Computer Vision: 8. Two views (a)

I-Chen Lin, Assistant Professor
Dept. of CS, National Chiao Tung University

Objective

- Epipolar geometry
- Depth reconstruction
- Binocular fusion

Textbook:

• David A. Forsyth and Jean Ponce, Computer Vision: A Modern Approach, Prentice Hall, New Jersey, 2003.

• R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision 2nd Ed., Cambridge University Press, 2004.

Plenty of slides are modified from the reference lecture notes or project pages:

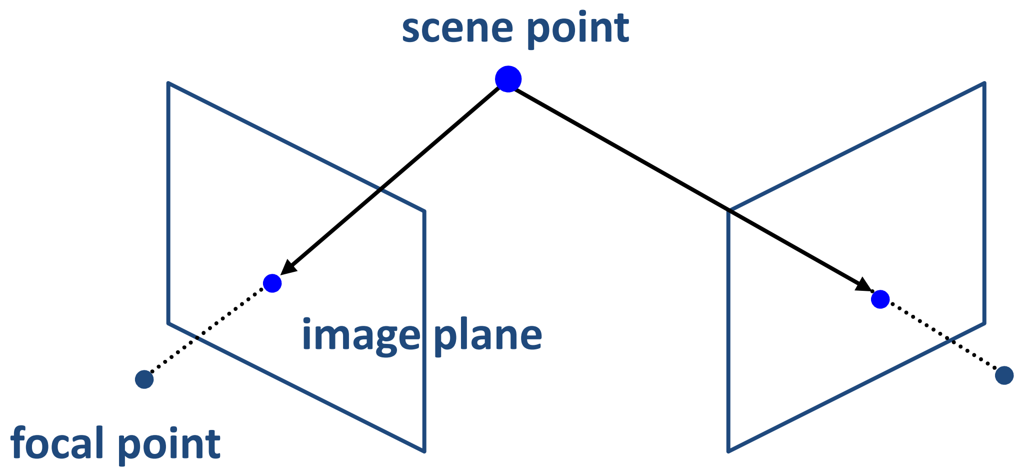
• Prof. J. Rehg, Computer Vision, Georgia Inst. of Tech.

• Prof. S. Seitz and P. Heckbert, Image-based modeling and rendering course notes, CMU.

• Dr. Ng Teck Khim, Computer Vision and Graphics for Special Effects lecture notes.

• Prof. D.A. Forsyth, Computer Vision, UIUC.

Two-view projective geometry



- How to relate point positions in different views?
  - Central question in stereo vision
  - Projective geometry gives us some powerful tools
    - constraints between two or more images
    - equations to transfer points from one image to another

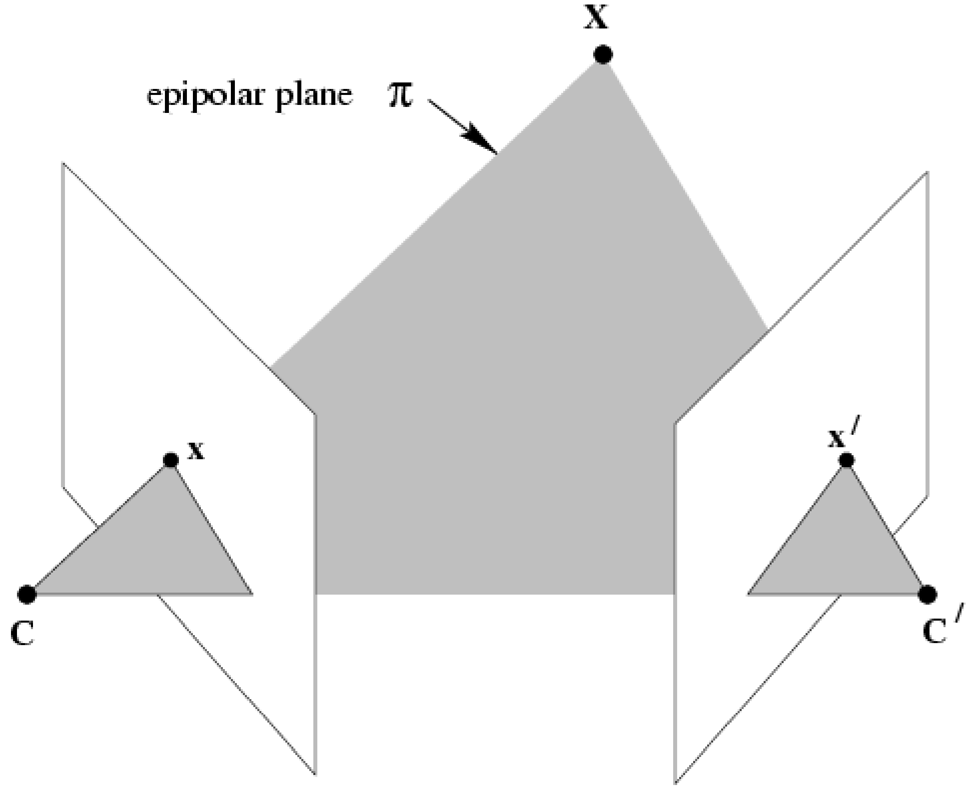
Two-view projective geometry

- Correspondence geometry:
  - Given an image point x in the first view, how does this constrain the position of the corresponding point x' in the second image?

- Camera geometry (motion):
  - Given a set of corresponding image points $\{x_i \leftrightarrow x'_i\}$, $i=1,\ldots,n$, what are the cameras C and C' for the two views? Or what is the geometric transformation between the views?

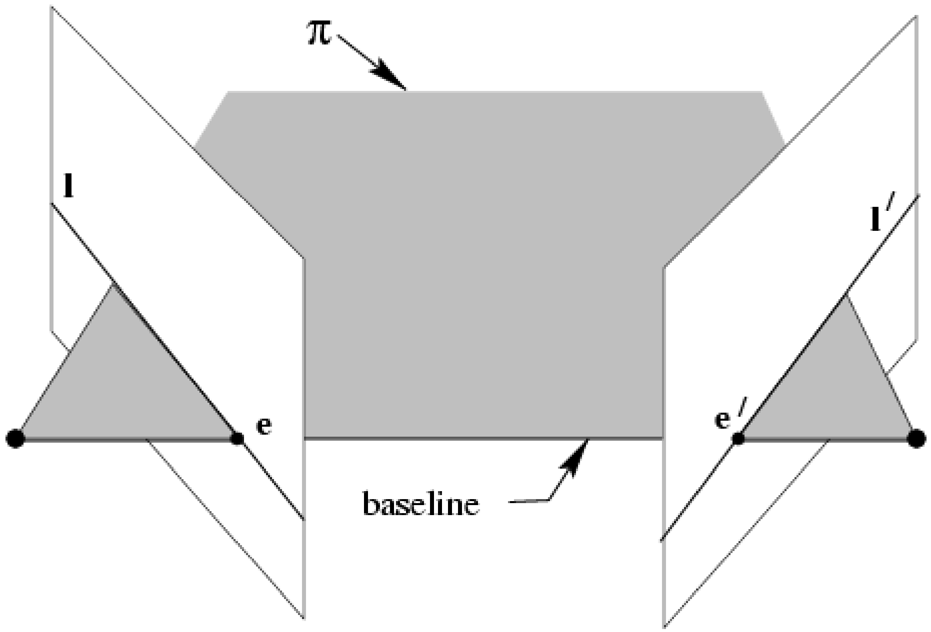
- Scene geometry (structure):
  - Given corresponding image points $x_i \leftrightarrow x'_i$ and cameras C, C', what is the position of the point X in space?

Epipolar geometry



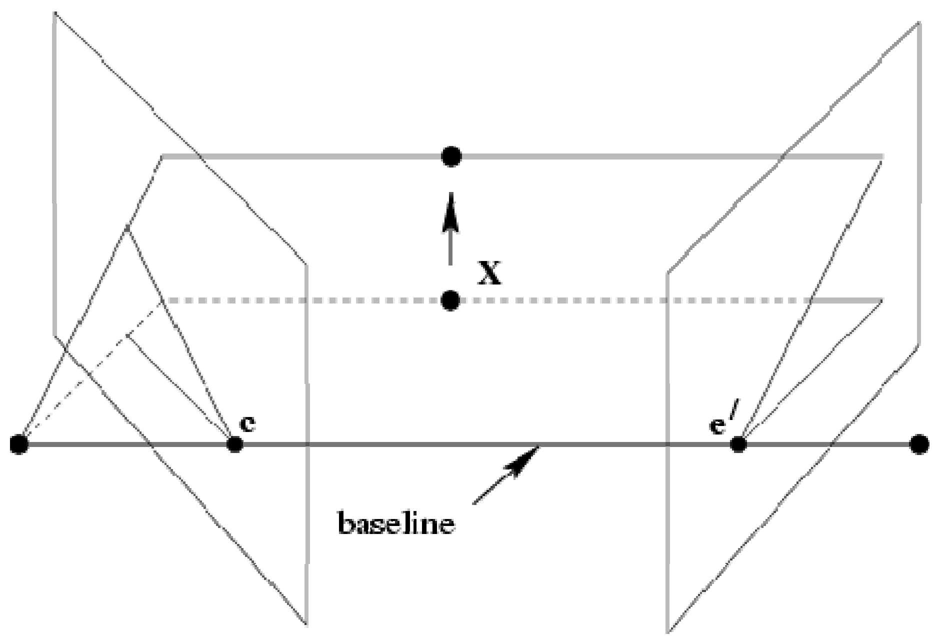
C,C’,x,x’ and X are coplanar

Epipolar geometry



All points on π project on l and l’

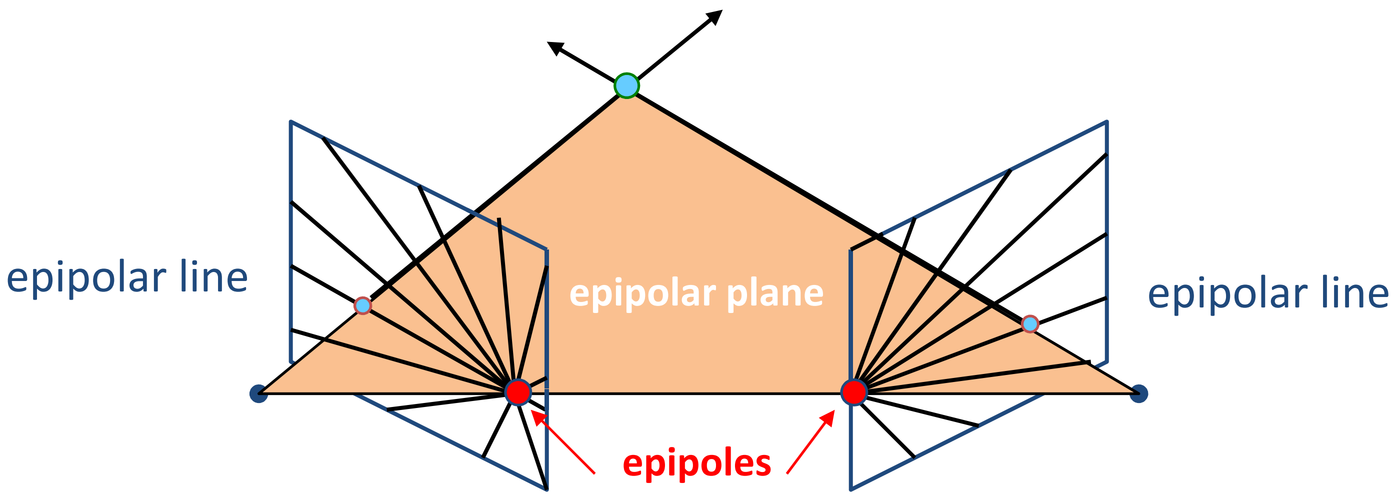
Epipolar geometry



Family of planes π and lines l and l'
Intersection in e and e'

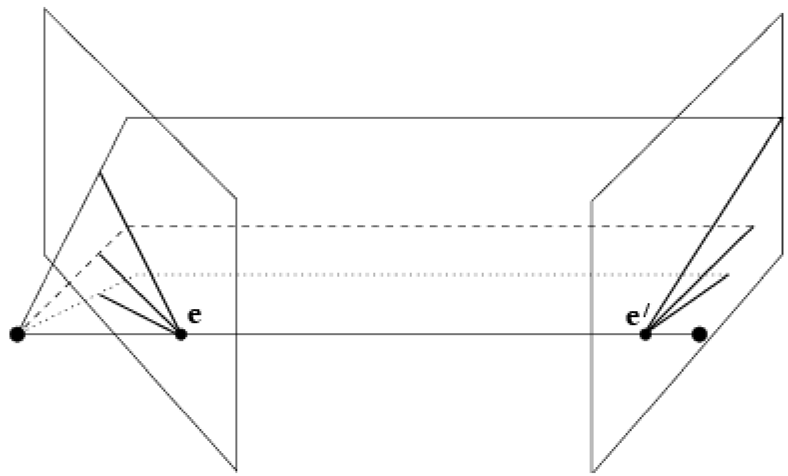
Epipolar geometry

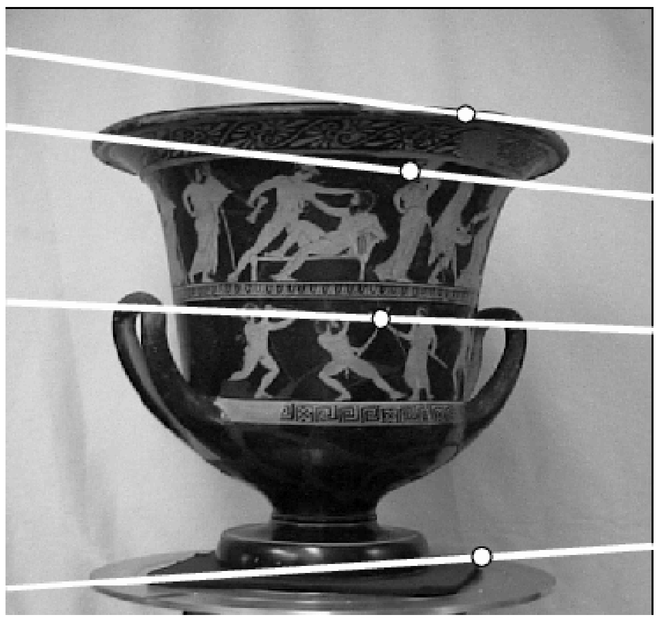
- What does one view tell us about another?
  – Point positions in 2nd view must lie along a known line

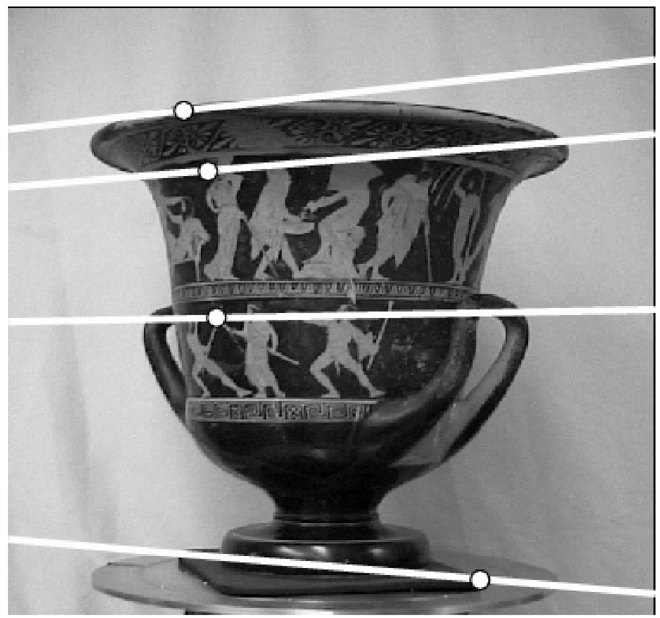


- Epipolar Constraint
  - Extremely useful for stereo matching
    - Reduces problem to 1D search along *conjugate epipolar lines*
  - Also useful for view interpolation...

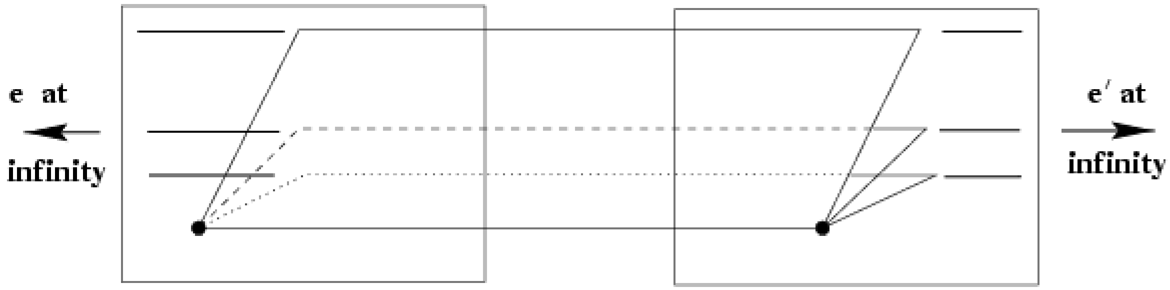
Example: converging cameras



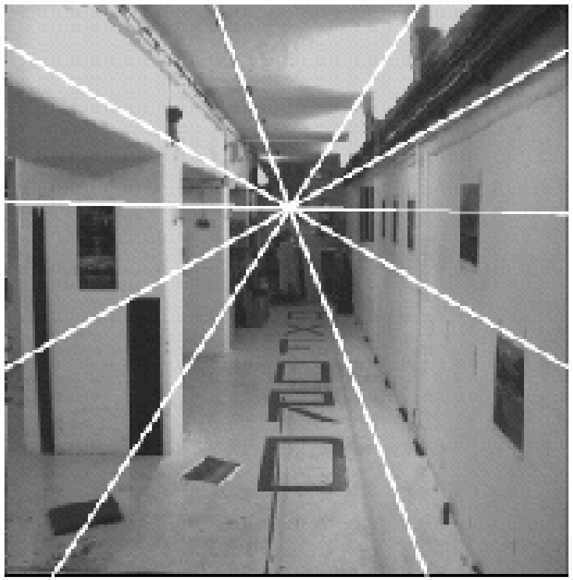


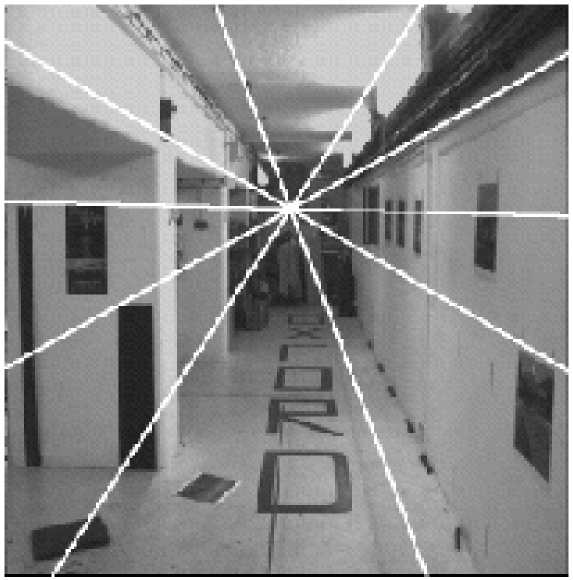


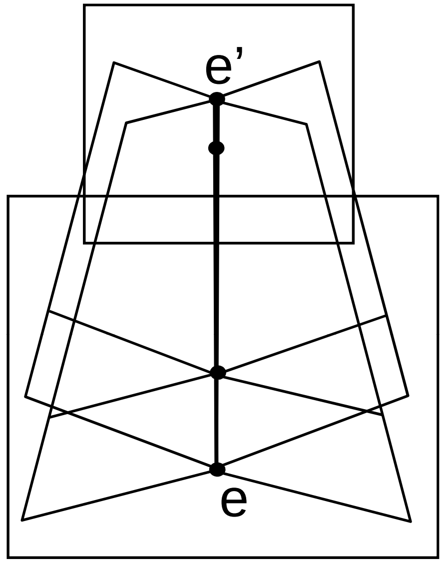
Example: motion parallel with image plane



Example: forward motion







3D to 2D: perspective projection

- Matrix Projection:

$$\mathbf{p} = \begin{bmatrix} sx \\ sy \\ s \end{bmatrix} = \begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathbf{\Pi P}$$

$\prod$ can be decomposed into $\mathbf{T} \rightarrow \mathbf{R} \rightarrow \text{project} \rightarrow \mathbf{A}$

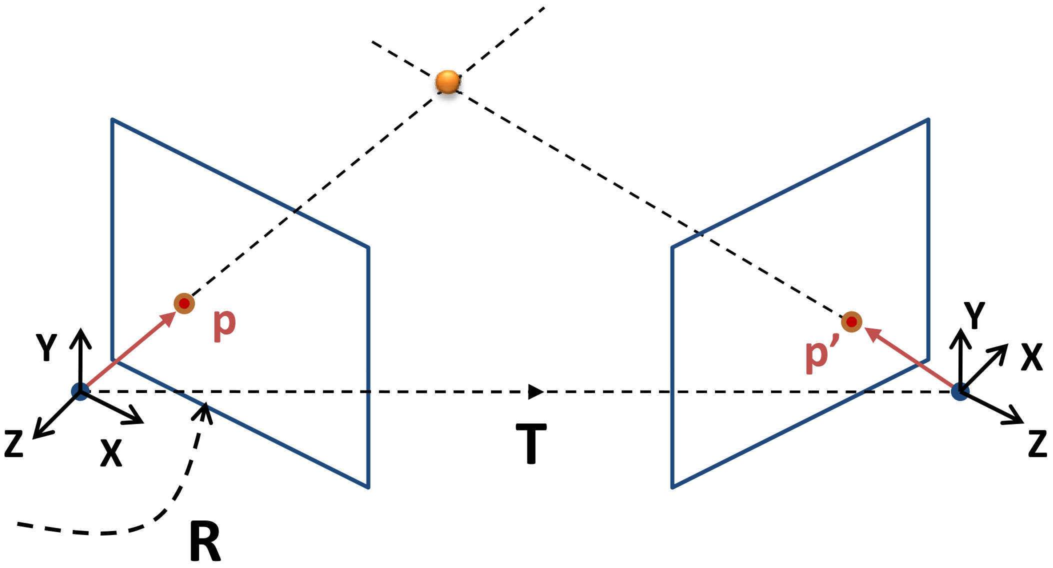
$$\mathbf{\Pi} = \begin{bmatrix} \alpha & -\alpha \cot\theta & u_0 \\ 0 & \dfrac{\beta}{\sin\theta} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{3x3} & \mathbf{0}_{3x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I}_{3x3} & \mathbf{T}_{3x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix}$$

Then we can write the projection as:

$$\mathbf{p} = \mathbf{\Pi P} = K\mathbf{R}(\mathbf{P} + \mathbf{T})$$

Epipolar algebra

- How do we compute epipolar lines?
  - Can trace out lines, reproject. But that is overkill



$$\mathbf{p'} = \mathbf{R}\mathbf{p} + \mathbf{T}$$

- Note that **p'** is $\perp$ to **T**$\times$**p'**
  - So 0 = $\mathbf{p'}^T \mathbf{T}\times\mathbf{p'} = \mathbf{p'}^T \mathbf{T}\times(\mathbf{R}\mathbf{p} + \mathbf{T}) = \mathbf{p'}^T \mathbf{T}\times(\mathbf{R}\mathbf{p})$

Simplifying: $\mathbf{p'}^T \, \mathbf{T} \times (\mathbf{Rp}) = 0$

- We can write a cross-product $a \times b$ as a matrix equation
  - $\mathbf{a} \times \mathbf{b} = \mathbf{A}_\times \mathbf{b}$ **where**

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_\times = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ y & x & 0 \end{bmatrix}$$

- Therefore: $$0 = \mathbf{p'}^T \mathbf{E} \mathbf{p}$$
  - Where $\mathbf{E} = \mathbf{T}_\times \mathbf{R}$ is the 3x3 "essential matrix"
  - Holds whenever **p** and **p'** correspond to the same scene point

Simplifying: $\mathbf{p'^T T \times (Rp) = 0}$

- Properties of E
  - **Ep** is the epipolar line of **p**; $\mathbf{p'}^T$ **E** is the epipolar line of **p'**
    - $\mathbf{p'}^T$ **E p** = 0 for every pair of corresponding points
    - **0 = Ee =** $\mathbf{e'}^T$ **E** where **e** and **e'** are the epipoles
      - **E** has rank < 3, has 5 independent parameters

  - **E** tells us *everything* about the epipolar geometry

Linear two-view relations

- The Essential Matrix: 0 **= p'**^{T} **E p**
  - First derived by Longuet-Higgins, Nature 1981
    - also showed how to compute camera **R** and **T** matrices from **E**
    - **E** has only 5 free parameters (three rotation angles, two transl. directions)
  - Only applies when cameras have same internal parameters
    - same focal length, aspect ratio, and image center
    - Usually for normalized camera coordinates

Linear two-view relations**(cont.)**

- The Fundamental Matrix: 0 **= p'**^{T} **F p**
  - **F** = (**K'**$^{-1}$)^T **E K**$^{-1}$, where **K**$_{3x3}$ and **K'**$_{3x3}$ contain the internal parameters
  - Gives epipoles, epipolar lines

$$K = \begin{pmatrix} f_u & s & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{pmatrix}$$

  - **F** (like **E**) is defined only up to a scale factor and has rank 2 (7 free params) [There are 9 elements, but scaling is not significant and det F = 0]
    - Generalization of the essential matrix
    - Can't uniquely solve for **R** and **T** (or **A** and **A'**) from **F**
    - Can be computed using linear methods
      - R. Hartley, *In Defence of the 8-point Algorithm*, ICCV 95
    - Or nonlinear methods

Estimating F

- *When solving matrix equations, one needs to take care of the conditioning of matrices. In the computation of Fundamental matrix, issues on conditioning needs to be taken care off.*

- We will go through the material below which is taken from the paper by Richard Hartley:

  – "In Defence of the Eight-Point Algorithm", *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 19, no. 6, June 1997

Estimating F (cont.)

$\tilde{p}_1$ and $\tilde{p}_2$ represent the image point on image 1 and image2 respectively:

$$\tilde{p}_1 = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \qquad \tilde{p}_2 = \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix}$$

Given a set of point correspondences, i.e. pairs of $\tilde{p}_1$ and $\tilde{p}_2$, we want to compute F using

$$\tilde{p}_2^T \; F \; \tilde{p}_1 = 0$$

Estimating F (cont.)

We can write the following to represent $\tilde{p}_2^T \ F \ \tilde{p}_1 = 0$

$$[u' \quad v' \quad 1]\underbrace{\begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix}}_{F}\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = 0$$

Since [u' v' 1] and [u v 1] are all known through point correspondences, *F* is the only unknown to be solved. We can write the above equation in *Ax = b* form.

Estimating F (cont.)

$$\begin{bmatrix} uu' & vu' & u' & uv' & vv' & v' & u & v & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$$

Estimating F (cont.)

- The solution for f is the least eigenvector of A^TA.
- Once we get the solution for f, we can form the 3x3 matrix F.
- But F should be rank 2. How to enforce the rank 2 condition ?

  -- we can use SVD to enforce the rank 2 condition.

Estimating F (cont.)

Suppose

$$F = U \begin{bmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & s_3 \end{bmatrix} V^T$$

Let

$$F' = U \begin{bmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T$$

Then F’ is the rank-2 matrix that most closely approximates F in the Frobenius norm sense i.e. |F – F’| is minimum, where |.| denotes the Frobenius norm.

The normalized eight-point algorithm

- Linear solution known as 8-point algorithm, due to Longuet-Higgins (1981)
  - Naïve implementation can be numerically unstable

- [R. Hartley 1995]
  - Center the image data at the origin, and scale it so the mean squared distance between the origin and the data points is 2 pixels:

    $q_i = S\, p_i \quad q_i' = S'\, p_i'.$
  - Use the eight-point algorithm to compute F from the points q_i and q'_i.
  - Enforce the rank-2 constraint.
  - Output S^{-1}F S'.

The normalized eight-point algorithm

- Linear 8-ppoint algorithm and the normalized one.

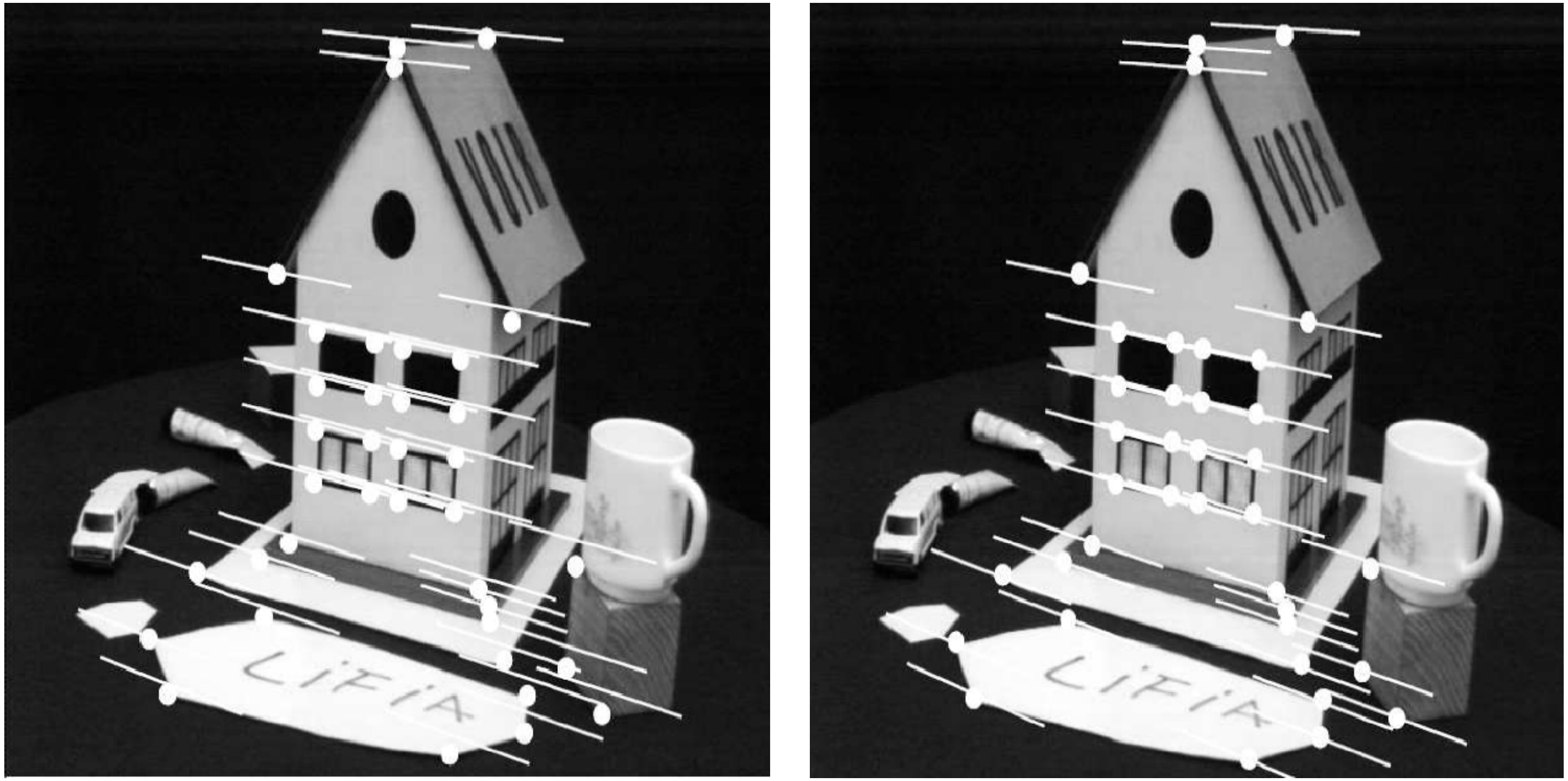


Figure 10.4 of the textbook