# CSE 252B: Computer Vision II

**Lecturer: Serge Belongie**
**Scribe: Tasha Vanesian**

# LECTURE 3
# Calibrated 3D Reconstruction

## 3.1. Geometric View of Epipolar Constraint

We are trying to solve the following problem: given 2 photos of a 3-dimensional scene, understand the relationship of the points between the photos, and use this information to do 3D reconstruction. Last lecture we derived the epipolar constraint algebraically. This lecture we will look at it geometrically.

Figure 1 shows the geometry of the epipolar constraint. Note that all points in the figure are in homogeneous coordinates. Any point which lies along the line between $o_1$ and the point $p$ will project to $\boldsymbol{x}_1$ on the image plane of camera 1. However, all points along this line will sweep out the epipolar line $\boldsymbol{l}_2$ on the image plane of camera 2. The line along the vector $\boldsymbol{T}$ connecting $o_1$ and $o_2$ is called the *baseline*.

## 3.2. Coplanarity Constraint

The coplanarity constraint says that the three vectors $\boldsymbol{T}$, $\boldsymbol{X}_2$, and $R\boldsymbol{X}_1$ must be coplanar, as shown in Figure 1.

[1]Department of Computer Science and Engineering, University of California, San Diego.

April 5, 2004

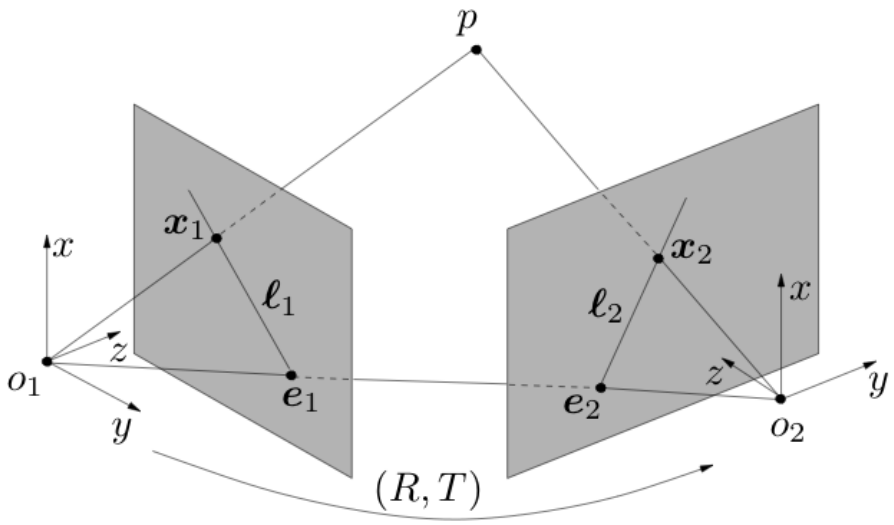


**Figure 1.** The epipolar model, from *Ch. 5 of MaSKS*. The original caption reads: "Two projections $\boldsymbol{x_1}$,$\boldsymbol{x_2} \in \mathbb{R}^3$ of a 3-D point $p$ from two vantage points. The Euclidean transformation between the two cameras is given by *(R,T)* $\in$ *SE(3)*. The intersections of the line ($o_1$, $o_2$) with each image plane are called *epipoles* and are denoted by $\boldsymbol{e_1}$ and $\boldsymbol{e_2}$. The line $\boldsymbol{l_1}$, $\boldsymbol{l_2}$ are called *epipolar lines*, which are the intersection of the plane ($o_1$, $o_2$, $p$) with the two image planes."

### 3.2.1. How do we test the coplanarity of 3 vectors?

It is often useful to be able to test if 3 vectors are coplanar. To do this, we use the (scalar) triple product. For three vectors $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$, the triple product is defined as follows:

$$\boldsymbol{a} \cdot (\boldsymbol{b} \times \boldsymbol{c}) = \boldsymbol{a}^\top \widehat{\boldsymbol{b}} \boldsymbol{c} \tag{3.1}$$

In general, the triple product represents the volume of the parallelepiped spanned by the vectors $\boldsymbol{a}$, $\boldsymbol{b}$, and $\boldsymbol{c}$. If the triple product is equal to zero, this means that the three vectors are coplanar.

This means that we have

$$\boldsymbol{X}_2^\top \widehat{T} R \boldsymbol{X}_1 = 0 \tag{3.2}$$

We can divide equation (3.2) by anything we want, since the RHS is zero. Noting that $\boldsymbol{X}_1 = \lambda_1 \boldsymbol{x}_1$ and $\boldsymbol{X}_2 = \lambda_2 \boldsymbol{x}_2$, and dividing by $\lambda_1 \lambda_2$, we obtain:

$$\boldsymbol{x}_2^\top \widehat{T} R \boldsymbol{x}_1 = 0$$

which can be rewritten as

$$\boldsymbol{x}_2^\top E \boldsymbol{x}_1 = 0 \qquad \text{where} \qquad E = \widehat{T}R \tag{3.3}$$

$E$ is the essential matrix, and is a special case of the "fundamental matrix" $F$. (In particular, $E = F$ if $K_1 = K_2 = I$). We will see later that $E$ lives in the "essential space."

### 3.2.2. Solving for $E$

$$E = \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix} \in \mathbb{R}^{3\times 3}$$

One method of solving for $E$ is to use the Longuet-Higgins 8 point algorithm. This is Algorithm 5.1 in MaSKS.[1]

We would like to convert Equation (3.3) from its current bilinear form to a form that matches the null space problem. We will do this by "stacking" the elements of matrix $E$ into a vector $\boldsymbol{E}^s \in \mathbb{R}^9$. Using MATLAB notation, the stacking operation is denoted by

$$\boldsymbol{E}^s = E(:)$$

As a caveat, if the values in $E$ have any special structure, we lose it when stacking the elements in this manner. (For example, an $E$ estimated in this way will not guarantee that all epipolar lines will intersect at exactly the same point, i.e., at the epipole.) This is discussed more below.

#### Kronecker Product

The Kronecker Product (`kron.m`) is denoted by the $\otimes$ symbol. Given two vectors

$$\boldsymbol{x}_1 = (x_1, y_1, z_1)^\top \text{ and } \boldsymbol{x}_2 = (x_2, y_2, z_2)^\top,$$

their Kronecker product is:

$$\boldsymbol{a} = \boldsymbol{x}_1 \otimes \boldsymbol{x}_2 = \begin{bmatrix} x_1\boldsymbol{x}_2 \\ y_1\boldsymbol{x}_2 \\ z_1\boldsymbol{x}_2 \end{bmatrix} \in \mathbb{R}^9. \tag{3.4}$$

Using the Kronecker product, we can write equation (3.3) as

$$\boldsymbol{a}^\top \boldsymbol{E}^s = 0.$$

This expresses the epipolar constraint for a single correspondence.

[1] Alternative methods for estimating $E$ from fewer than 8 points are possible; see MaSKS p. 122.

If we denote the correspondences by

$$(\boldsymbol{x}_1^j, \boldsymbol{x}_2^j), \qquad j = 1, 2, \ldots, n$$

and construct an $\boldsymbol{a}$ for each correspondence such that

$$\boldsymbol{a}^j = \boldsymbol{x}_1^j \otimes \boldsymbol{x}_2^j$$

then we can stack the $\boldsymbol{a}$'s into the design matrix $\chi$ such that:

$$\chi = \begin{bmatrix} (\boldsymbol{a}^1)^\top \\ \vdots \\ (\boldsymbol{a}^n)^\top \end{bmatrix} \in \mathbb{R}^{n \times 9}.$$

This allows us finally to rewrite equation (3.3) as

$$\chi \boldsymbol{E}^s = \boldsymbol{0} \tag{3.5}$$

Solve this equation for $\boldsymbol{E}^s$ using the SVD (see lecture 2 notes), then reshape back into a matrix form to obtain $E$.

### Degrees of Freedom in $E$

$E$ has 9 entries, but because we are in homogeneous coordinates, we have one less degree of freedom, for a total of 8 degrees of freedom. (Taking into account the known structure of $E$, the true number of degrees of freedom is actually 5, as we will see below.)

In the ideal case (no noise),

$$\text{rank}(\chi) = 8, \qquad \|\chi E^s\|^2 = 0.$$

### What space does $E$ live in?

$E$ is a member of $\mathcal{E}$, known as the "essential space."

$$\mathcal{E} \doteq \{\widehat{T}R \mid R \in SO(3), \boldsymbol{T} \in \mathbb{R}^3\} \subset \mathbb{R}^{3\times 3}$$

Note that $\text{rank}(E) = 2$. The proof is left as an exercise. (Hint: $\text{rank}(\widehat{A}) = 2$ for any $\widehat{A} \in so(3)$.)

Here we can see that $E$ has 5 true degrees of freedom, as it is the product of a skew symmetric matrix (3 dof) and a rotation matrix (3 dof) for a total of 6 dof less 1 dof due to the scale ambiguity.

### The Structure of $E$

Huang and Faugeras [1989], showed that

$$E = U\Sigma V^\top \quad \text{where} \quad \Sigma = diag\{\sigma, \sigma, 0\}$$

The proof can be found on p. 114 of MaSKS. This is the characterization of an essential matrix. If you want to create a valid essential matrix, you can

do so by creating a $\Sigma$ as given above and using two orthogonal matrices to compute $E = U\Sigma V^\top$.

To solve for a valid $E$, we need to project it onto $\mathcal{E}$. Call $F$ the initial estimate of $E$, and let

$$F = U diag\{\lambda_1, \lambda_2, \lambda_3\} V^\top, \quad \lambda_1 \geq \lambda_2 \geq \lambda_3.$$

Then choose

$$E = U diag\{\sigma, \sigma, 0\} V^\top, \quad \sigma = \frac{\lambda_1 + \lambda_2}{2}$$

This choice of $E$ is the one the minimizes the Frobenius norm $\|E - F\|_f^2$.

This two step process of estimating $E$ from the stacked vector $\boldsymbol{E}^s$ and then projecting it onto $\mathcal{E}$ is a good first cut, but note that there could be a value of $E$ leading to a smaller value of $\|\chi E^s\|^2$.

### 3.2.3. The Universal Scale Ambiguity

Reconstruction of a scene can only be found up to a universal scale factor. Note that if we doubled the size of the scene while doubling the length of $\boldsymbol{T}$, we'd get the same projected points on the image planes, and consequently the same epipolar lines. We will use a "normalized essential matrix" which means we will take $\sigma = 1$. This is equivalent to choosing $\|\boldsymbol{T}\|^2 = 1$, and does not affect the rotation matrix $R$.

### 3.2.4. Finding $\boldsymbol{T}$

To tease out $\boldsymbol{T}$ from $E$, note that

$$EE^\top = \widehat{T} \underbrace{RR^\top}_{=I} \widehat{T}^\top \tag{3.6}$$

$$= -\widehat{T}^2 \tag{3.7}$$

since $\widehat{T}^\top = -\widehat{T}$.

$$tr(EE^\top) = \text{sum of the squares of the singular values} \tag{3.8}$$

$$= 2\sigma^2 = 2 \cdot 1 = 2 \quad \text{(using convention } \sigma = 1) \tag{3.9}$$

Remember that $\boldsymbol{T} = (T_x, T_y, T_z)^\top$, so

$$\widehat{T} = \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix}$$

and

$$\widehat{T}^2 = \begin{bmatrix} -T_z^2 - T_y^2 & T_x T_y & T_x T_z \\ T_x T_y & -T_z^2 - T_x^2 & T_y T_z \\ T_x T_z & T_y T_z & -T_y^2 - T_x^2 \end{bmatrix}$$

Note that $\widehat{T}^2$ is symmetric.

$$-tr(\widehat{T}^2) = 2(T_x^2 + T_y^2 + T_z^2) \tag{3.10}$$

$$= 2\|\boldsymbol{T}\|^2 \tag{3.11}$$

Since $\|\boldsymbol{T}\|^2 = 1$, this means that

$$-\widehat{T}^2 = \begin{bmatrix} 1 - T_x^2 & -T_x T_y & -T_x T_z \\ -T_x T_y & 1 - T_y^2 & -T_y T_z \\ -T_x T_z & -T_y T_z & 1 - T_z^2 \end{bmatrix}$$

From this matrix, we can only get the three entries in $\boldsymbol{T}$ (normalized) up to a global sign. This means we don't know which way the baseline goes.

### 3.2.5. Finding $R$

The Longuet-Higgins 8-point algorithm paper is on the main class website, and describes one way of finding $R$. MaSKS has a less intuitive but more elegant method which was presented in class.

Let $E = U\Sigma V^\top$ with $\Sigma = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ and $R_Z(\pm\frac{\pi}{2}) = \begin{bmatrix} 0 & \mp 1 & 0 \\ \pm 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ (rotation about the $Z$ axis). Then the equations for pose recovery are:

$$R = UR_Z^\top(\pm\frac{\pi}{2})V^\top, \qquad \widehat{T} = UR_Z^\top(\pm\frac{\pi}{2})\Sigma U^\top$$

#### Twisted Pairs

Accounting for the $\pm$ signs, there are 4 possible combinations of solutions for $R$ and $\boldsymbol{T}$. These are shown geometrically in Figure 2. (a) is the correct solution, in which the reconstructed point is in front of both cameras. The direction of the translation vector from the first to the second camera is reversed in (b). (c) and (d) are known as "twisted pairs" – they are related to (a) and (b) by a rotation of 180° about the baseline. In practice, you can make a program output all 4 possibilities for $R$ and $\boldsymbol{T}$, but the only solution that makes sense is (a), when the all points are in front of the camera.

### 3.2.6. Estimating Depths given $R$ and $\boldsymbol{T}$

The Longuet-Higgins paper solves this problem algebraically; again, MaSKS has a more elegant solution:
For the $j^{\text{th}}$ point:

$$\boldsymbol{X}_2^j = R\boldsymbol{X}_1^j + \gamma\boldsymbol{T}, \quad \text{where } \gamma \text{ is the universal scale factor}$$

$$\lambda_2^j \boldsymbol{x}_2^j = \lambda_1^j R\boldsymbol{x}_1^j + \gamma\boldsymbol{T}$$

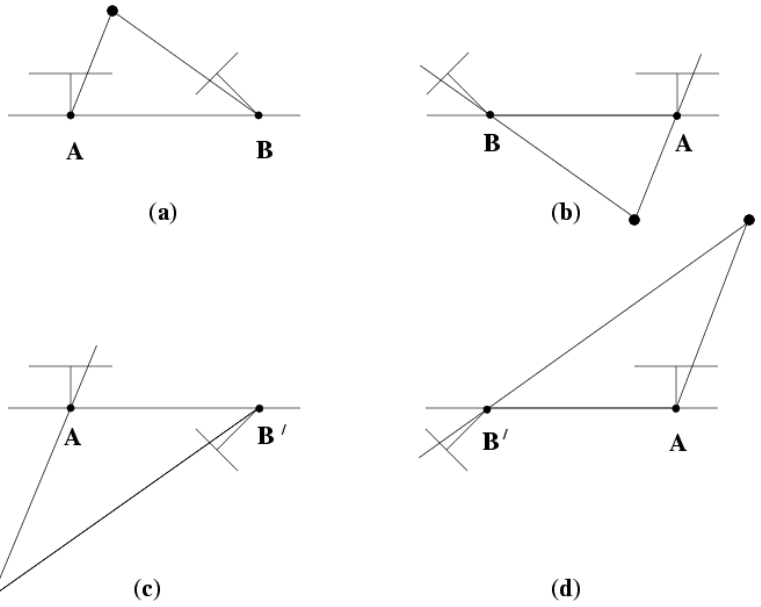


**Figure 2.** Original caption from *Ch. 8 of Hartley and Zisserman* reads: "The four possible solutions for calibrated reconstruction from E. Between the left and right sides there is a baseline reversal. Between the top and bottom rows camera B rotates 180° about the baseline. Note, only in (a) is the reconstructed point in front of both cameras."

Since one set of depths can be obtained from the other, we can algebraically eliminate one set of depths by hitting both sides with $\widehat{\boldsymbol{x}_2^j}$ and using the fact that $\widehat{\boldsymbol{x}_2^j}\boldsymbol{x}_2^j = 0$

$$\lambda_1^j \widehat{\boldsymbol{x}_2^j} R \boldsymbol{x}_1^j + \gamma \widehat{\boldsymbol{x}_2^j} \boldsymbol{T} = 0, \quad j = 1, 2, \ldots, n$$

For a single point, we have:

$$M^j \boldsymbol{\lambda}^j = \underbrace{[\widehat{\boldsymbol{x}_2^j} R \boldsymbol{x}_1^j, \widehat{\boldsymbol{x}_2^j} \boldsymbol{T}]}_{\in \mathbb{R}^{3\times 2}} \underbrace{\begin{bmatrix} \lambda_1^j \\ \gamma \end{bmatrix}}_{\in \mathbb{R}^2} = 0$$

Since we want all of the depths, let

$$\boldsymbol{\lambda} = (\lambda_1^1, \lambda_1^2, \lambda_1^3, \ldots, \lambda_1^n, \gamma)^\top \in \mathbb{R}^{n+1}$$

and form the matrix $M \in \mathbb{R}^{3n\times(n+1)}$:

$$M \doteq \begin{bmatrix} \widehat{\boldsymbol{x}_2^1} R \boldsymbol{x}_1^1 & 0 & 0 & 0 & \widehat{\boldsymbol{x}_2^1} \boldsymbol{T} \\ 0 & \widehat{\boldsymbol{x}_2^2} R \boldsymbol{x}_1^2 & 0 & 0 & \widehat{\boldsymbol{x}_2^2} \boldsymbol{T} \\ \vdots & 0 & \ddots & 0 & \vdots \\ 0 & 0 & 0 & \widehat{\boldsymbol{x}_2^n} R \boldsymbol{x}_1^n & \widehat{\boldsymbol{x}_2^n} \boldsymbol{T} \end{bmatrix} \tag{3.12}$$

so that $M\boldsymbol{\lambda} = \mathbf{0}$. This is a problem we know how to solve using the SVD.

All of this assumes that we have perfect correspondence, no noise, calibrated cameras, and that all 8 points are in "general position," e.g., they do not lie on the same plane (see next lecture).

The Longuet-Higgins 8 point algorithm was mostly of theoretical interest until Hartley wrote a paper called "In Defence of the 8-Point Algorithm" [1995] showing that the algorithm is actually practical (viz. numerically stable) with real data after a simple normalization step.