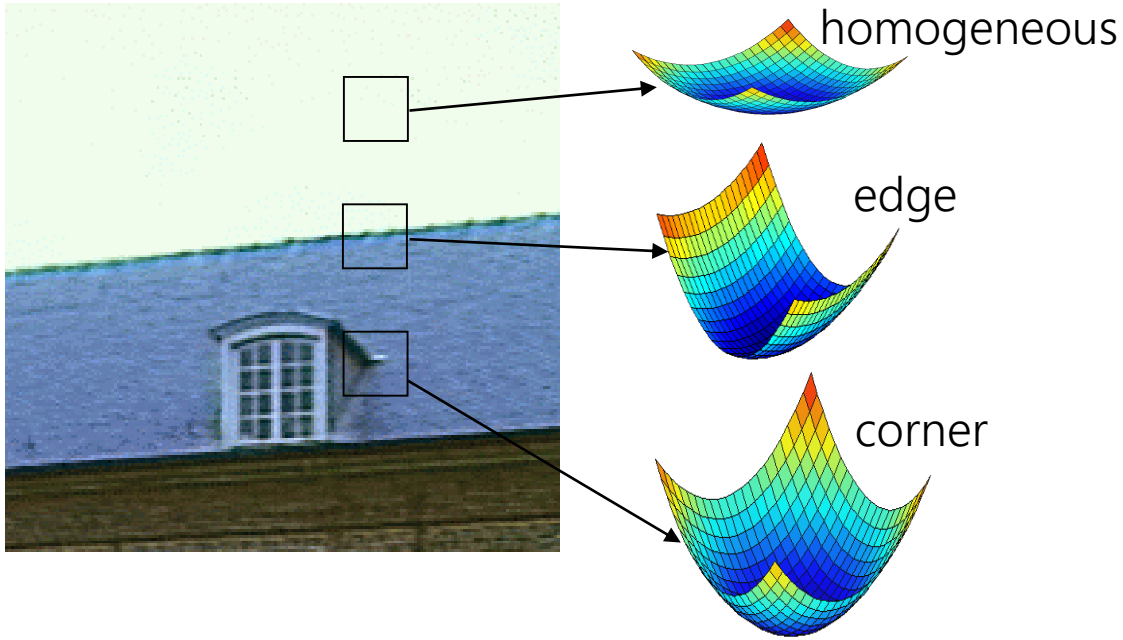
(v) Guided matching

(repeat (iv) and (v) until stable)

Slide by Marc Pollefeys

Feature points

- Find points that differ as much as possible from neighbouring points, for example:
  - Harris corners
  - SIFT
  - SURF
  - etc.



Slide by Marc Pollefeys

RANSAC

Step 1. Extract features

Step 2. Compute a set of potential matches

Step 3. do

Step 3.1 select minimal sample (i.e. 7 matches)

Step 3.2 compute solution(s) for F

} (generate hypothesis)

Step 3.3 determine inliers (verify hypothesis)

until $\Gamma(\#inliers, \#samples) < 95\%$

$$\Gamma = 1 - (1 - \left(\frac{\#inliers}{\#matches}\right)^7)^{\#samples}$$

Step 4. Compute F based on all inliers

Step 5. Look for additional matches

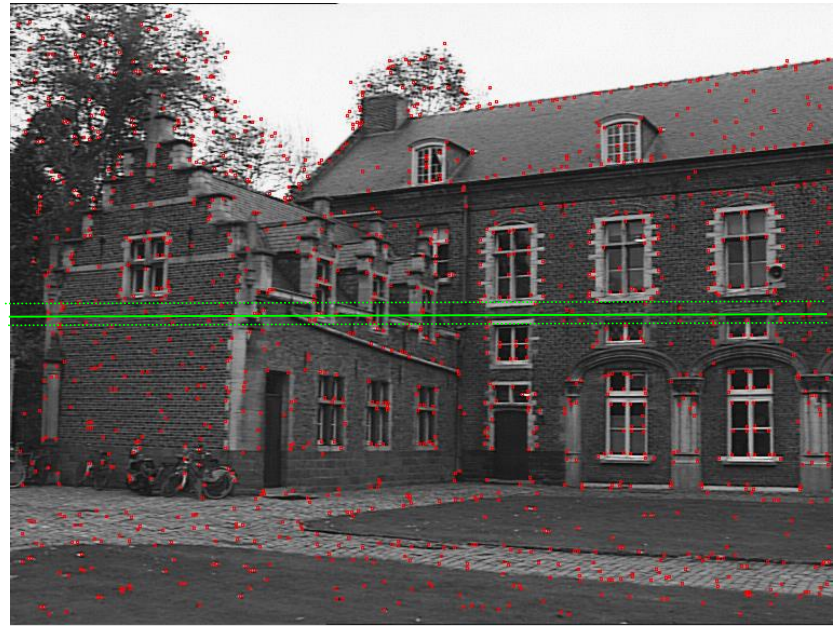
Step 6. Refine F based on all correct matches

#inliers	90%	80%	70%	60%	50%
#samples	5	13	35	106	382

Slide by Marc Pollefeys

Finding more matches

- Restrict search range to neighbourhood of epipolar line
  - ±1.5 pixels
- relax disparity restriction (along epipolar line)



Slide by Marc Pollefeys

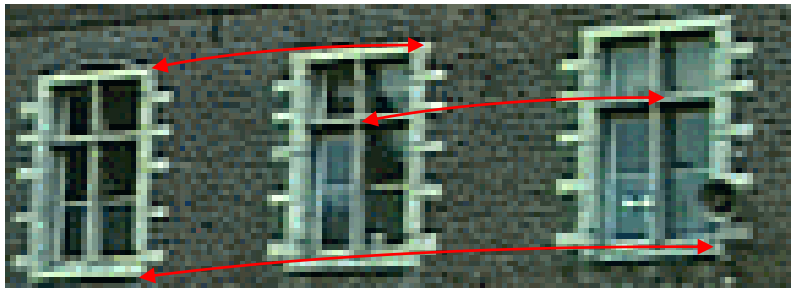
Degenerate cases

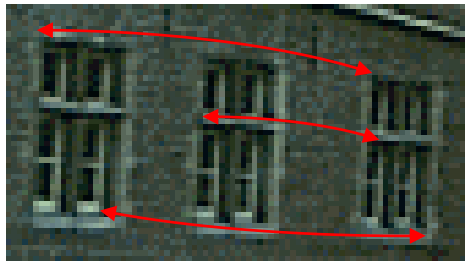
- Planar scene
- Pure rotation
- No unique solution
  - Remaining DOF filled by noise
  - Use simpler model (e.g. homography)
- Model selection (Torr et al., ICCV´98, Kanatani, Akaike)
  - Compare **H** and **F** according to expected residual error (compensate for model complexity)

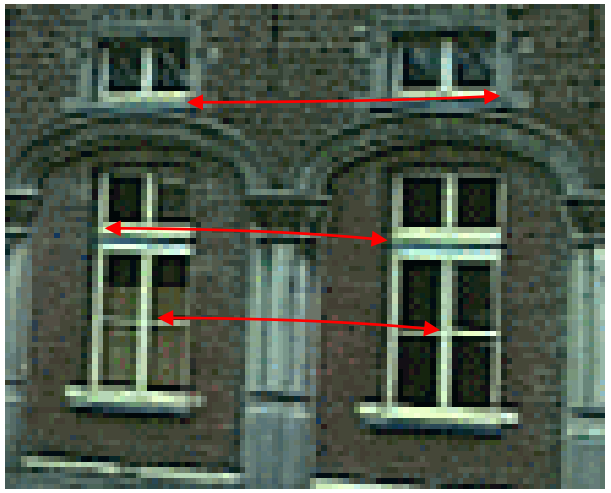
Slide by Marc Pollefeys

More problems

- Absence of sufficient features (i.e. no texture)
- Repeated structure ambiguity
- Robust matcher also finds support for wrong hypothesis
  - solution: detect repetition (Schaffalitzky and Zisserman, BMVC'98)



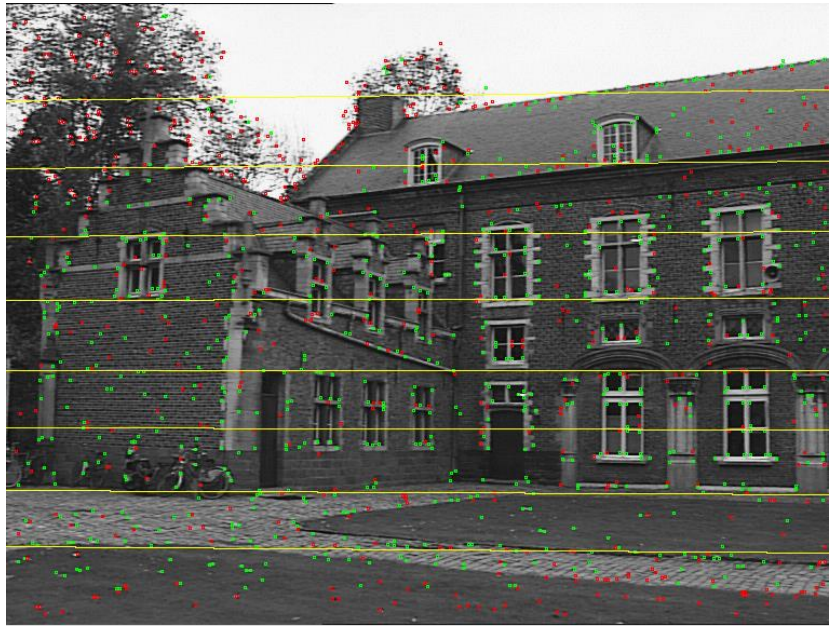




Slide by Marc Pollefeys

Two-view geometry

geometric relations between two views is fully described by recovered 3×3 matrix **F**



Slide by Marc Pollefeys

Triangulation

Triangulation

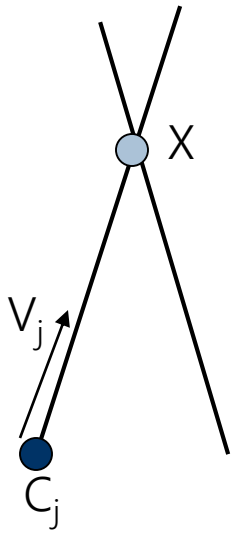
- **Problem:**

  Given some points $\{(u_j, v_j)\}$ in correspondence across two or more images (taken from calibrated cameras), compute the 3D location $\mathbf{X}$

Slide by Steve Seitz

Method I: intersecting viewing rays in 3D

- Minimise $\arg\min_{\mathbf{X}} \sum_j \|\mathbf{C}_j + s\mathbf{V}_j - \mathbf{X}\|$
  - $\mathbf{X}$ is the unknown 3D point
  - $\mathbf{C}_j$ is the optical center of camera j
  - $\mathbf{V}_j$ is the *viewing ray* for pixel (u_j, v_j)
  - s_j is unknown distance along $\mathbf{V}_j$
- Advantage: geometrically intuitive



Slide by Steve Seitz

Method II: linear triangulation (algebraic solution)

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

For the first camera:

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

where $\mathbf{p}^{i\mathsf{T}}$ are the rows of P

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

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

- rearrange as (first two equations only)

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

Method II: linear triangulation (algebraic solution)

Similarly for the second camera:

$$\begin{bmatrix} x'\mathbf{p}'^{3\mathsf{T}} - \mathbf{p}'^{1\mathsf{T}} \\ y'\mathbf{p}'^{3\mathsf{T}} - \mathbf{p}'^{2\mathsf{T}} \end{bmatrix} \mathbf{X} = \mathbf{0}$$

Collecting together gives

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

where $\mathtt{A}$ is the 4×4 matrix

$$\mathtt{A} = \begin{bmatrix} x\mathbf{p}^{3\mathsf{T}} - \mathbf{p}^{1\mathsf{T}} \\ y\mathbf{p}^{3\mathsf{T}} - \mathbf{p}^{2\mathsf{T}} \\ x'\mathbf{p}'^{3\mathsf{T}} - \mathbf{p}'^{1\mathsf{T}} \\ y'\mathbf{p}'^{3\mathsf{T}} - \mathbf{p}'^{2\mathsf{T}} \end{bmatrix}$$

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

Problem: does not minimize anything meaningful

Advantage: extends to more than two views

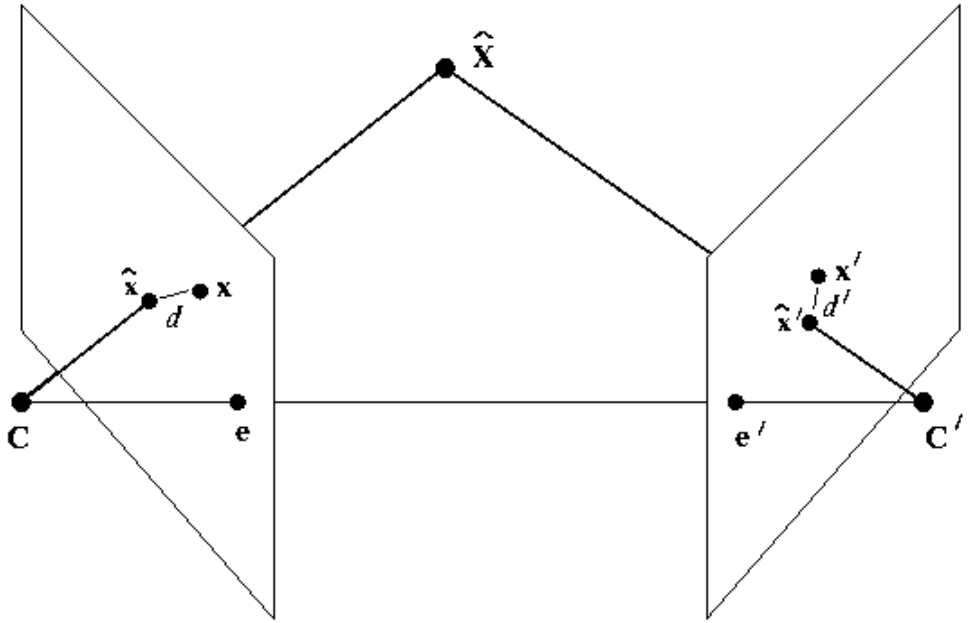
Slide by Andrew Zisserman & Svetlana Lazebnik

Method III: minimising a geometric/statistical error

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

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

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



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

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

Slide by Andrew Zisserman & Svetlana Lazebnik

Conclusion + Questions