Computer Vision

9. Two Views

I-Chen Lin

College of Computer Science

National Yang Ming Chiao Tung University

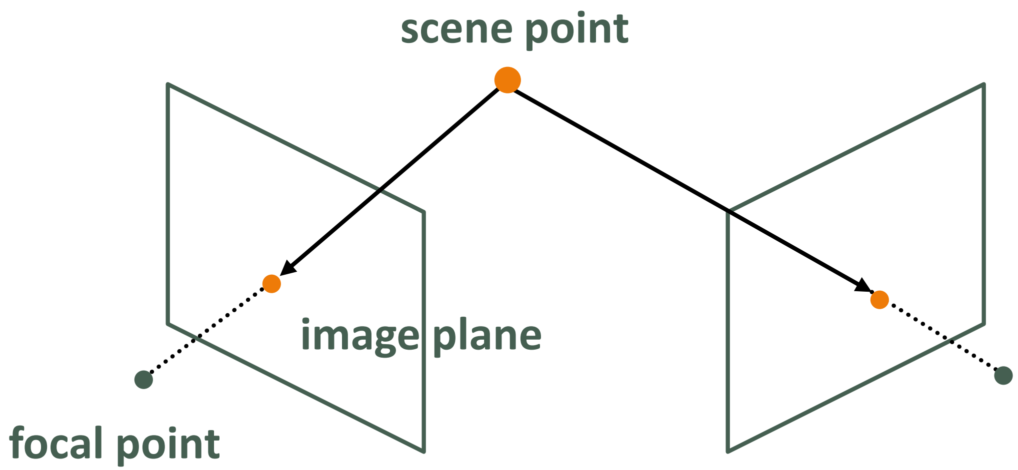
Objective

- Epipolar geometry

Textbook:
• David A. Forsyth and Jean Ponce, Computer Vision: A Modern Approach, Prentice Hall, New Jersey, (1^{st} Ed. 2003, 2^{nd} Ed. 2012)
• 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

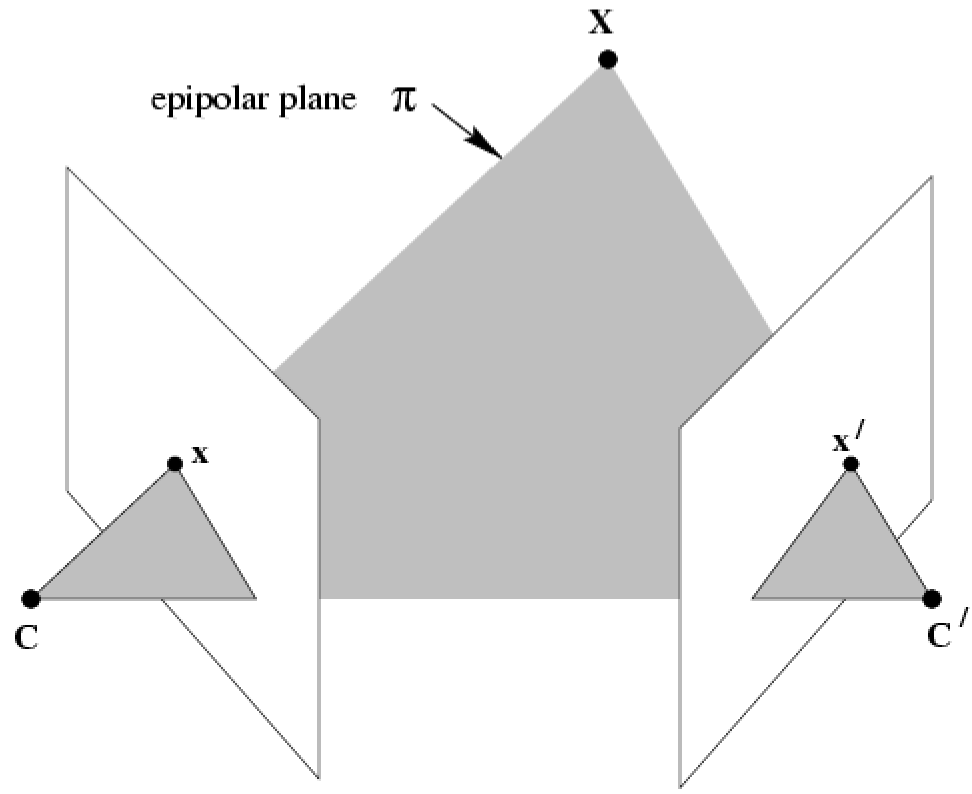
Two-view projective geometry

- 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 $\{x_i \leftrightarrow x'_i\}$, $i=1,\dots,n$, what are the cameras C and C' for the two views? Or what is the geometric transformation between the views?

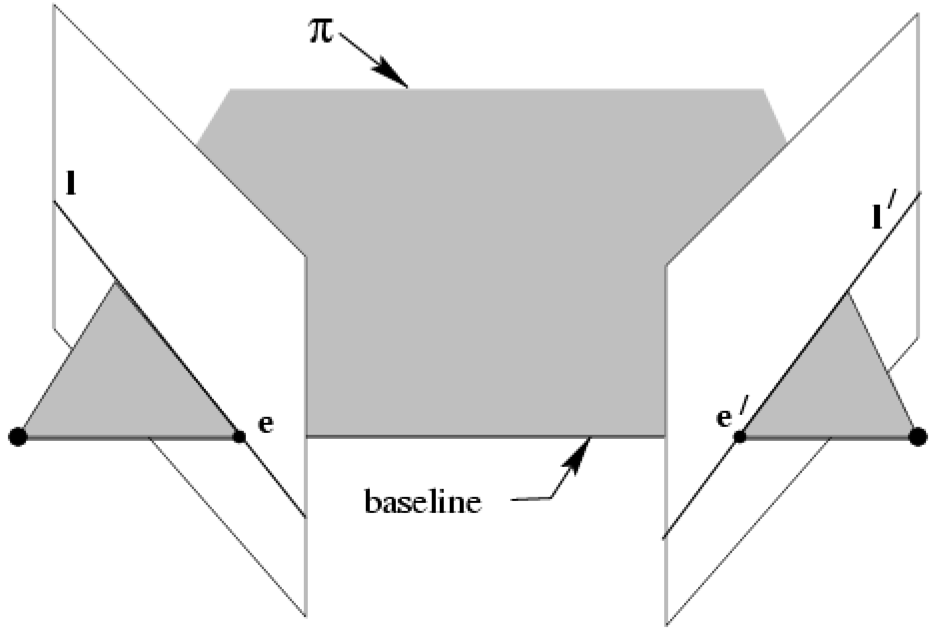
- Scene geometry (structure):
  - Given corresponding image points $x_i \leftrightarrow x'_i$ and cameras C, C', what is the position of the point X in space?

Epipolar geometry



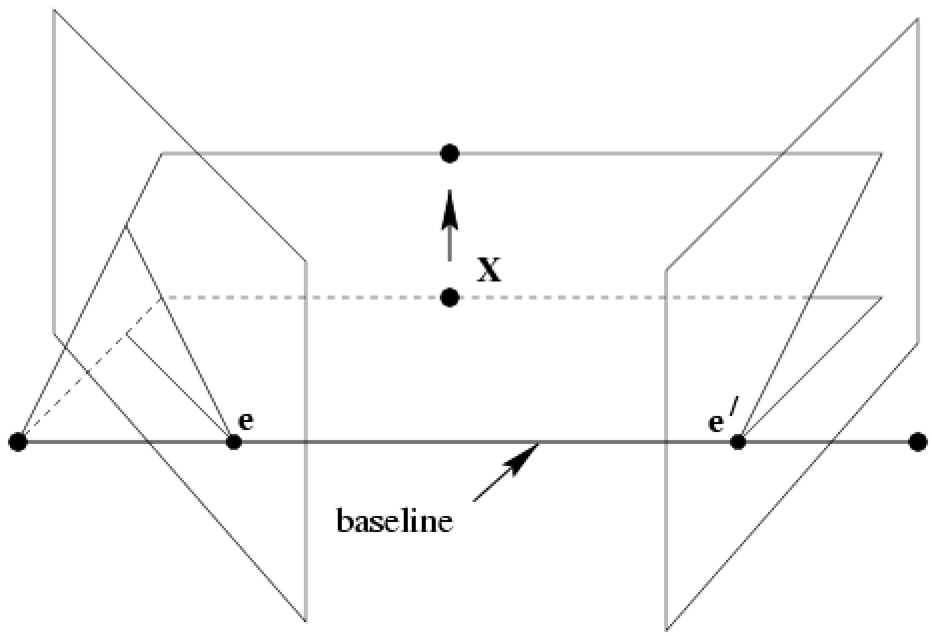
C,C',x,x' and X are coplanar

Epipolar geometry



All points on π project on l and l'

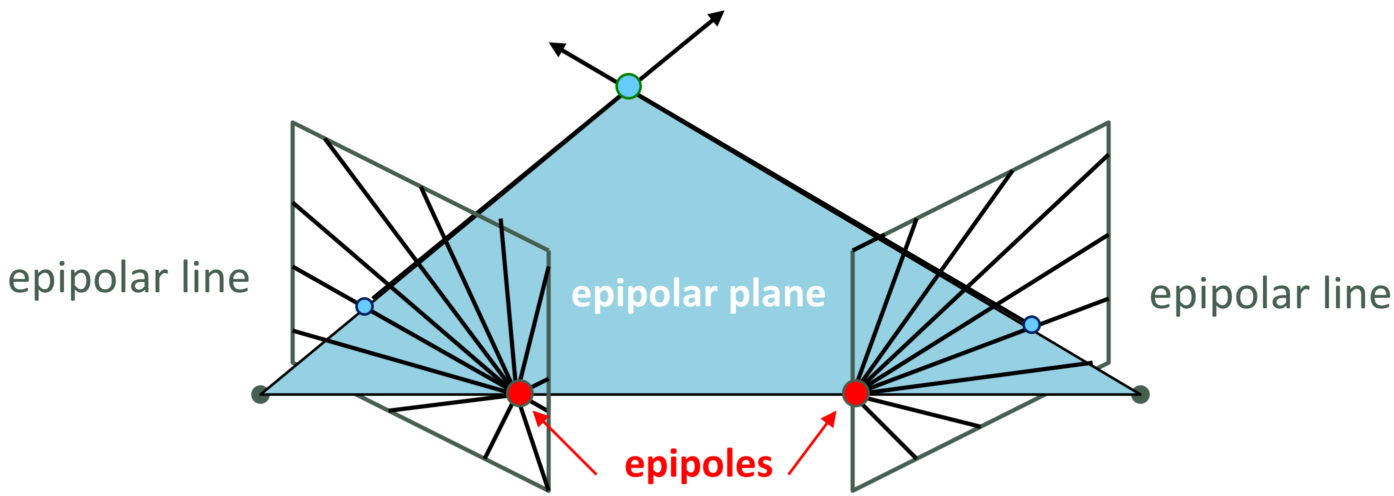
Epipolar geometry



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

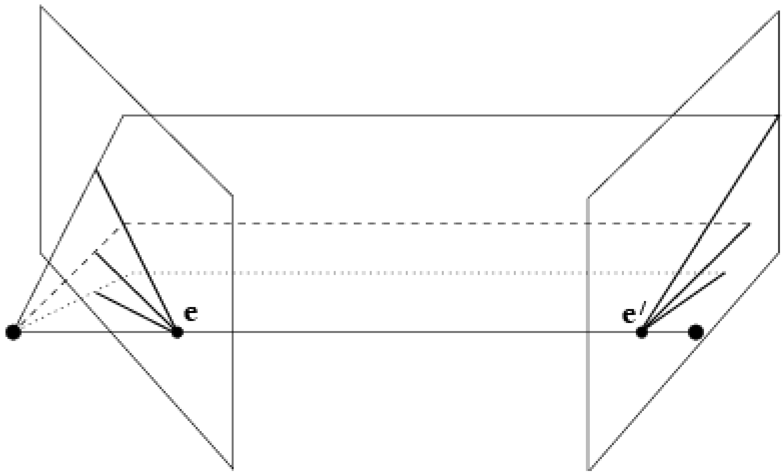
Epipolar geometry

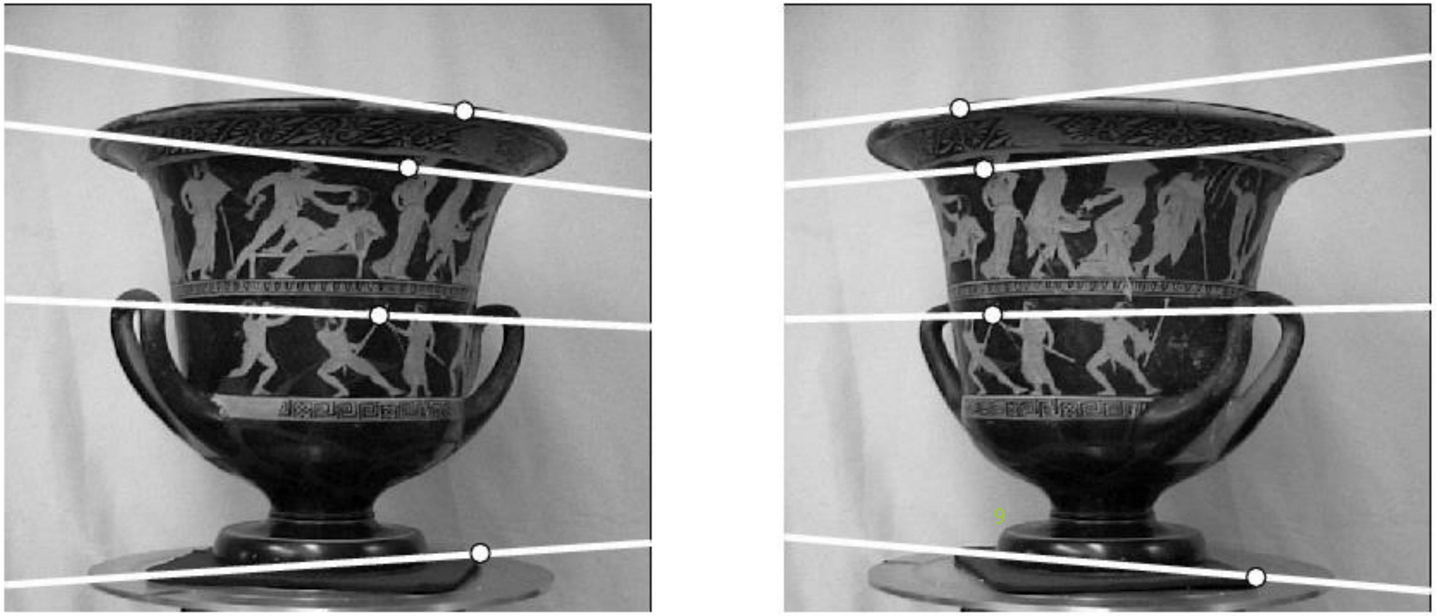
- What does one view tell us about another?
  - Point positions in 2nd view must lie along a known line



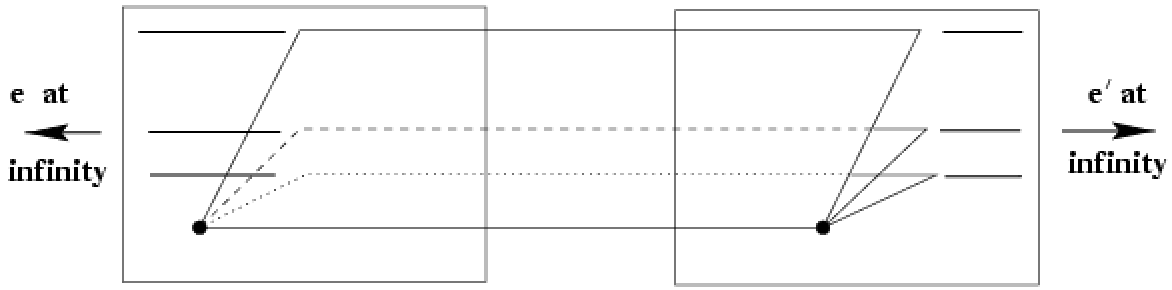
- Epipolar Constraint
  - Extremely useful for stereo matching
    - Reduces problem to 1D search along *conjugate epipolar lines*
  - Also useful for view interpolation...

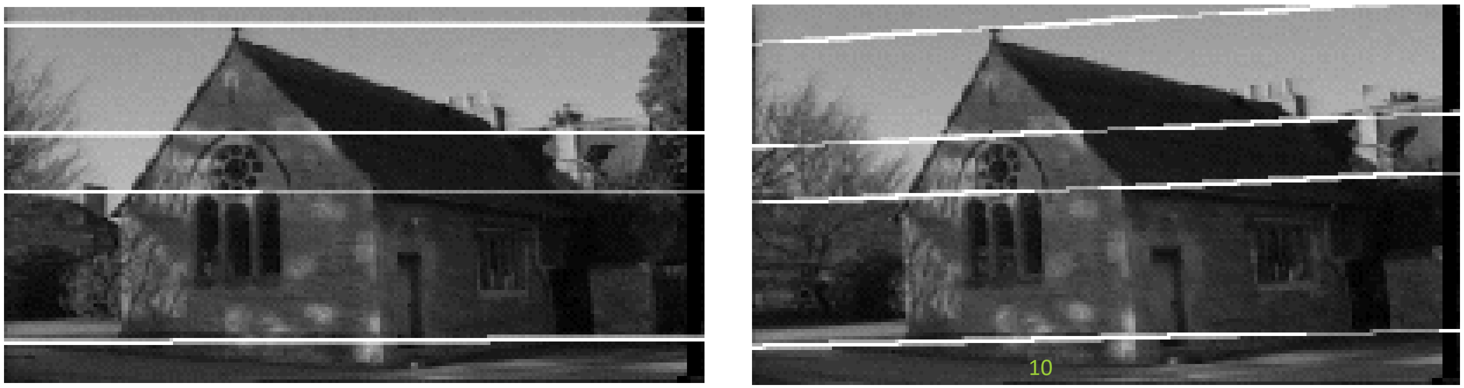
Example: converging cameras



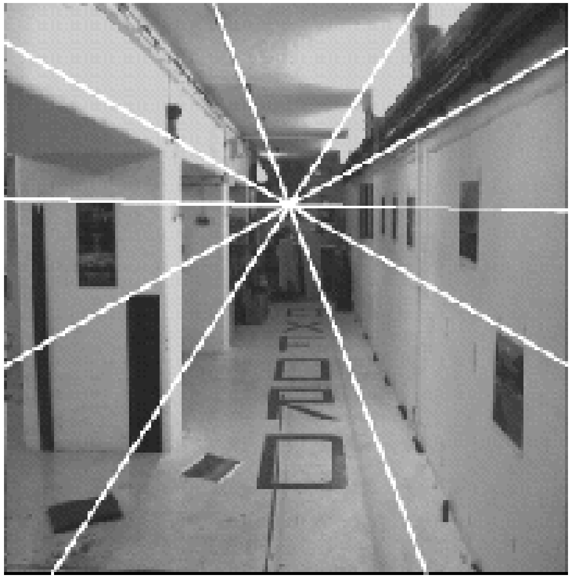


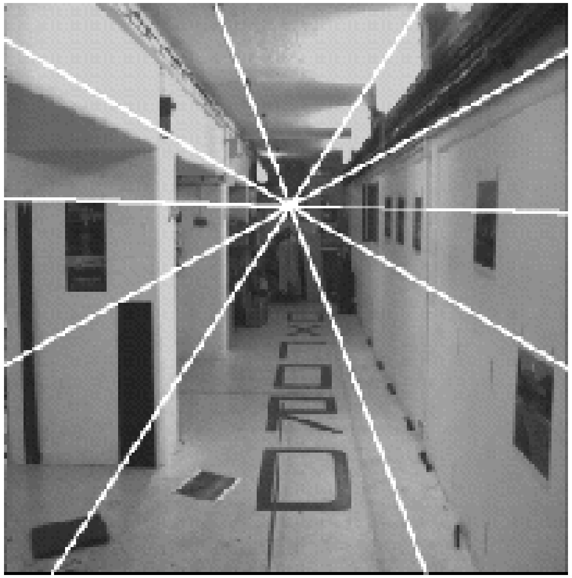
Example: motion parallel with the image plane

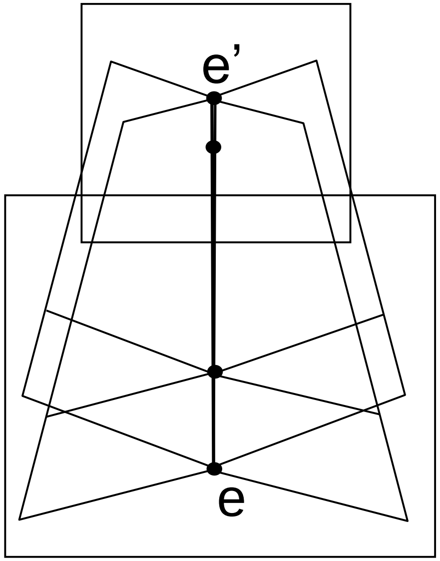




Example: forward motion







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 **R, T** $\rightarrow$ project $\rightarrow$ **A**

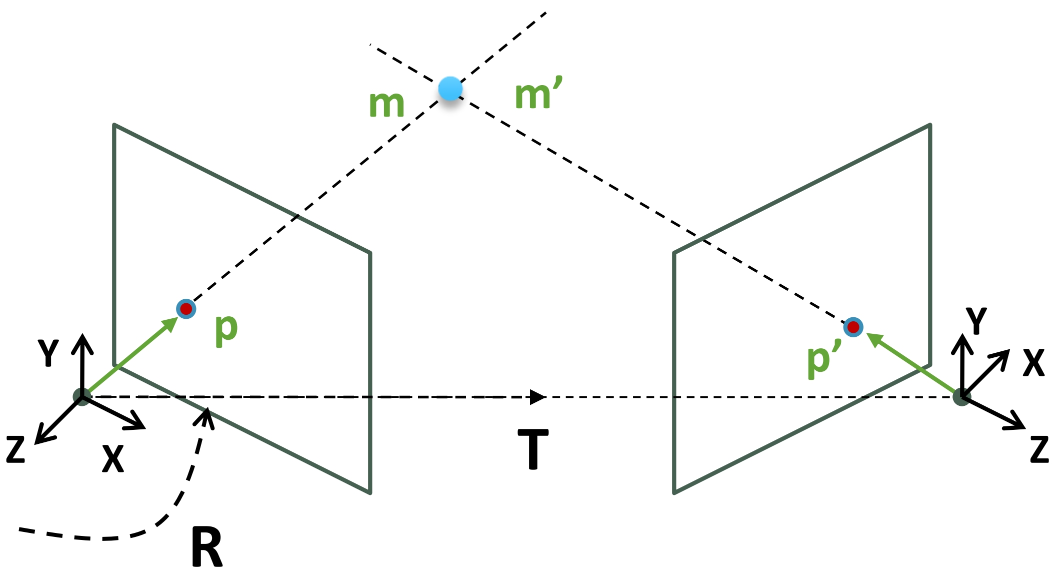
$$\mathbf{\Pi} = \begin{bmatrix} \alpha & -\alpha \cot\theta & u_0 \\ 0 & \dfrac{\beta}{\sin\theta} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{3x3} & \mathbf{T}_{3x1} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix}$$

Then we can write the projection as:

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

Epipolar algebra

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



$$\mathbf{m'} = \mathbf{Rm} + \mathbf{T}$$
$$\mathbf{z'p'} = \mathbf{zRp} + \mathbf{T}$$

- Note that **m'** is $\perp$ to **T**$\times$**m'**
- So 0 = $\mathbf{m'}^T \mathbf{T}\times\mathbf{m'} = \mathbf{m'}^T \mathbf{T}\times(\mathbf{Rm} + \mathbf{T}) = \mathbf{m'}^T \mathbf{T}\times(\mathbf{Rm}) = \mathbf{z'p'Tx(zRp)}$

  0 = p'(TxR)p (z, z' are scalar)

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

- We can write a cross-product $a\times b$ as a matrix equation
  - $\mathbf{a \times b = A_{\times} b}$ **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: $p'^T T\times(Rp) = 0$

- Properties of E
  - **Ep** is the epipolar line of **p**; **p'**^{T} **E** is the epipolar line of **p'**
    - **p'**^{T} **E p** = 0 for every pair of corresponding points
    - **0 = Ee = 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**: $0 = \mathbf{p'}^T \mathbf{E}\, \mathbf{p}$
  - 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

p and *p'* here are in a general (not normalized) camera coordinate

Linear two-view relations (cont.)

- The **Fundamental Matrix**: $0 = \mathbf{p'}^T \mathbf{F} \mathbf{p}$
  - $\mathbf{F} = (\mathbf{K'}^{-1})^T \mathbf{E} \mathbf{K}^{-1}$, where $\mathbf{K}_{3x3}$ and $\mathbf{K'}_{3x3}$ 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 $\tilde{p}_2^T \; F \; \tilde{p}_1 = 0$

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

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 = 0* 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

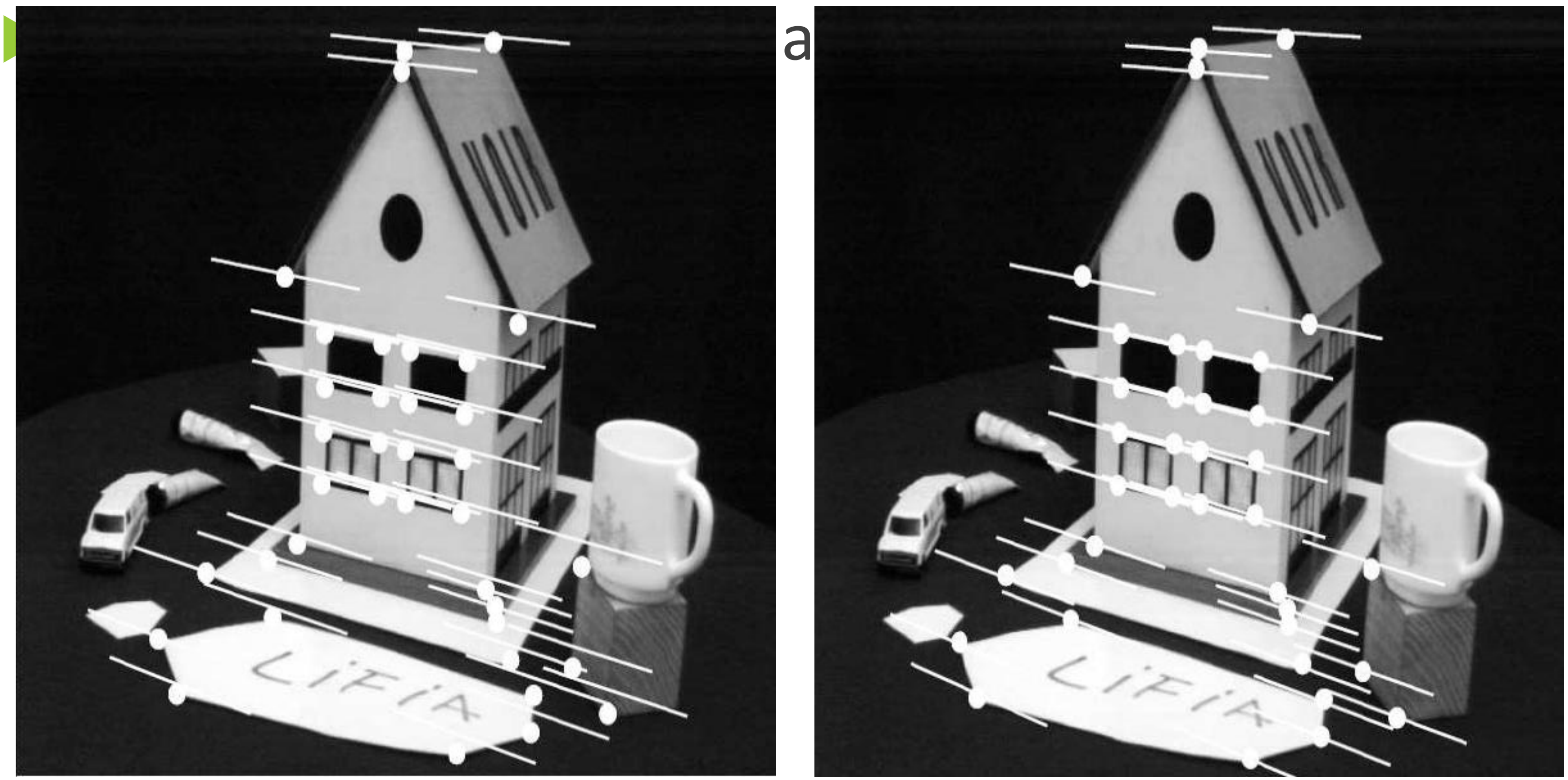


Figure 10.4 of the textbook

Objective

- Epipolar geometry
- Two-view approach for R, T

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.

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

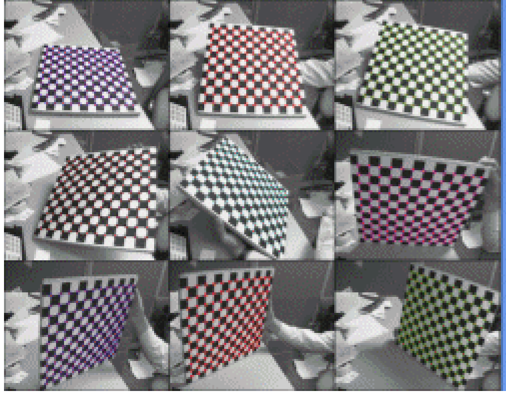
• Prof. S. Seitz and P. Heckbert, Image-based modeling and rendering course notes, CMU.

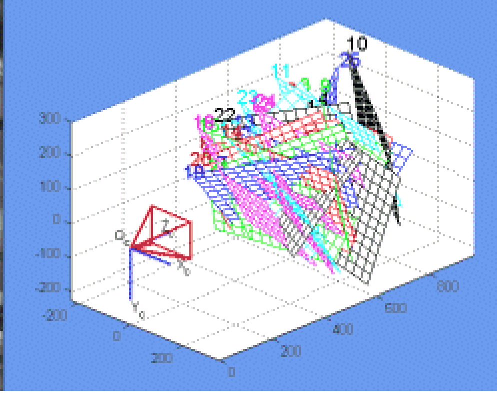
• J. Weng, T.S. Huang, N. Ahuja, Motion and Structure from Image Sequences, 1993.

• Camera calibration toolbox for matlab.

• D. Frolova, D. Simakov, Slides of “Matching with Invariant Features”.

Camera calibration

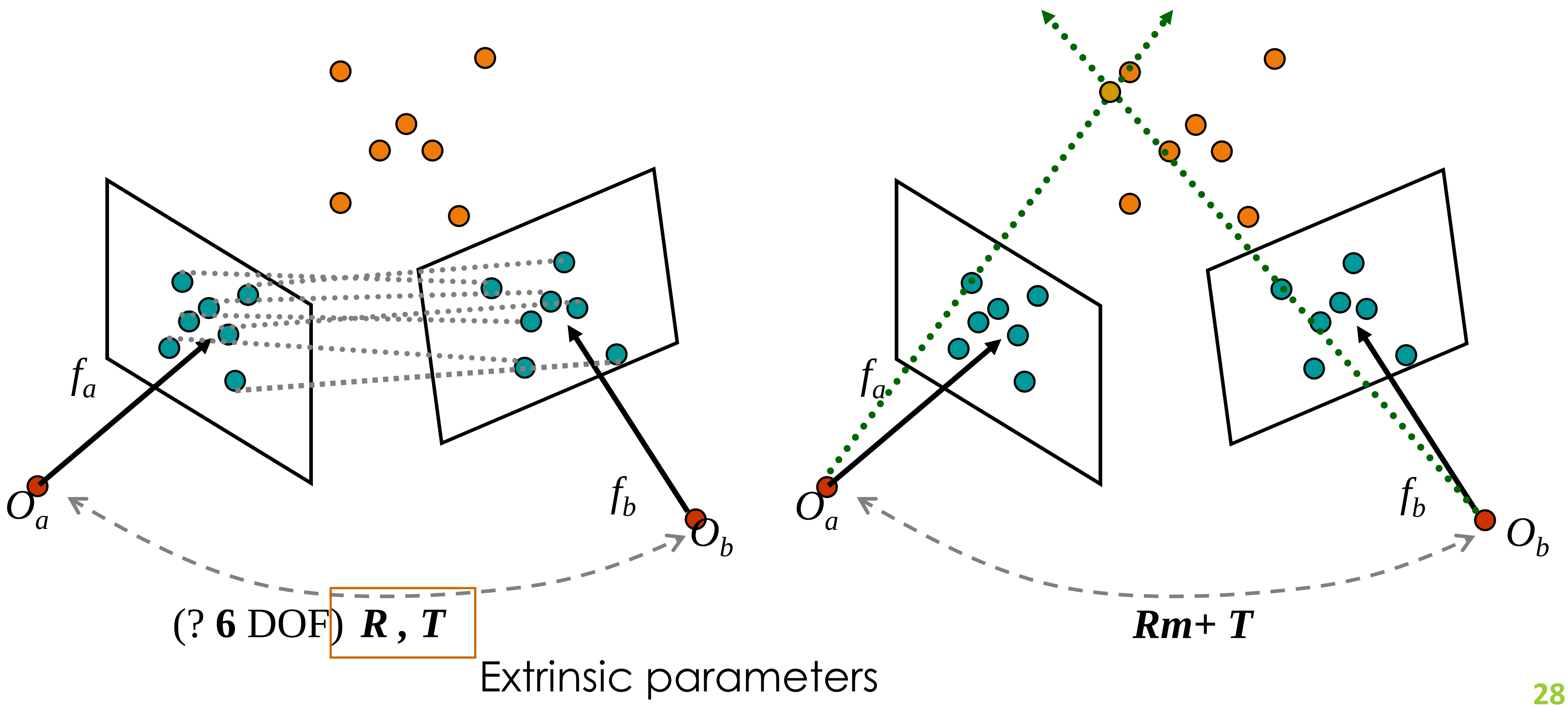




- Famous tools
  - A flexible new technique for camera calibration
    - http://research.microsoft.com/~zhang/calib/
    - Z. Zhang. A flexible new technique for camera calibration. IEEE Trans. Pattern Analysis and Machine Intelligence, 22(11):1330-1334, 2000.
  - Camera calibration toolbox for matlab
    - http://www.vision.caltech.edu/bouguetj/calib_doc/
  - OpenCV camera calibration
  - Tsai's camera model
    - http://www.cs.cmu.edu/~rgw/TsaiDesc.html
    - "A versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses", Roger Y. Tsai, IEEE J. Robotics and Automation, Vol. RA-3, No. 4, 1987, pages 323-344.

Triangulation (stereo)

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



Triangulation (stereo)

- Constructing 3D structure from two views.
  - H.C. Longuet-Higgins (Nature’81).
  - J.Weng et al. The two-view approach (PAMI’89).

- Given some points in correspondence across two images (in a normalized camera model), $\{(u_j, v_j)\}$,
  - Estimate R, T from corresponding points.
  - 3D position estimation from triangulation.
  - (optional) non-linear optimization

The two-view approach

- Without loss of generality, the images of different view direction *d1, d2* is regarded as a rigid-body motion of an object between *t1, t2*.

$\boldsymbol{x_i} = (x_i, y_i, z_i)$t is the 3D position of point P_i at time t_1.

$\boldsymbol{x_i}' = (x_i', y_i', z_i')$t is the 3D position of point P_i at time t_2.

$\boldsymbol{X_i} = (u_i, v_i, 1)$t is the projected vector of P_i at time t_1.

$\boldsymbol{X_i}' = (u_i', v_i', 1)$t = $(x_i'/z_i', y_i'/z_i', 1)$t is the projected vector of P_i at time t_2.

The two-view approach (1)

- Step (1). Solving for essential matrix E.

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

$$A = \begin{bmatrix} u_1u'_1 & u_1v'_1 & u_1 & v_1u'_1 & v_1v'_1 & v_1 & u'_1 & v'_1 & 1 \\ u_2u'_2 & u_2v'_2 & u_2 & v_2u'_2 & v_2v'_2 & v_2 & u'_2 & v'_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_nu'_n & u_nv'_n & u_n & v_nu'_n & v_nv'_n & v_n & u'_2 & v'_n & 1 \end{bmatrix}$$

- $\min_h \| Ah \|$, subject to $\| h \| = 1$.

$$E = \begin{bmatrix} E_1 & E_2 & E_3 \end{bmatrix} = \sqrt{2} \begin{bmatrix} h_1 & h_4 & h_7 \\ h_2 & h_5 & h_8 \\ h_3 & h_6 & h_9 \end{bmatrix}$$

The solution of h is the unit eigenvector of A^tA associated with the smallest eigenvalue.

The two-view approach (2)

- Step (2). Determining a unit vector T_s with $T_0 = \pm T_s$.
  - $\min_{T_s} \| E^t T_s \|$, subject to $\| T_s \| = 1$.

The solution of T_s is the unit eigenvector of EE^t associated with the smallest eigenvalue.

$$\boxed{\begin{aligned} & E = T_{\times} R = \begin{bmatrix} E_1 & E_2 & E_3 \end{bmatrix} = \begin{bmatrix} T_{\times} R_1 & T_{\times} R_2 & T_{\times} R_3 \end{bmatrix} \\ & \therefore E_1, E_2, E_3 \perp T \quad \Rightarrow \quad E^t T_s = 0 \end{aligned}}$$

T_s vs. **e'**

- if($\Sigma_i (T_s \times X_i') \bullet (E\, X_i) < 0$), $T_s = -T_s$.

The two-view approach (3)

- Step (3). Determining rotation matrix *R*.
  - Without noise, $W=R$

    $$W = [(E_1 \times T_s + E_2 \times E_3)\ \ (E_2 \times T_s + E_3 \times E_1)\ \ (E_3 \times T_s + E_1 \times E_2)]$$

    Using the identity equation: (a x b) x c = (a · c) b – (b · c) a

  - With noise,

    $\min_R \| R - W \|$, subject to: R is a rotation matrix.

The two-view approach (4)

- Step (4). Checking $T = 0$, If $T \neq 0$, determine the sign of T_0.

if for all $i = 1 \sim n$, $\| X_i' \times R X_i \| / (\|X_i'\| \cdot \| X_i\|) \leq \alpha$
then report $T \approx 0$.

else if ($\Sigma_i(T_s \times X_i') \bullet (X_i' \times R X_i) > 0$), then $T_0 = T_s$,
otherwise $T_0 = -T_s$.

The two-view approach (5)

- Step (5). If $T \neq 0$, estimate relative depths.

- To find
$$Z_i = \left(\frac{z_i'}{\|T\|}, \frac{z_i}{\|T\|} \right)^t = \left(\tilde{z}_i', \tilde{z}_i \right)$$

by
$$\min \left\| \left[X_i' \quad -RX_i \right] Z_i - T^0 \right\|$$

m’ = Rm + T
z’x’ = zRx + T
z’x’ – zRx = T

Note: The T_0 is an unit vector.
We do not know the scale $|T|$ without additional reference.

The nonlinear optimization

- Two-view linear algorithms are often easily disturbed by noise.
  - More calibration points.
  - Nonlinear optimization.
- First, take the result of the two-view linear algorithm as an initial guess.
- Approximate the R, T by $\min_m \{\| f(u,m) \|\}$ in a nonlinear least square approach
  - E.g. the Levenberg-Marquardt method, or the Gauss-Newton method.
  - $f(u,m) = prj(m, y(u,m)) - u$

where u is the observed projected position,
m is the motion parameters(R,T), $y(u,m)$ is the best 3D positions of P,
and $prj(m, x)$ is the projected position of the input structure x and motion m.

Limits of the triangulation method

- Motion results from R and T could be ambiguous:
  - Insufficient (*x* or *y*) rotational variation
  - Insufficient depth variation
- Camera calibration is important (for *K* matrix)
- Need good feature point trackers or manual assistance