3D Reconstruction with two Calibrated Cameras

Carlo Tomasi

The *standard reference frame* for a camera C is a right-handed Cartesian frame with its origin at the center of projection of C, its positive Z axis pointing towards the scene along the optical axis of the lens, and its X axis pointing to the right[1] along the rows of the camera sensor. As a consequence, the Y axis points downwards along the columns of the sensor. Coordinates in the standard reference frame are measured in units of focal distance.

Let $\mathbf{P}$ and $\mathbf{Q}$ denote the homogeneous coordinates of the same 3D point $\mathcal{P}$ in the standard reference frames of two cameras C and D, and let

$$\mathbf{Q} \sim g\mathbf{P} \quad \text{where} \quad g = \begin{bmatrix} R & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{1}$$

be the rigid transformation between the two reference systems. The symbol $\sim$ denotes equality up to a nonzero multiplicative constant. In the expression for g above, the column vector $\mathbf{0}$ contains three zeros, and the 3×3 matrix R represents a rotation, so that

$$R^T R = R R^T = I_3 \quad \text{and} \quad \det(R) = 1 \tag{2}$$

where I_3 is the 3×3 identity matrix. The columns of R are the unit vectors along the positive X, Y, Z axes of C expressed in the reference frame of D, and the rows of R are the unit vectors along the positive X, Y, Z axes of D in the reference frame of C. The 3×1 vector $\mathbf{t}$ contains the coordinates of the center of projection of C in the reference frame of D, so that the vector of Euclidean[2] coordinates of the center of projection of D in the reference frame of C is

$$\mathbf{s} = -R^T \mathbf{t} \ . \tag{3}$$

The homogeneous coordinates $\mathbf{p}$ and $\mathbf{q}$ of the projections of $\mathcal{P}$ measured in the two images are

$$\mathbf{p} \sim \Pi_0 \mathbf{P} \quad \text{and} \quad \mathbf{q} \sim \Pi_0 \mathbf{Q} \quad \text{where} \quad \Pi_0 = \begin{bmatrix} I_3 & \mathbf{0} \end{bmatrix} \ . \tag{4}$$

After summarizing key aspects of the geometry of a camera pair in section 1, section 2 shows a method for computing estimates of g and estimates of the coordinates of a set of n points $\mathcal{P}_1, \ldots, \mathcal{P}_n$ in the two camera reference frames from the n pairs $(\mathbf{p}_1, \mathbf{q}_1), \ldots (\mathbf{p}_n, \mathbf{q}_n)$ of noisy measurements of their corresponding images. The transformation g is called camera *motion*, and the point coordinates $\mathbf{P}_i$, $\mathbf{Q}_i$ are collectively called the scene *structure*. Since cameras fundamentally measure angles, both structure and motion are estimated up to a common nonzero multiplicative scale factor. The resulting degree of freedom is eliminated by assuming that

$$\|\mathbf{s}\| = \|\mathbf{t}\| = 1 \ . \tag{5}$$

[1] When the camera is upside-up and viewed from behind it, as when looking through its viewfinder.

[2] "Euclidean" here means non-homogeneous.

An initial version of the method described below appeared in 1981 [3] and is often called the *eight-point algorithm*, because it requires a minimum of $n = 8$ pairs of corresponding image points. The method is suboptimal because it computes a solution in two separate optimization steps rather than one. Because of this, the eight-point algorithm is often followed by a *bundle adjustment* step [4] that minimizes the geometric *reprojection error*

$$e = \sum_{i=1}^{n} \|\mathbf{p}_i - \Pi_0 \mathbf{P}_i\|^2 + \|\mathbf{q}_i - \Pi_0 \mathbf{Q}_i\|^2$$

under the constraints (2) and (5).

1 The Epipolar Geometry

Figure 1 shows the main elements of the *epipolar geometry* of a pair of cameras.

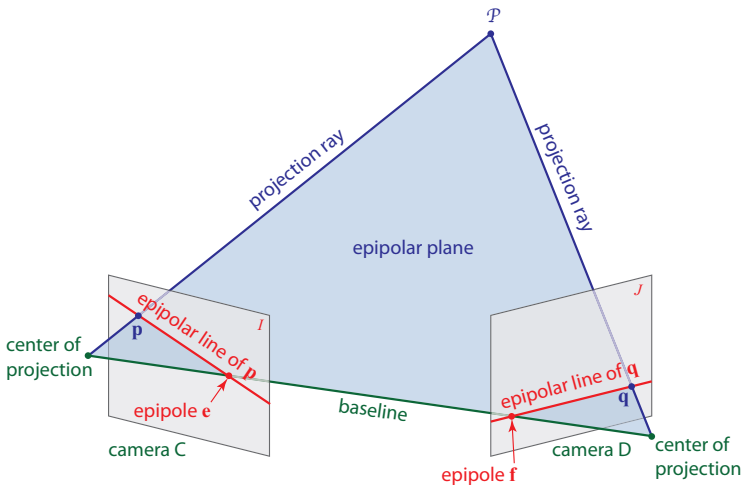

Figure 1: Essential elements of the epipolar geometry of a camera pair.

The world point $\mathcal{P}$ and the centers of projection of the two cameras identify a plane in space, the *epipolar plane* of point $\mathcal{P}$. The Figure shows a triangle of this plane, delimited by the two projection rays and by the *baseline* of the camera pair, that is, the line segment that connects the two centers of projection.[3]

If the image planes are thought of extending indefinitely, the baseline intersects the two image planes at two points called the *epipoles* of the two images. In particular, if the baseline is parallel to an image plane, then the corresponding epipole is a point at infinity.

The epipoles are fixed points for a given camera pair configuration. With cameras somewhat tilted towards each other, and with a sufficiently wide field of view, the epipoles would be image points. Epipole $\mathbf{e}$ in the image I taken by camera C would be literally the image of the center of projection of camera D in I, and *vice versa*. Even if the two cameras do not physically see each other, we maintain this description in an abstract sense: each epipole is the image of one camera in the other image.

The epipolar plane intersects the two image planes along the two *epipolar lines* of point $\mathcal{P}$, each of which passes by construction through one of the two projection points $\mathbf{p}$ and $\mathbf{q}$ and one of the two epipoles. Thus,

[3] We use the term "baseline" for the line *segment*. However, this term is also often used for the entire *line* through the two centers of projection.

epipolar lines come in corresponding pairs, and the correspondence is established by the single epipolar plane for the given point $\mathcal{P}$.

For a different world point $\mathcal{P}$, the epipolar plane changes, and with it do the image projections of $\mathcal{P}$ and the epipolar lines. However, all epipolar planes contain the baseline. Thus, the set of epipolar planes forms a *pencil* of planes supported by the line through the baseline, and the epipoles are fixed.

Suppose now that we are given the two images I and J taken by cameras C and D and a point $\mathbf{p}$ in I. We do not know where the corresponding point $\mathbf{q}$ is in the other image, nor where the world point $\mathcal{P}$ is, except that $\mathcal{P}$ must be somewhere along the projection ray of $\mathbf{p}$. However, the two centers of projection and point $\mathbf{p}$ identify the epipolar plane, and this in turn determines the epipolar line of point $\mathbf{p}$ in image J. The point $\mathbf{q}$ must be somewhere on this line. This same construction holds for any other point $\mathbf{p}$ on the epipolar line in image I.

The projection rays for two arbitrary points in the two images are generically two skew lines in space. The projection rays of two corresponding points, on the the other hand, are coplanar with each other and with the baseline. This so-called *epipolar constraint* can be expressed mathematically as follows.

When expressed in the reference frame of camera D, the directions of the projection rays through corresponding image points with homogeneous coordinates $\mathbf{p}$ and $\mathbf{q}$ are along the vectors

$$R\mathbf{p} \quad \text{and} \quad \mathbf{q}\,,$$

and the baseline in this reference frame is along the translation vector $\mathbf{t}$. Coplanarity of these three vectors can be expressed by stating that their triple product is zero:

$$\mathbf{q}^T(\mathbf{t} \times R\mathbf{p}) = 0 \quad \text{that is,} \quad \mathbf{q}^T[\mathbf{t}]_\times R\mathbf{q} = 0$$

where $\mathbf{t} = \left[\begin{array}{ccc} t(1) & t(2) & t(3) \end{array}\right]^T$ and

$$[\mathbf{t}]_\times = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}\,.$$

In summary, for corresponding points $\mathbf{p}$ and $\mathbf{q}$ the following equation holds:

$$\mathbf{q}^T E\mathbf{p} = 0 \tag{6}$$

where

$$E = [\mathbf{t}]_\times R\,. \tag{7}$$

Equation (6) is called the *epipolar constraint* and the matrix E is called the *essential matrix*. Equation (6) expresses the coplanarity between *any* two points $\mathbf{p}$ and $\mathbf{q}$, on the same epipolar plane. If $\mathbf{p}$ is fixed in image I, then any point $\mathbf{q}$ in image J for which equation (6) holds must be on the epipolar line of $\mathbf{p}$. Formally, the product

$$\boldsymbol{\mu} = E\mathbf{p} \tag{8}$$

for fixed $\mathbf{p}$ is a column vector, and equation (6) becomes

$$\mathbf{q}^T\boldsymbol{\mu} = 0\,.$$

This is the equation of the epipolar line in image J. Conversely, if $\mathbf{q}$ is fixed in the second image, then the product

$$\boldsymbol{\lambda}^T = \mathbf{q}^T E \tag{9}$$

is a row vector, and equation (6) becomes

$$\boldsymbol{\lambda}^T \mathbf{p} = 0 \tag{10}$$

which is the equation of the epipolar line of $\mathbf{q}$ in image I.

Since the epipole in image I belongs to all epipolar lines in I, the vector $\mathbf{e}$ of its homogeneous coordinates must satisfy equation (10) regardless of what point $\mathbf{q}$ is used in the definition (9) of $\boldsymbol{\lambda}$. This can happen only if $\mathbf{e}$ is in the null space of E, which must therefore be degenerate.

It is easy to see that the rank of E is two for any nonzero $\mathbf{t}$. To this end, note first that the matrix $[\mathbf{t}]_\times$ has rank two if $\mathbf{t}$ is nonzero, because

$$[\mathbf{t}]_\times \mathbf{t} = \mathbf{t} \times \mathbf{t} = \mathbf{0}$$

so the null space of $[\mathbf{t}]_\times$ is the line through the origin and along $\mathbf{t}$. Since R is full rank, also the product $E = [\mathbf{t}]_\times R$ has rank 2 if $\mathbf{t} \neq \mathbf{0}$.

Since $[\mathbf{t}]_\times$ is skew-symmetric, $[\mathbf{t}]_\times^T = -[\mathbf{t}]_\times$, its left null space is also along $\mathbf{t}$, and so is that of E. Finally, from equations (3) and (7) we obtain

$$E\mathbf{s} = [\mathbf{t}]_\times R\mathbf{s} = -[\mathbf{t}]_\times R R^T \mathbf{t} = -[\mathbf{t}]_\times \mathbf{t} = \mathbf{0}$$

so that $\mathbf{t}$ spans the left null space of E and $\mathbf{s}$ spans its right space.

This discussion shows that $\mathbf{e}$ and $\mathbf{s}$ are both in the null space of E:

$$E\mathbf{e} = E\mathbf{s} = \mathbf{0} \ .$$

Similar reasoning applies to the epipole $\mathbf{f}$ in image J and the translation vector $\mathbf{t}$, which are in the left null space of E:

$$\mathbf{f}^T E = \mathbf{t}^T E = \mathbf{0}^T \ .$$

Finally, for any vector $\mathbf{v}$ orthogonal to $\mathbf{t}$, the definition of cross product yields

$$\|[\mathbf{t}]_\times \mathbf{v}\| = \|\mathbf{t}\| \, \|\mathbf{v}\| \ .$$

Because of this, the matrix $[\mathbf{t}]_\times$ maps all unit vectors in its row space into vectors of magnitude $\|\mathbf{t}\|$. In other words, the two nonzero singular values of $[\mathbf{t}]_\times$ are equal to each other (and to $\|\mathbf{t}\|$). Since multiplication by an orthogonal matrix does not change the matrix's singular values, we conclude that the essential matrix E has two nonzero singular values equal to each other, and a zero singular value. The left and right singular vectors $\mathbf{u}_3$ and $\mathbf{v}_3$ corresponding to the zero singular value of E are unit vectors along the epipoles and the translation vectors,

$$\mathbf{v}_3 \sim \mathbf{e} \sim \mathbf{s} \quad \text{and} \quad \mathbf{u}_3 \sim \mathbf{f} \sim \mathbf{t} \ . \tag{11}$$

Since the other two singular values of E are equal to each other, the corresponding singular vectors are arbitrary, as long as they form orthonormal triples with $\mathbf{u}_3$ and $\mathbf{v}_3$. The preceding discussion is summarized in Table 1.

2 The Eight-Point Algorithm

The epipolar constraint (6) can be rewritten in the following form:

$$\mathbf{a}^T \boldsymbol{\eta} = 0 \quad \text{where} \quad \mathbf{a} = \mathbf{p} \otimes \mathbf{q} = \begin{bmatrix} p_1 \mathbf{q} \\ p_2 \mathbf{q} \\ p_3 \mathbf{q} \end{bmatrix} \tag{13}$$

For a camera pair (C, D) with nonzero baseline, let

$$\mathbf{Q} = g\mathbf{P} = \begin{bmatrix} R & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \mathbf{P} \quad \text{and} \quad \mathbf{P} = g^{-1}\mathbf{Q} = \begin{bmatrix} R^T & \mathbf{s} \\ \mathbf{0}^T & 1 \end{bmatrix} \mathbf{Q} \quad \text{with} \quad \mathbf{s} = -R^T\mathbf{t}$$

be the coordinate transformations between points $\mathbf{P}$ in C and points $\mathbf{Q}$ in D. The *essential matrix* of the camera pair is the matrix

$$E = [\mathbf{t}]_\times R \quad \text{where} \quad [\mathbf{t}]_\times = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix} .$$

The *epipole* $\mathbf{e}$ in the image I taken by C and that of $\mathbf{f}$ in the image J taken by D satisfy

$$E\mathbf{e} = E^T\mathbf{f} = \mathbf{0} .$$

A point $\mathbf{p}$ in image I and its corresponding point $\mathbf{q}$ in image J satisfy the *epipolar constraint*

$$\mathbf{q}^T E\mathbf{p} = 0 .$$

This equation can also be written as follows:

$$\boldsymbol{\lambda}^T\mathbf{p} = \boldsymbol{\mu}^T\mathbf{q} = 0$$

where

$$\boldsymbol{\lambda} = E^T\mathbf{q} \quad \text{and} \quad \boldsymbol{\mu} = E\mathbf{p}$$

are the vectors of coefficients of the *epipolar line* of $\mathbf{q}$ in image I and that of $\mathbf{p}$ in image J respectively.
The singular value decomposition of E is

$$E \sim U\Sigma V^T = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \mathbf{u}_3 \end{bmatrix} \text{diag}(1, 1, 0) \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3 \end{bmatrix}^T \tag{12}$$

where

$$\mathbf{v}_3 \sim \mathbf{e} \sim \mathbf{s} \quad \text{and} \quad \mathbf{u}_3 \sim \mathbf{f} \sim \mathbf{t}$$

and $\mathbf{u}_1$, $\mathbf{u}_2$, $\mathbf{v}_1$, $\mathbf{v}_2$ are any vectors for which U and V become orthogonal.

Table 1: Definition and properties of the essential matrix.

is the Kronecker product of $\mathbf{p} = \begin{bmatrix} p_1 & p_2 & p_3 \end{bmatrix}^T$ and $\mathbf{q}$, and

$$\boldsymbol{\eta} = E(:) = \begin{bmatrix} e_{11} & e_{21} & e_{31} & e_{12} & e_{22} & e_{32} & e_{13} & e_{23} & e_{33} \end{bmatrix}^T$$

is the stack of entries in E read by columns. Equation (13) can be replicated n times, one per image point pair, to yield a linear system

$$A\boldsymbol{\eta} = \mathbf{0} \quad \text{where} \quad A = \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_n \end{bmatrix}^T$$

is an $n \times 9$ matrix. The homogeneous nature of this system reflects the fact that translation $\mathbf{t}$ and therefore the essential matrix (see equation 7) is defined up to a nonzero multiplicative scale factor. To prevent the trivial solution $\boldsymbol{\eta} = \mathbf{0}$ and at the same time solve the system above in the least-squares sense to account for measurement inaccuracies, one computes

$$\boldsymbol{\eta} = \arg \min_{\|\boldsymbol{\eta}\|=1} A\boldsymbol{\eta} = \mathbf{v}_9 \quad \text{where} \quad A = U_A \Sigma_A V_A^T$$

is the Singular Value Decomposition (SVD) of A and $\mathbf{v}_9$ is the last column of V_A. The resulting vector $\boldsymbol{\eta}$ is then reshaped into an estimate E of the essential matrix.[4]

From equation (11) it follows immediately that the left null space of E is the one-dimensional space spanned by $\mathbf{t}$, which also spans the left null space of the skew matrix $[\mathbf{t}]_\times$. So an estimate of $\mathbf{t}$ is

$$\mathbf{t}_{1,2} = \pm \mathbf{u}_3 \quad \text{where} \quad E = U \Sigma V^T$$

is the SVD of E and $\mathbf{u}_3$ is the last column of U. The ambiguity in the sign of $\mathbf{t}$ will be resolved later.

Given $\mathbf{t}$, one can construct the skew matrix $[\mathbf{t}]_\times$, and then estimate R by solving the following *Procrustes problem* [1]:

$$E = [\mathbf{t}]_\times R \; . \tag{14}$$

If E and $[\mathbf{t}]_\times$ were full rank, the solution to problem (14) would be

$$R = C \det(C) \quad \text{where} \quad C = U_B V_B^T \quad \text{and} \quad B = U_B \Sigma_B V_B^T$$

is the SVD of the 3×3 matrix

$$B = [\mathbf{t}]_\times^T E \; , \tag{15}$$

and where the multiplication by $\det(C)$ ensures that the resulting orthogonal matrix is a rotation. This multiplication is allowed, because if E is an essential matrix then so is $-E$.

However, the two matrices E and $[\mathbf{t}]_\times$ have rank 2, and their third singular vectors (both left and right)—equal to the unit translation vectors $\mathbf{s}$ and $\mathbf{t}$ from the discussion above—are defined up to a sign. Because of this, the Procrustes problem has two solutions:

$$R_{1,2} = W_{1,2} \det(W_{1,2}) \quad \text{where} \quad W_{1,2} = \boldsymbol{\alpha}_1 \boldsymbol{\beta}_1^T + \boldsymbol{\alpha}_2 \boldsymbol{\beta}_2^T \pm \mathbf{t}\mathbf{s}^T$$

where

$$U_B = \begin{bmatrix} \boldsymbol{\alpha}_1 & \boldsymbol{\alpha}_2 & -\mathbf{t} \end{bmatrix} \quad , \quad V_B = \begin{bmatrix} \boldsymbol{\beta}_1 & \boldsymbol{\beta}_2 & \mathbf{s} \end{bmatrix}$$

[4] The two nonzero singular values of the essential matrix are equal to each other, and the matrix $\tilde{E}$ that satisfies this constraint and is closest to E in the Frobenius norm is $\tilde{E} = U \mathrm{diag}([1, 1, 0]) V^T$ where $E = U \Sigma V^T$ is the SVD of E. However, the singular values of $\tilde{E}$ are not needed in the computation that follows, so this correction is unnecessary.

(the minus sign for $\mathbf{t}$ comes from equation (3)). Equivalently, if U_B and V_B are first replaced by their rotation versions $U_B \det(U_B)$ and $V_B \det(V_B)$ (so that their determinants are equal to 1), we have

$$R_1 = \boldsymbol{\alpha}_1 \boldsymbol{\beta}_1^T + \boldsymbol{\alpha}_2 \boldsymbol{\beta}_2^T - \mathbf{t}\mathbf{s}^T \quad \text{and} \quad R_2 = -\boldsymbol{\alpha}_1 \boldsymbol{\beta}_1^T - \boldsymbol{\alpha}_2 \boldsymbol{\beta}_2^T - \mathbf{t}\mathbf{s}^T \ . \tag{16}$$

These equations reveal that R_1 and R_2 relate to each other through a 180-degree rotation of either camera reference frame around the baseline. To see this, write the transformation between these two frames of reference as a transformation from frame 1 to the world frame composed with one from world frame to frame 2:

$$R_2 R_1^T = (-\boldsymbol{\alpha}_1 \boldsymbol{\beta}_1^T - \boldsymbol{\alpha}_2 \boldsymbol{\beta}_2^T - \mathbf{t}\mathbf{s}^T)(\boldsymbol{\beta}_1 \boldsymbol{\alpha}_1^T + \boldsymbol{\beta}_2 \boldsymbol{\alpha}_2^T - \mathbf{s}\mathbf{t}^T) = -\boldsymbol{\alpha}_1 \boldsymbol{\alpha}_1^T - \boldsymbol{\alpha}_2 \boldsymbol{\alpha}_2^T + \mathbf{t}\mathbf{t}^T \ ,$$

and this rotation maps $\boldsymbol{\alpha}_1$ to $-\boldsymbol{\alpha}_1$, $\boldsymbol{\alpha}_2$ to $-\boldsymbol{\alpha}_2$, and $\mathbf{t}$ to itself, as promised.

Combining the twofold ambiguity in $\mathbf{t}$ with that in R yields four solutions, each corresponding to a different essential matrix:

$$(\mathbf{t}, R_1)\ ,\ (-\mathbf{t}, R_2)\ ,\ (\mathbf{t}, R_2)\ ,\ (-\mathbf{t}, R_1)\ .$$

The transformation between the first and the last of these solutions places camera 2 on the opposite side of camera 1 along the baseline.[5] This transformation can equivalently described as leaving the cameras where they are, pointing in the same way, but replacing all structure vectors $\mathbf{P}_i$ and $\mathbf{Q}_i$ by their opposites $-\mathbf{P}_i$ and $-\mathbf{Q}_i$. This transformation is said to change the *chirality* of structure in the literature [2], because superposing the original structure with the transformed one requires a change of handedness of the reference system (that is, a mirror flip). This transformation has the effect of placing the scene *behind* the two cameras if it is in front of them to begin with. With some abuse of terminology, a change of chirality in computer vision means merely changing whether structure is in front or behind a camera. In this sense, structure has two values of chirality, one per camera. A 180-degree rotation around the baseline—obtained by replacing R_1 with R_2 or *vice versa*—changes chirality once more, but only for the camera being rotated.

The four motion solutions above correspond to using top right, top left, bottom right, and bottom left camera pairs in Figure 2, in this order. The two top pairs in the figure are said to form a *twisted pair*, and so are the two bottom pairs.

The correct solution can then be identified by computing structure for all four cases by triangulation, and choosing the one solution that enforces structure to be in front of both cameras:

$$\mathbf{k}^T \mathbf{P}_i^e > 0 \quad \text{and} \quad \mathbf{k}^T \mathbf{Q}_i^e > 0 \quad \text{for} \quad i = 1, \ldots, n \quad \text{where} \quad \mathbf{k}^T = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$$

and where $\mathbf{P}_i^e$ and $\mathbf{Q}_i^e$ are the Euclidean coordinates corresponding to homogeneous coordinates $\mathbf{P}_i$ and $\mathbf{Q}_i$. Allowing for reconstruction errors, a safer approach is to chose the solution with a *majority* of points in front of the camera.

Since only the sign of the structure components is needed, a simple triangulation method will do. The projection equation (4) can be written as follows:

$$\alpha \mathbf{p} = \Pi \mathbf{P} \quad \text{and} \quad \beta \mathbf{q} = \Phi \mathbf{P} \quad \text{where} \quad \Pi = \Pi_0 \quad \text{and} \quad \Phi = \Pi_0 g$$

and vectors $\mathbf{p}$ and $\mathbf{q}$ can be normalized so their third components p_3 and q_3 are equal to 1. These equations can then be spelled out into their separate rows as follows:

$$\alpha p_1 = \boldsymbol{\pi}_1^T \mathbf{P} \quad , \quad \alpha p_2 = \boldsymbol{\pi}_2^T \mathbf{P} \quad , \quad \alpha = \boldsymbol{\pi}_3^T \mathbf{P} \quad , \quad \beta q_1 = \boldsymbol{\varphi}_1^T \mathbf{P} \quad , \quad \beta q_2 = \boldsymbol{\varphi}_2^T \mathbf{P} \quad , \quad \beta = \boldsymbol{\varphi}_3^T \mathbf{P} \quad .$$

[5] Of course, the same transformation can be described as a displacement of camera 1 relative to camera 2.

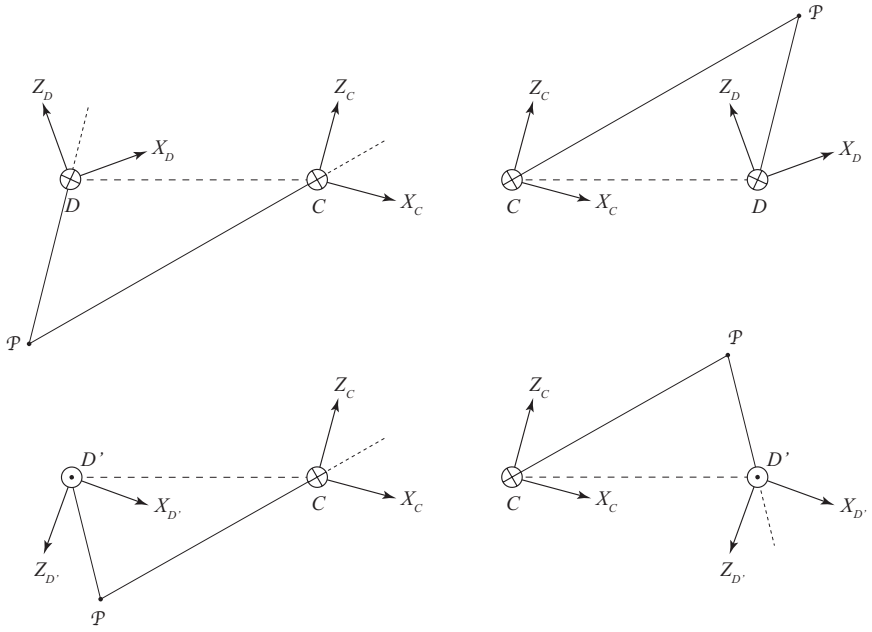

Figure 2: The fourfold ambiguity of reconstruction corresponds to the two ways to pick the sign of $\mathbf{t}$ (left or right diagrams) and the two ways to choose the rotation matrix R (top or bottom diagrams). A circle with a cross (a dot) denotes a Y axis pointing into (out of) the page. Only the arrangement in the top right has the scene structure (represented by the single point $\mathcal{P}$ and its two projection rays) in front of both cameras.

The expressions for α and β given by the third and sixth equation above can be replaced into the other equations to obtain the following 4×4 homogeneous linear system in $\mathbf{P}$:

$$G\mathbf{P} = \begin{bmatrix} p_1\boldsymbol{\pi}_3 - \boldsymbol{\pi}_1 \ , & p_2\boldsymbol{\pi}_3 - \boldsymbol{\pi}_2 \ , & q_1\boldsymbol{\varphi}_3 - \boldsymbol{\varphi}_1 \ , & q_2\boldsymbol{\pi}_3 - \boldsymbol{\varphi}_2 \end{bmatrix}^T \mathbf{P} = \mathbf{0} \ .$$

The solution $\mathbf{P}$ is the last right singular vector of G, and $\mathbf{Q}$ can be found by equation (1).

The complete MATLAB code for 3D reconstruction with two cameras is listed in Figures 3 and 4.

References

[1] G. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd edition, 1996.

[2] R. I. Hartley. Chirality. *International Journal of Computer Vision*, 26(1):41–61, 1998.

[3] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, September 1981.

[4] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment – A modern synthesis. In W. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, LNCS, pages 298–375. Springer Verlag, 2000.

```
function [g, P, Q]  = longuetHiggins(p, q)

n = size(p, 2);         % Number of point correspondences
if size(q, 2) ~= n
    error('The number of points in the two images must be the same');
end

% Estimate the essential matrix
A = zeros(n, 9);
for i = 1:n, A(i,:) = kron(p(:,i),q(:,i))'; end
if rank(A) < 8, error('Measurement matrix rank deficient'), end
[~, ~, V] = svd(A);
E = reshape(V(:,9), 3, 3);

% One of two choices for the translation vector (the other is -t)
[UE, ~, ~] = svd(E);
t = UE(:, 3);

% Two choices for the rotation matrix
tx = skew(t);
[UR, ~, VR] = svd(tx' * E);
R1 = UR * VR'; R1 = R1 * det(R1);
UR(:, 3) = -UR(:, 3);
R2 = UR * VR'; R2 = R2 * det(R2);

% Combine the two sign options for t with the two choices for R
t = [t, t, -t, -t];
R = cat(3, R1, R2, R1, R2);

% Pick the correct combination by enforcing positive chirality
npd = zeros(4, 1);
P = zeros(4, n, 4);
Q = zeros(4, n, 4);
for k = 1:4
    g = [R(:, :, k), t(:, k); 0 0 0 1];
    [P(:, :, k), Q(:, :, k)] = triangulate(p, q, g);
    npd(k) = sum(P(3, :, k) > 0 & Q(3, :, k) > 0);
end
[~, best] = max(npd);
g = [R(:, :, best), t(:, best); 0 0 0 1];
P = P(:, :, best);
Q = Q(:, :, best);
```

Figure 3: Main function of the MATLAB code for 3D reconstruction with two cameras.

```
function [P, Q] = triangulate(p, q, g)
    n = size(p, 2);
    Pi = [eye(3), zeros(3, 1)];
    Phi = Pi * g;
    P = zeros(4, n);
    for i=1:n
        G = [p(1, i) * Pi(3, :) - Pi(1, :);
             p(2, i) * Pi(3, :) - Pi(2, :);
             q(1, i) * Phi(3, :) - Phi(1, :);
             q(2, i) * Phi(3, :) - Phi(2, :)];

         [~, ~, v] = svd(G);
        P(:, i) = v(:,4);
    end
    Q = g * P;
    % Normalize the fourth coordinate to 1
    P = homogeneous(euclidean(P));
    Q = homogeneous(euclidean(Q));
end

function T = skew(t)
    T = [0 -t(3) t(2); t(3) 0 -t(1); -t(2) t(1) 0];
end

function h = homogeneous(e)
    h = [e; ones(1, size(e, 2))];
end

function e = euclidean(h)
    w = h(end, :);
    d = size(h, 1) - 1;
    e = h(1:d, :);
    nz = w ~= 0;
    e(:,  nz) = h(1:d, nz) ./ (ones(d, 1) * w(nz));
end
```

Figure 4: Auxiliary MATLAB functions for 3D reconstruction with two cameras.