Relative Camera Pose Recovery and Scene Reconstruction with the Essential Matrix in a Nutshell

Technical Report · March 2013

1 author:

George Terzakis
University of Portsmouth

Relative Camera Pose Recovery and Scene Reconstruction with the Essential Matrix in a Nutshell

George Terzakis

Centre for Robotics and Neural Systems

Technical Report#: MIDAS.SNMSE.2013.TR.007

Plymouth University, March 11, 2013

Abstract

This paper intends to present and analyze all the necessary steps one needs to take in order to build a simple application for relative pose structure from motion using only the epipolar constraint. In other words, to illustrate in practice (i.e,, cook-book style) about how the most fundamental of concepts in epipolar geometry can be combined into a complete and disambiguated series of steps for camera motion estimation and 3D reconstruction in a relatively simple computer program. In particular, the essential matrix is defined with respect to the algebraic interpretation of the rotation matrix, in order to eliminate ambiguities with respect to how relative orientation is extracted from the epipolar constraint. Based on the latter formulation, we list the prominent properties of the essential matrix with proofs to provide necessary intuition into the series of derivations that ultimately lead to the recovery of the orientation and baseline vector.

1. Introduction

The problem of 3D reconstruction and odometry estimations has been under the spotlight of research in computer vision and robotics for the past two decades. Since the early paper by Longuett – Higgins [1] which built a start-up theoretical layer on the original photogrammetric equations by introducing the epipolar constraint, there have been numerous studies and proposed algorithms to estimate camera motion and to realize the world locations of a sparse cloud of points [2-5]. Such problems are now known as structure from motion (SFM) problems and fall in the broader field of multiple view geometry in computer vision.

Although the geometry of multiple images has been extensively researched, structure from motion remains a vastly unexplored territory, mainly due to great informational bandwidth that images carry. The pipeline of SFM starts with the detection of corresponding points in two distinct images of the same scene. There have been several algorithms that match points across multiple views, known as optical flow estimation methods [6-8]. Although such methods are being extensively employed with a great deal of success in many cases, they are nevertheless relying on a number of strong assumptions such as brightness constancy, no occlusions and relatively slow camera motion. Limitations in point will consequently propagate a fair amount of noise throughout geometrical computations, thereby introducing unstable results in many cases.

2. The Epipolar Constraint

2.1 Moving Between Coordinate Frames

The goal of this section is to derive a formula for the coordinates of a 3D point M in a given camera frame located at a location b, given the orientation of the frame as three unit vectors (u_1, u_2, u_3), where b and u_1, u_2, u_3 are expressed in terms of the first camera frame (w_1, w_2, w_3) which is chosen for global reference. Figure D.1 illustrates the two frames, the baseline and the vectors that connect the two camera centers with the point M.

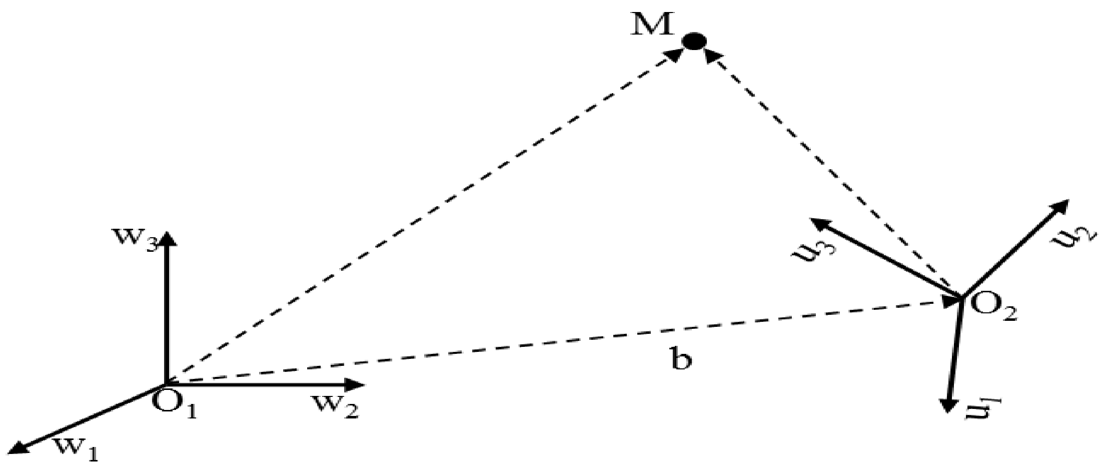

Figure 1. The two camera frames; the baseline and the vectors that connect the two camera centers O_1 and O_2 with M are indicated with dashed lines.

Let now $M_1 = M$ be the coordinates of M in the first camera frame and M_2 the respective coordinates in the second camera frame. Also, let $(O_2M)_1$ be the vector O_2M expressed in the coordinate frame of the first camera. Then, the coordinates of M_2 will be the projections of $(O_2M)_1$ on the direction vectors u_1, u_2, u_3:

$$M_2 = \begin{bmatrix} u_1^T(O_2M)_1 \\ u_2^T(O_2M)_1 \\ u_3^T(O_2M)_1 \end{bmatrix} = \begin{bmatrix} u_1^T \\ u_2^T \\ u_3^T \end{bmatrix} (O_2M)_1 = [u_1 \quad u_2 \quad u_3]^T (O_2M)_1 = R^T(O_2M)_1 \qquad (1)$$

But $(O_2M)_1$ can be written as the difference of vectors $(O_1M)_1 = M_1$ and b:

$$(O_2M)_1 = M - b \qquad (2)$$

Substituting from (1) into (2) yields:

$$M_2 = R^T(M_1 - b) \qquad (3)$$

2.2 The Essential Matrix

With the above in place, consider now the normalized Euclidean projections m_1 and m_2 of M. Epipolar geometry in two views concerns the geometry of the plane induced by the baseline and the projection rays that connect the real-world location of a feature with the two projection (camera) centers. Thus, for each pair of correspondences there exists a plane that contains the respective world point and the baseline vector. Figure 2 illustrates the epipolar plane of a pair of correspondences. The projections of the two camera centers in the first and second view are the epipoles e_1, e_2. The lines λ_1 and λ_2 defined by the epipoles and the normalized Euclidean projections m_1, m_2 are known as the epipolar lines; epipolar lines are in fact, the projections of the two rays that pass through the camera centers O_1, O_2 and the point M onto the opposite image planes.

Let now u_1 and u_2 be the direction vectors (in the first camera coordinate frame) of the projection rays that connect the first and second camera center with the point *M*. Evidently, u_1, u_2 and b span the *epipolar plane* that corresponds to *M*.

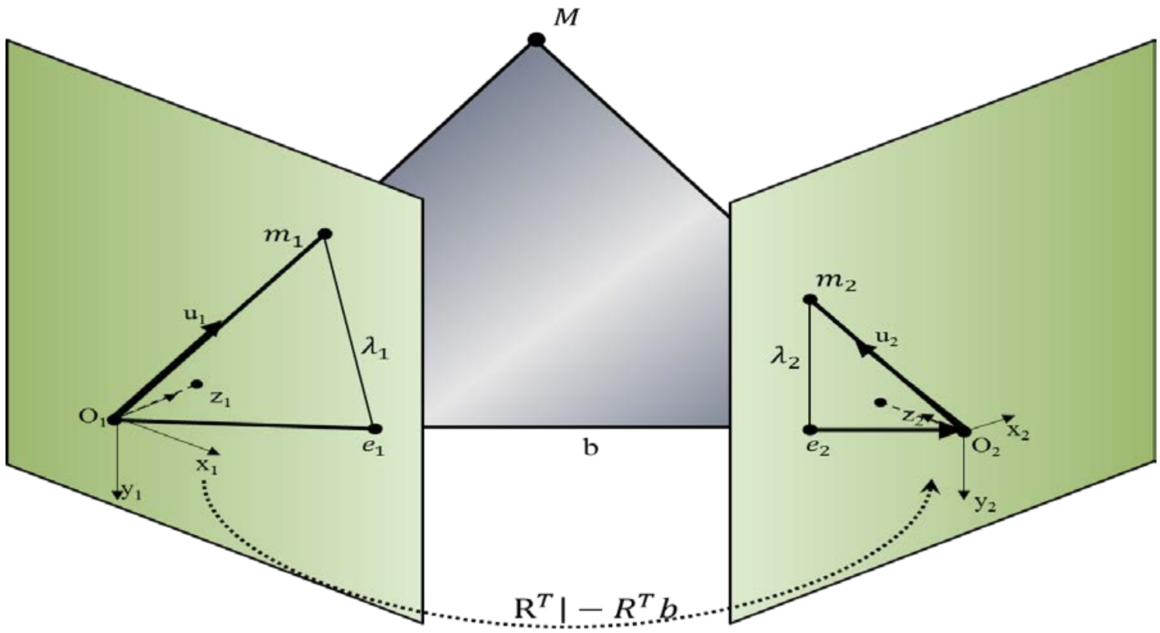

Figure 2. Epipolar plane induced by corresponding projections.

An equivalent way of expressing this coplanarity is by considering the orthogonality relationship between u_2 and the cross product of b and u_1:

$$u_2 \cdot (u_1 \times b) = 0 \tag{4}$$

where $\cdot$ is the inner product operator. Since $u_2 = u_1 - b = M_1 - b$ and $u_1 = Ru_2 + b = RM_2 + b$, substituting in equation (3.10) yields,

$$(M_1 - b)^T \big([b]_\times (RM_2 + b)\big) = 0 \tag{5}$$

where $[b]_\times$ is the cross product skew symmetric matrix of b. By simply applying the distributive law and taking into consideration the fact that $b^T [b]_\times = [b]_\times b = 0$, the following constraint is obtained:

$$M_1{}^T [b]_\times R M_2 = 0 \quad \Leftrightarrow \; M_2{}^T R^T [b]_\times M_1 = 0 \tag{6}$$

By definition, the normalized Euclidean projections are projectively equal to M_1 and M_2 and therefore the constraint can be re-expressed in terms of m_1 and m_2:

$$m_1{}^T ([b]_\times R) m_2 = 0 \quad \Leftrightarrow \; m_2{}^T (R^T [b]_\times) m_1 = 0 \tag{7}$$

The essential matrix. The 3×3 matrix $R^T [b]_\times$ is known in literature as the *essential matrix E*. Using the essential matrix notation, the epipolar constraint is obtained in its widely recognized form [1][1]:

$$m_2{}^T \underbrace{R^T [b]_\times}_{E} m_1 = 0 \tag{8}$$

[1] Evidently, there can be many essential matrices corresponding to the same set of 2-view projections depending on the interpretation given to the rigid transformation that links the two camera poses. In this thesis, the rotation matrix R is perceived as the matrix that contains the second camera frame directions (expressed in the first camera frame) as its columns and b is the baseline vector (also in the first camera coordinate frame): Thus, the essential matrix will always be given by $E = R^T [b]_\times$.

4. Properties of the essential matrix

The most important properties of the essential matrix are listed in this section. These properties are very useful either as constraints or as supplementary formulas in the course of scene structure and relative camera pose estimation.

Lemma 1. *Suppose E is an essential matrix. Then, the epipoles e_1 and e_2 are the left and right null spaces of E respectively.*

Proof. Since the epipoles are the scaled images of the camera centers, then there exist λ_1 and λ_2 such that $e_2 = -\lambda_2 R^T b$ and $e_1 = \lambda_1 b$. By the expression of the essential matrix in equation (D.1), $E = R^T[b]_\times$. Therefore:

$$e_2^T E = -\lambda_2 (R^T b)^T R^T [b]_\times = -\lambda_2 b^T \underbrace{R R^T}_{I_3} [b]_\times = -\lambda_2 \underbrace{b^T [b]_\times}_{-b\times b=0} = 0 \tag{9}$$

Also,

$$E e_1 = \lambda_2 R^T \underbrace{[b]_\times b}_{b\times b=0} = 0 \tag{10}$$

Lemma 2. *Any point m_1, m_2 has an associated epipolar line l_1, l_2 in the opposite view given by:*

$$l_2 = E m_1 = e_2 \times m_2 \qquad l_1 = E m_2 = e_1 \times m_1$$

Proof. The proof is a direct consequence of the epipolar constraint for m_1 and m_2.

Lemma *3.* For any orthonormal matrix $R \in \mathbb{R}^{3\times 3}$ and for any vector $a \in \mathbb{R}^3$ the following holds:

$$[Ra]_\times = R[a]_\times R^T \tag{11}$$

Proof. Let *R* be an orthonormal matrix. Then by the definition of cross-product,

$$(Ra) \times b = [Ra]_\times b \tag{12}$$

Multiplying (D.8) with R^T from the left yields,

$$R^T\big((Ra) \times b\big) = R^T [Ra]_\times b \tag{13}$$

We now resort to the following property of the cross product that holds for any linear transformation *M* and any pair of vectors *a*, *b*:

$$(Ma) \times (Mb) = M(a \times b) \tag{14}$$

Making use of the cross product property in (14), equation (13) becomes:

$$(R^T R)a \times (R^T b) = R^T [Ra]_\times b$$

$$\Leftrightarrow a \times (R^T b) = R^T [Ra]_\times b$$

$$\Leftrightarrow [a]_\times R^T b = R^T [Ra]_\times b \ \Leftrightarrow ([a]_\times R^T - R^T [Ra]_\times) b = 0$$

$$\Leftrightarrow [a]_\times = R^T [Ra]_\times R \tag{15}$$

A very significant theorem that provides a simple but hard criterion for the existence of an essential matrix is the following (Faugeras, Luong et al. 2004).

Theorem 1. *A $3x3$ matrix E is an essential matrix if and only if it has a singular value decomposition $E = USV^T$ such that:*

$$S = diag\{\sigma, \sigma, 0\}, \qquad \sigma > 0$$

where U, V are orthonormal matrices.

Proof. Let E be a fundamental matrix. Then, E is given by, $E = R^T[b]_\times$. Taking E^TE yields,

$$E^TE = (R^T[b]_\times)^T R^T[b]_\times = [b]_\times{}^T RR^TR^T[b]_\times = [b]_\times{}^T[b]_\times \tag{16}$$

Let now R_0 be the rotation that aligns the baseline vector with the z-axis, that is, $R_0b = [0 \quad 0 \quad \|b\|]^T$. Then, using lemma 3, it follows that $[a]_\times = R_0^T[R_0b]_\times R_0$. Substituting in (D.7) yields,

$$E^TE = \left(R_0{}^T[R_0a]_\times R_0\right)^T R_0{}^T[R_0a]_\times R_0 = R_0{}^T[R_0a]_\times{}^T[R_0a]_\times[R_0a]_\times R_0$$

$$\Leftrightarrow E^TE = R_0{}^T\begin{bmatrix} 0 & \|b\| & 0 \\ -\|b\| & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} 0 & -\|b\| & 0 \\ \|b\| & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} R_0$$

$$\Leftrightarrow E^TE = R_0{}^T\begin{bmatrix} \|b\|^2 & 0 & 0 \\ 0 & \|b\|^2 & 0 \\ 0 & 0 & 0 \end{bmatrix} R_0 \tag{17}$$

The decomposition of equation (17) is a SVD (one out many) of E^TE. Hence, E will also decompose as follows:

$$E = U\begin{bmatrix} \|b\| & 0 & 0 \\ 0 & \|b\| & 0 \\ 0 & 0 & 0 \end{bmatrix} R_0 \tag{18}$$

for some orthonormal matrix *U*.

Let now *E* be a $3x3$ matrix with two exactly non-zero singular values which are equal. In this case, it is easier to proceed by gradually constructing the sought result. The decomposition of the matrix *E* is,

$$E = U\begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T = USV^T \tag{19}$$

where *U* and *V* are orthonormal matrices. The construction that follows is relying on the observation that *S* can be written in the following way:

$$S = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} \sigma & 0 & 0 \\ 0 & \sigma & 0 \\ 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} = R_z\left(-\frac{\pi}{2}\right) SR_z\left(-\frac{\pi}{2}\right)^T \tag{20}$$

or,

$$S = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} \sigma & 0 & 0 \\ 0 & \sigma & 0 \\ 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} = R_z\left(\frac{\pi}{2}\right) SR_z\left(\frac{\pi}{2}\right)^T \tag{21}$$

where $R_z\left(-\frac{\pi}{2}\right)$ and $R_z\left(\frac{\pi}{2}\right)$ are rotations by $\pm\pi/2$ about the z axis. We observed that the product of S with the preceding or following rotation yields a skew symmetric matrix:

$$S = \underbrace{\left(R_z\left(\frac{\pi}{2}\right)S\right)}_{skew\ symmetric} R_z\left(\frac{\pi}{2}\right)^T = \begin{bmatrix} 0 & -\sigma & 0 \\ \sigma & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} R_z\left(\frac{\pi}{2}\right)^T \tag{22}$$

or,

$$S = R_z\left(-\frac{\pi}{2}\right)\underbrace{\left(SR_z\left(-\frac{\pi}{2}\right)^T\right)}_{skew\ symmetric} = \begin{bmatrix} 0 & \sigma & 0 \\ -\sigma & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} R_z\left(-\frac{\pi}{2}\right)^T \tag{23}$$

Now, the property of skew symmetric matrices in lemma 3 can be of great use in a heuristic sense. In particular, it guarantees that for any rotation matrix U and for any skew symmetric matrix $S_\times$, the matrix $US_\times U^T$ is also a skew symmetric matrix. In the light of this consequence and choosing (22), the SVD of E can be expressed as follows:

$$E = UR_z\left(-\frac{\pi}{2}\right)\begin{bmatrix} 0 & -\sigma & 0 \\ \sigma & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T$$

$$= U \underbrace{R_z\left(-\frac{\pi}{2}\right)^T V^T V R_z\left(-\frac{\pi}{2}\right)}_{equal\ to\ identity} \begin{bmatrix} 0 & -\sigma & 0 \\ \sigma & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} R_z\left(-\frac{\pi}{2}\right)^T V^T$$

$$\Leftrightarrow E = \underbrace{\left(UR_z\left(-\frac{\pi}{2}\right)^T V^T\right)}_{a\ rotation\ matrix} \underbrace{\left(VR_z\left(-\frac{\pi}{2}\right)\right)\begin{bmatrix} 0 & -\sigma & 0 \\ \sigma & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}\left(VR_z\left(-\frac{\pi}{2}\right)\right)^T}_{a\ skew\ symmetric\ matrix\ by\ virtue\ of\ lemma\ D.3} \tag{24}$$

In the very same way, one arrives at a similar result starting from (D.14):

$$E = \underbrace{\left(UR_z\left(\frac{\pi}{2}\right)^T V^T\right)}_{a\ rotation\ matrix} \underbrace{\left(VR_z\left(\frac{\pi}{2}\right)\right)\begin{bmatrix} 0 & \sigma & 0 \\ -\sigma & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}\left(VR_z\left(\frac{\pi}{2}\right)\right)^T}_{a\ skew\ symmetric\ matrix\ by\ virtue\ of\ lemma\ D.3} \tag{25}$$

The only remaining "loose end" now is to show that matrices $UR_z\left(-\frac{\pi}{2}\right)^T V^T$ and $UR_z\left(\frac{\pi}{2}\right)^T V^T$ are indeed rotation matrices. Since the product involves both U and V, then the sign of the product of their determinants should be positive due to the fact that these matrices participate in the SVD of E^TE which always has a positive determinant. And since the determinant of R_z is also positive, it follows that the two aforementioned products are orthonormal matrices with positive determinants, hence rotations. And that concludes the proof.

Lemma 4. *If E is an essential matrix such that $E = R^T[b]_\times$, then:*

$$Tr(EE^T) = Tr(E^TE) = 2\|b\|^2$$

Proof. Taking $E^T E$ yields:

$$E^T E = (R^T [b]_\times)^T R^T [b]_\times = -{[b]_\times}^2 \tag{26}$$

If now $b = [b_1 \quad b_2 \quad b_3]^T$, then,

$$E^T E = -\begin{bmatrix} 0 & -b_3 & b_2 \\ b_3 & 0 & -b_1 \\ -b_2 & b_1 & 0 \end{bmatrix}^2 = \begin{bmatrix} {b_2}^2 + {b_3}^2 & -b_1 b_2 & -b_1 b_3 \\ -b_1 b_2 & {b_1}^2 + {b_3}^2 & -b_2 b_3 \\ -b_1 b_3 & -b_2 b_3 & {b_1}^2 + {b_2}^2 \end{bmatrix} \tag{27}$$

From (D.18) it is clear that $Tr(E^T E) = 2(b_1^2 + b_2^2 + b_3^2) = 2\|b\|^2$.

In the case of EE^T, the following is obtained:

$$EE^T = R^T [b]_\times (R^T [b]_\times)^T = -R^T {[b]_\times}^2 R$$
$$\Leftrightarrow EE^T = (R^T [b]_\times R)(R^T [b]_\times R) \tag{28}$$

Once again, lemma 3 states that $[b]_\times = R^T [Rb]_\times R$ for any orthogonal matrix R. Hence, (28) becomes:

$$E^T E = \left(R^T \underbrace{(R[b]_\times R^T)}_{[b]_\times} R \right) \left(R^T \underbrace{(R[b]_\times R^T)}_{[b]_\times} R \right) = -{[b]_\times}^2 \tag{29}$$

And since $\|R^T b\| = \|b\|$, it follows from (27) that $Tr(EE^T) = 2\|b\|^2$.

Theorem 5. A 3×3 non-zero matrix E is an essential matrix if and only if the following relationship holds:

$$EE^T E = \frac{Tr(E^T E)}{2} E$$

Proof. Proving that if the relationship holds then E is an essential matrix could be done through its SVD. Let $E = USV^T$ where U, V^T are orthonormal matrices and S is a diagonal matrix with positive entries. Substituting in the given relationship, yields:

$$EE^T E - \frac{Tr(EE^T)}{2} E = US^2 U^T USV^T - \frac{Tr(EE^T)}{2} USV^T = 0$$
$$\Leftrightarrow S^2 = \frac{Tr(EE^T)}{2} S \tag{30}$$

Let s_1, s_2, s_2 be the singular values of S. Then, the trace of EE^T should be equal to the sum of the squared singular values:

$$Tr(EE^T) = s_1^2 + s_2^2 + s_3^2 \tag{31}$$

It follows from (30) and (31) that,

$$\begin{aligned} 2s_1^3 &= (s_1^2 + s_2^2 + s_3^2) s_1 \\ 2s_2^3 &= (s_1^2 + s_2^2 + s_3^2) s_2 \\ 2s_3^3 &= (s_1^2 + s_2^2 + s_3^2) s_1 \end{aligned} \tag{32}$$

Since E is non-zero, one singular value must be strictly positive. Without constraining generality, let s_1 be strictly positive. It follows that,

$$s_1^2 = \frac{(s_1^2 + s_2^2 + s_3^2)}{2} \Leftrightarrow s_1^2 = s_1^2 + s_2^2 \quad (33)$$

Substituting in the expression for the second singular value in (32) yields:

$$s_2^3 = (s_2^2 + s_3^2)s_2 \Leftrightarrow s_2 s_3^2 = 0 \quad (34)$$

It follows that either s_2 or s_3 is zero. If they are both zero, then so is s_1 which is a contradiction. Exactly 2 of the 3 singular values are non-zero and have the same value, $s_1 = s_2 = \frac{Tr(EE^T)}{2}$.

Consider now the case in we which we known that E is an essential matrix. Taking the given relationship and substituting from (D.18) and using the fact that $[b]_\times^2 = bb^T - \|b\|^2 I$, we have:

$$\begin{aligned} E \underbrace{E^T E}_{-[b]_\times^2} &= -E \underbrace{[b]_\times^2}_{bb^T - \|b\|^2 I} = \|b\|^2 E - \underbrace{E}_{R^T[b]_\times} bb^T = \|b\|^2 E - R^T \underbrace{[b]_\times b}_{0_{3\times1}} b^T \\ &= \underbrace{\|b\|^2}_{\frac{Tr(E^T E)}{2}} E = \frac{Tr(E^T E)}{2} E \end{aligned} \quad (35)$$

where $0_{3\times1}$ is the zero 3×1 vector. And that concludes the proof in the opposite direction.

5. Recovering baseline and orientation

The method for relative pose extraction detailed in this section is based on the brilliant observation by Berthold Horn (Horn 1990) that the matrix of cofactors of an essential matrix can be expressed in terms of the rotation matrix, the skew symmetric matrix of the baseline and the essential matrix itself.

Consider a 3D point M and its normalized Euclidean projections m_1 and m_2 in two views as show in Figure D.1. With the essential matrix in place, the next step is to obtain the rotation matrix R and the unit-length baseline vector b. As a first step, from theorem D.1, scale can be removed from the essential matrix by dividing it with $\|b\|$. Also, lemma D.4 states that $Tr(E^T E) = 2\|b\|^2$; thus, a “normalized” essential matrix E_n is obtained as follows:

$$E_n = \frac{E}{\sqrt{\frac{Tr(E^T E)}{2}}} \quad (36)$$

5.1 Baseline

From lemma 4 it is easy to extract the absolute values of the baseline components as follows:

$$|b_1| = \sqrt{1 - [E_n^T E_n]_{11}} \quad (37)$$

$$|b_2| = \sqrt{1 - [E_n^T E_n]_{22}} \tag{38}$$

$$|b_3| = \sqrt{1 - [E_n^T E_n]_{33}} \tag{39}$$

where $[E_n^T E_n]_{ij}$ denotes the element of $E_n^T E_n$ in the i^{th} row and j^{th} column. To resolve the sign ambiguity, the largest squared component is assumed to be a positive square root and the remaining signs are inferred from the off-diagonal elements of $E_n^T E_n$:

$$E_n^T E_n = \begin{bmatrix} b_2^2 + b_3^2 & -b_1 b_2 & -b_1 b_3 \\ -b_1 b_2 & b_1^2 + b_3^2 & -b_2 b_3 \\ -b_1 b_3 & -b_2 b_3 & b_1^2 + b_2^2 \end{bmatrix} \tag{40}$$

It suffices to recover one baseline vector from $E_n^T E_n$ as described above, as the second baseline will simply be a vector of opposite direction.

5.2 Orientation

Recovering the rotation matrix requires slightly more elaborate pre-processing. Consider the matrix of cofactors of E_n:

$$C_n = \begin{bmatrix} \begin{vmatrix} e_{22} & e_{23} \\ e_{32} & e_{33} \end{vmatrix} & -\begin{vmatrix} e_{21} & e_{23} \\ e_{31} & e_{33} \end{vmatrix} & \begin{vmatrix} e_{21} & e_{22} \\ e_{31} & e_{32} \end{vmatrix} \\ -\begin{vmatrix} e_{12} & e_{13} \\ e_{32} & e_{33} \end{vmatrix} & \begin{vmatrix} e_{11} & e_{13} \\ e_{31} & e_{33} \end{vmatrix} & -\begin{vmatrix} e_{11} & e_{12} \\ e_{31} & e_{32} \end{vmatrix} \\ \begin{vmatrix} e_{12} & e_{13} \\ e_{22} & e_{23} \end{vmatrix} & -\begin{vmatrix} e_{11} & e_{13} \\ e_{21} & e_{23} \end{vmatrix} & \begin{vmatrix} e_{11} & e_{12} \\ e_{21} & e_{22} \end{vmatrix} \end{bmatrix} \tag{41}$$

Standard tensor notation is adopted to denote matrix rows and columns as well as elements for the following derivations. Thus, for instance, e_j^i is the element of E_n in the i^{th} row and the j^{th} column. Also, e^i is the i^{th} row of E_n as a 1×3 vector, while e_j is the j^{th} column as a 3×1 vector. With notation in place, we observe that C_n can be written as follows:

$$C_n = \begin{bmatrix} ((e^2)^T \times (e^3)^T)^T \\ ((e^3)^T \times (e^1)^T)^T \\ ((e^1)^T \times (e^2)^T)^T \end{bmatrix} \tag{42}$$

Also, E_n can be expressed in terms of cross-products as follows:

$$E_n = R^T [b]_\times = \begin{bmatrix} {r_1}^T \\ {r_2}^T \\ {r_3}^T \end{bmatrix} [b]_\times = \begin{bmatrix} {r_1}^T [b]_\times \\ {r_2}^T [b]_\times \\ {r_3}^T [b]_\times \end{bmatrix} = \begin{bmatrix} -([b]_\times r_1)^T \\ -([b]_\times r_2)^T \\ -([b]_\times r_3)^T \end{bmatrix} = \begin{bmatrix} -(b \times r_1)^T \\ -(b \times r_2)^T \\ -(b \times r_3)^T \end{bmatrix} \tag{43}$$

Substituting from (43) in (42) yields triple products in the rows of C_n; applying the well-known *triple product expansion* formula leaves cross product expressions only between the columns of *R* (intermediate result) which can also be eliminated from the expression by orthonormality (final expression on the right):

$$C_n = \begin{bmatrix} \left((b \times r_2) \times (b \times r_3)\right)^T \\ \left((b \times r_3) \times (b \times r_1)\right)^T \\ \left((b \times r_1) \times (b \times r_2)\right)^T \end{bmatrix} = \begin{bmatrix} \left(b \cdot \underbrace{(r_2 \times r_3)}_{r_1}\right) b^T \\ \left(b \cdot \underbrace{(r_3 \times r_1)}_{r_2}\right) b^T \\ \left(b \cdot \underbrace{(r_1 \times r_2)}_{r_3}\right) b^T \end{bmatrix} = \begin{bmatrix} (b \cdot r_1) b^T \\ (b \cdot r_2) b^T \\ (b \cdot r_3) b^T \end{bmatrix} \tag{44}$$

It is now easy to re-arrange the inner products in (44) in order to factor-out the columns of *R*:

$$C_n = \begin{bmatrix} ({r_1}^T b) b^T \\ ({r_2}^T b) b^T \\ ({r_3}^T b) b^T \end{bmatrix} = \begin{bmatrix} {r_1}^T (b b^T) \\ {r_2}^T (b b^T) \\ {r_3}^T (b b^T) \end{bmatrix} = R^T (b b^T) \tag{45}$$

And a well-known skew symmetric matrix property is that,

$$b b^T = I_3 + [b]_\times^2 \tag{46}$$

Substituting (46) into (45) yields:

$$C_n = R^T\left(I_3 + [b]_\times^2\right) = R^T + \underbrace{\left(R^T [b]_\times\right)}_{E_n} [b]_\times = R^T + E_n [b]_\times \Leftrightarrow R = C_n^T + [b]_\times E_n^T \tag{47}$$

And since b is sign-ambiguous, it follows that there exist two possible rotation matrices and can be obtaining by flipping the sign of $[b]_\times$ in (D.38):

$$R = C_n^T \pm [b]_\times E_n^T \tag{48}$$

5.3 Resolving ambiguities in the baseline-rotation solution

From an algebraic point of view, it is clear that, given an essential matrix E, a very same baseline vector can be obtained either from E, or from $-E$, since $E^T E = (-E)^T(-E)$. Clearly, this ambiguity reflects the direction of the baseline, since the negative sign can only originate from $[b]_\times$.

The four possible relative pose configurations recovered from the essential matrix are illustrated in Figures 3 and 4 in terms of the location of the observed points. It is clear that only one combination of the two baselines and two rotation matrices aligns both cameras behind the observed point. This suggests that, in order to resolve the ambiguity between the four solutions, a reconstruction of the scene must be obtained and the transformation that yields positive (or negative) signs in both depths of an observed point as observed from the two camera views should be the correct one. In other words, the point should lie in front of both cameras. With noisy data, this may not be the case for all points even for the correct transformation and therefore in practice this is a matter of voting.

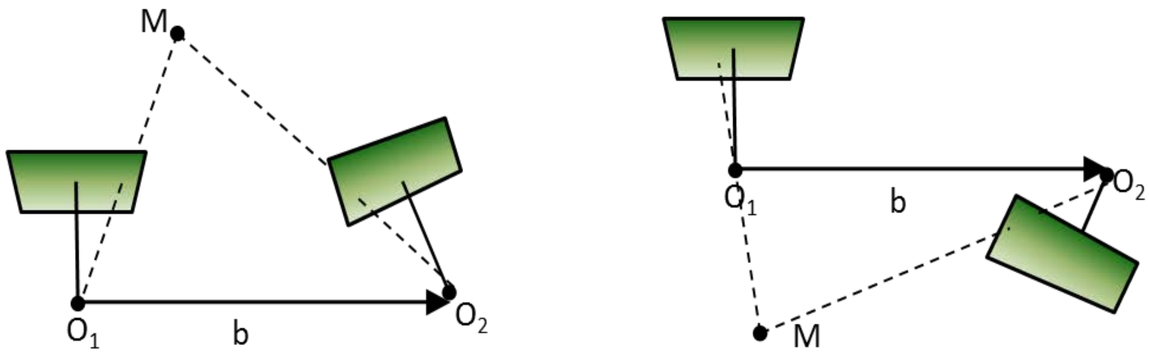

Figure 3. Camera orientation with respect to the observed points for "positive" baseline direction.

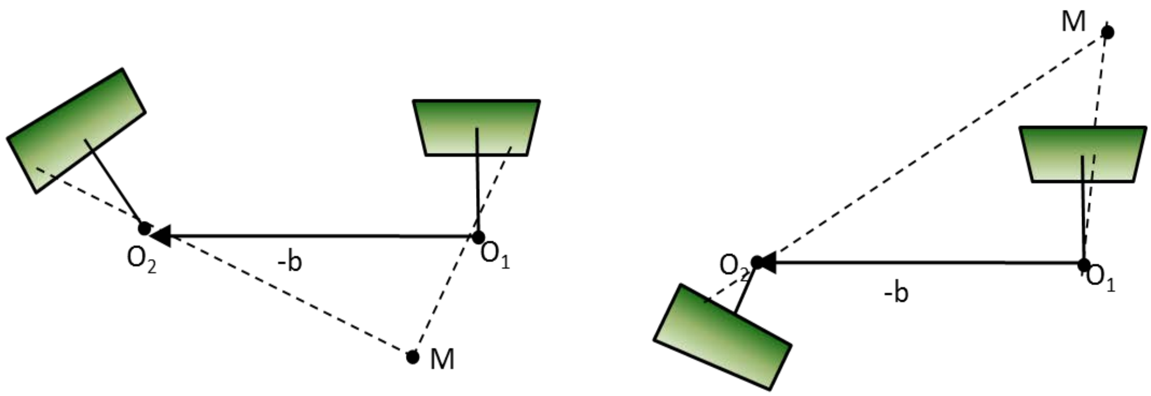

Figure 4. Camera orientation with respect to the observed points for "negative" baseline direction.

6. Scene Reconstruction

In this section, a general method for 3D scene reconstruction in two views from known relative pose and correspondences is presented. This method is also used to disambiguate the four relative pose solutions extracted from the essential matrix.

Let $M = [X \quad Y \quad Z]^T$ be the real-world location of an observed point expressed in the first camera coordinate frame and let m_1 and m_2 be the respective normalized Euclidean projections in the two camera views. Also, let b denote the baseline vector in the coordinate frame of the first camera and R be the orthonormal matrix containing the directions of the second camera frame (expressed in the first camera coordinate frame) arranged column-wise. It follows that M can be expressed in terms of m_1 as follows:

$$M = Zm_1 \tag{49}$$

The location of M in the second camera frame is then $R^T(M - b)$. The normalized Euclidean coordinates of M are given by the following:

$$m_2 = \frac{1}{1_z^T R^T (M - b)} R^T (M - b) \tag{50}$$

where $1_z = [0 \quad 0 \quad 1]^T$. Substituting (49) into (50) yields a relationship in which the only unknown is the depth Z:

$$\left(1_z^T R^T (Zm_1 - b)\right) m_2 = R^T (Zm_1 - b)$$
$$\Leftrightarrow \left(1_z^T R^T (Zm_1 - b)\right) m_2 - R^T (Zm_1 - b) = 0 \qquad (51)$$

For any vectors a, b, c of arbitrary dimension, it is easy to prove that $(a^T b)c = (ca^T)b$. With this identity at hand, the relationship in (3.27) becomes:

$$(m_2 1_z^T) R^T (Zm_1 - b) - R^T (Zm_1 - b) = 0$$
$$\Leftrightarrow (m_2 1_z^T - I_3) R^T (Zm_1 - b) = 0 \qquad (52)$$

where I_3 is the 3×3 identity matrix. Equation (52) is an over-determined system in Z and yields two solutions, one for each projection component, provided that the measurements are completely noise-free. However, in most cases the two solutions do not agree and, furthermore, we observed that in the majority of these cases, disparity tends to concentrate either on the *x* or on the *y* axis, thereby making one solution more "reliable" than the other. The proposed workaround is to regard Z as a minimizer of the following optimization problem:

$$\underset{Z}{minimize}\{(Zm_1 - b)^T RCR^T (Zm_1 - b)\} \qquad (53)$$

where C is the following non-invertible positive semi-definite (PSD) matrix:

$$C = (m_2 1_z^T - I_3)^T (m_2 1_z^T - I_3) = \begin{bmatrix} 1 & 0 & -x_2 \\ 0 & 1 & -y_2 \\ -x_2 & -y_2 & x_2^2 + y_2^2 \end{bmatrix} \qquad (54)$$

and x_2 and y_2 are the coordinates of m_2 in the directions of the local *x* and *y* axis respectively. Taking the derivative of the quadratic expression in (53) in terms of Z and setting it to zero, yields the following minimizer:

$$Z = \frac{m_1^T RCR^T b}{m_1^T RCR^T m_1} \qquad (55)$$

The expression in (55) is a robust depth estimate which takes the direction of disparity into consideration thereby avoiding the "pitfall" of having to choose between two solutions for depth without any criteria at hand on how to make that choice. It should be however noted that estimation can yield very erroneous (e.g., negative depth) results if disparity is very noisy in both axes; in such a case, it is preferable to discard the point.

6. The Algorithm and Sample Reconstructions

Examples of 2-view reconstructions using the methods described in the previous sections are illustrated in Figures 5-8. Features were detected with SIFT and the LK tracker was used to establish correspondences. Flow fields are shown on the images on the right (in green are new features detected in the second image for subsequent tracking). RANSAC outliers were omitted from the reconstruction and do not appear in the flow field illustration. Finally, since camera intrinsics were unknown in all cases, the reconstructions present a discrepancy up to an affine transformation with the ground truth (made-up intrinsics were used). Algorithm 1 describes the steps for relative pose and structure recovery from an essential matrix.

Algorith 1. ***3D Reconstruction and recovery of relative pose from two views***

Input*: a) Set of normalized Euclidean correspondences* $m_1{}^{(i)}$ *and* $m_2{}^{(i)}$ *b) Essential matrix, E.*

Output*: a) Camera relative pose* (R, b)*, b) 3D coordinates of all points* $M^{(i)}$*.*

comment *Obtain the SVD of E:*

$[\mathrm{U}, \mathrm{S}, \mathrm{V}] \leftarrow \mathrm{svd}(\mathrm{E})$

comment *Obtain a "normalized essential matrix" by removing scale and at the same time impose the necessary (and capable) condition of exactly two and equal singular values:*

$$\mathrm{E_n} \leftarrow \mathrm{U}\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 0\end{bmatrix}\mathrm{V^T}$$

$\mathrm{AcceptReconstruction} \leftarrow \mathrm{False.}$

comment *Compute the absolute values of the baseline components* b_1*,* b_2*,* b_3 *from* $E_n^T E_n$

$\mathrm{b_1} \leftarrow \sqrt{1 - [\mathrm{E_n^T E_n}]_{11}}$

$\mathrm{b_2} \leftarrow \sqrt{1 - [\mathrm{E_n^T E_n}]_{22}}$

$\mathrm{b_3} \leftarrow \sqrt{1 - [\mathrm{E_n^T E_n}]_{33}}$

comment *Choosing the greatest component (in absolute value) as positive and work-out the remaining signs from the off-diagonal elements of the essential matrix*

If ($\max\{\mathrm{b_1}, \mathrm{b_2}, \mathrm{b_3}\} = \mathrm{b_1}$):

  If ($[\mathrm{E_n^T E_n}]_{12} > 0$)

    $b_2 \leftarrow -b_2$

  If ($[\mathrm{E_n^T E_n}]_{13} > 0$):

    $b_3 \leftarrow -b_3$

Else If ($\max\{\mathrm{b_1}, \mathrm{b_2}, \mathrm{b_3}\} = \mathrm{b_2}$):

  If ($[\mathrm{E_n^T E_n}]_{12} > 0$):

    $b_1 \leftarrow -b_1$

  If ($[\mathrm{E_n^T E_n}]_{23} > 0$):

    $\mathrm{b_3} \leftarrow -\mathrm{b_3}$

Else:

  If ($[\mathrm{E_n^T E_n}]_{13} > 0$):

    $\mathrm{b_1} \leftarrow -\mathrm{b_1}$

If $([E_n^T E_n]_{23} > 0)$:

$b_2 \leftarrow -b_2$

comment *Storing the two possible baselines*

$\mathrm{Baselines} \leftarrow \{(b_1, b_2, b_3) \quad , \quad (-b_1, -b_2, -b_3)\}$

comment *Find the matrix of cofactors of E_n*

$$C_n \leftarrow \begin{bmatrix} e_{22}e_{33} - e_{32}e_{23} & -(e_{21}e_{33} - e_{31}e_{23}) & e_{21}e_{32} - e_{31}e_{22} \\ -(e_{12}e_{33} - e_{32}e_{13}) & e_{11}e_{33} - e_{31}e_{13} & -(e_{11}e_{32} - e_{31}e_{12}) \\ e_{12}e_{23} - e_{22}e_{13} & -(e_{11}e_{23} - e_{21}e_{13}) & e_{11}e_{22} - e_{21}e_{12} \end{bmatrix}$$

comment *Store the two possible rotation matrices*

$\mathrm{Rotations} \leftarrow \{C_n^T + [\mathrm{Baselines}(1)]_\times E_n^T \quad , \quad C_n^T + [\mathrm{Baselines}(2)]_\times E_n^T\}$

comment *Find the best scene reconstruction for the 4 possible relative poses*

$\mathrm{BestReconstruction} \leftarrow \{\mathrm{Rotations}(1)\,, \mathrm{Baselines}(1)\}$

$\mathrm{MinCount} \leftarrow \infty$

For each baseline b **in** Baselines:

For each rotation matrix *R* **in** Rotations:

$\mathrm{errorCount} \leftarrow 0$

For each pair of correspondences (m_1, m_2):

$C = (m_2 1_3^T - I_3)^T (m_2 1_3^T - I_3)$

$$Z_1 \leftarrow \frac{m_1^T R C R^T b}{m_1^T R C R^T m_1}$$

$Z_2 \leftarrow 1_z^T R^T (Z_1 m_1 - b)$

If $(Z_1 \leq 0)$ **Or** $(Z_2 \leq 0)$:

$\mathrm{errorCount} \leftarrow \mathrm{errorCount} + 1$

Else:

$M = Z_1 m_1$

If $(\mathrm{errorCount} < \mathrm{minCount})$:

$\mathrm{BestReconstruction} \leftarrow \{R\,, b\}$

$\mathrm{minCount} \leftarrow \mathrm{errorCount}$

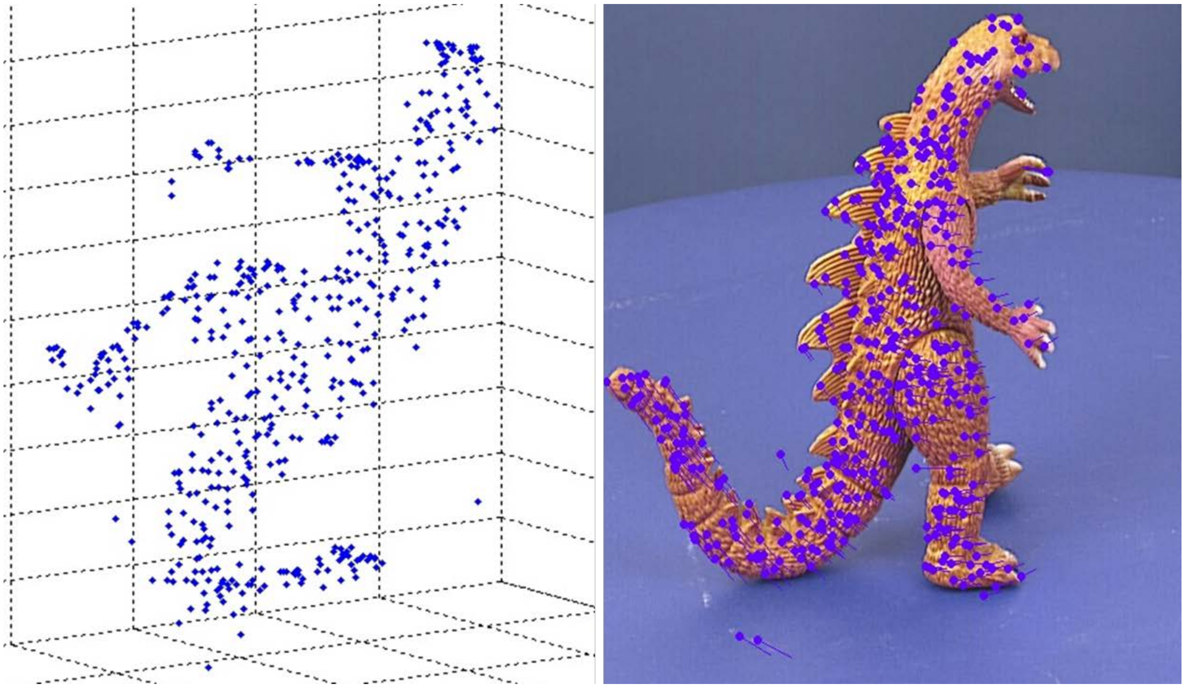

Figure 5. A reconstruction of the famous Hannover dinosaur (Niem and Buschmann 1995) from the first two views of the sequence.

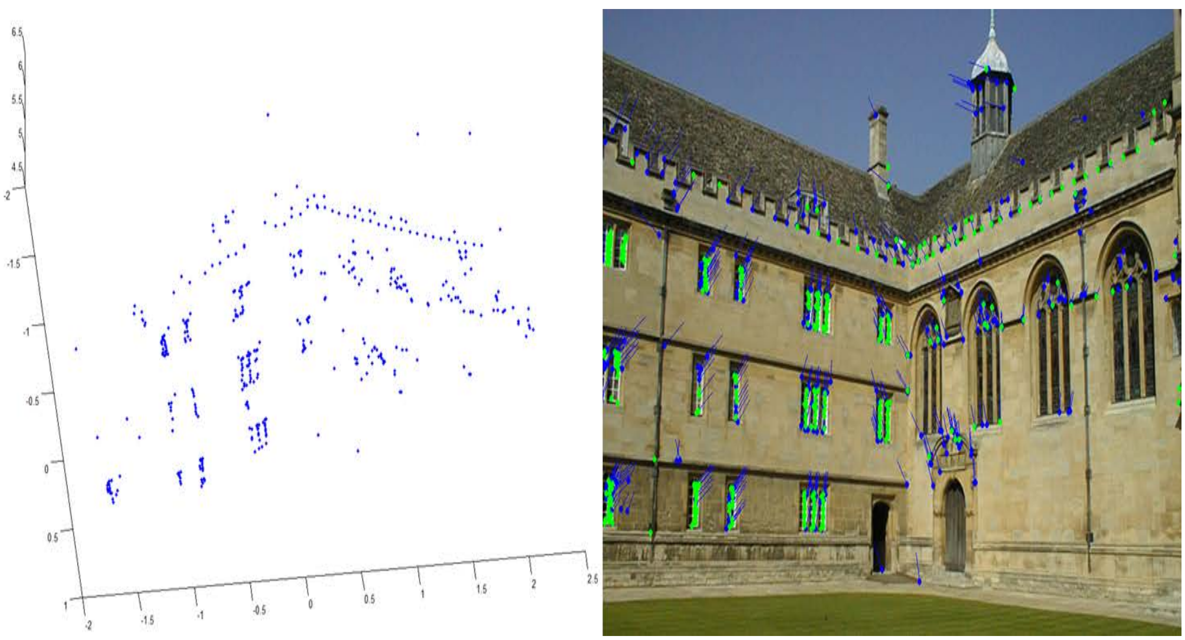

Figure 6. A reconstruction Oxford Wadham college from two views (Werner and Zisserman 2002).

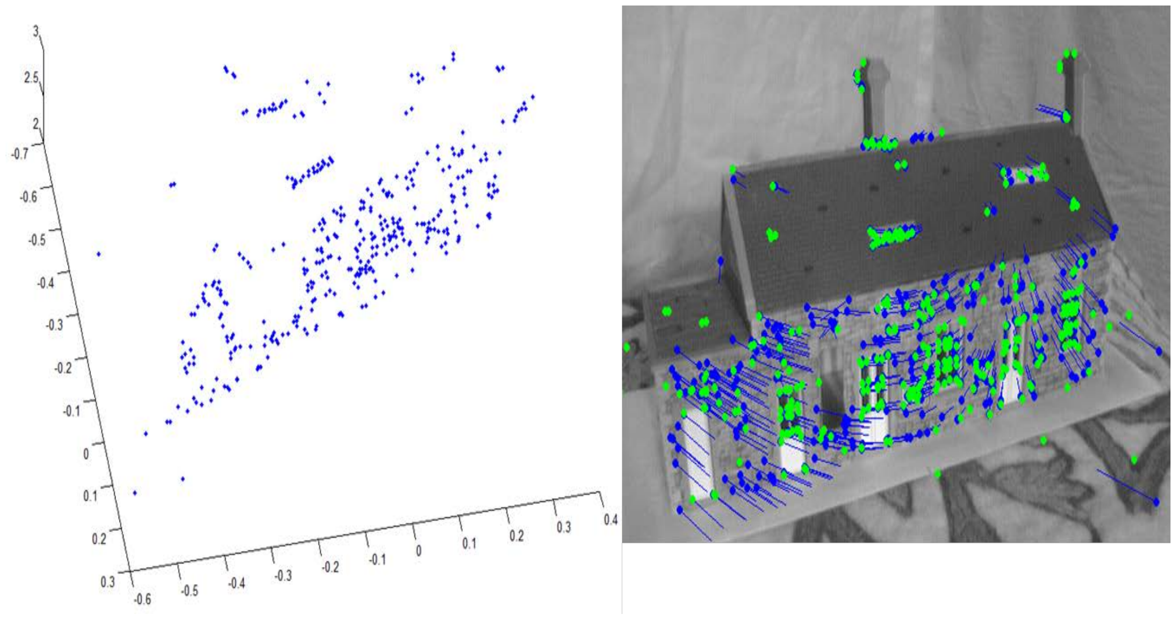

Figure 7. A reconstruction of a model house[2] from two views.

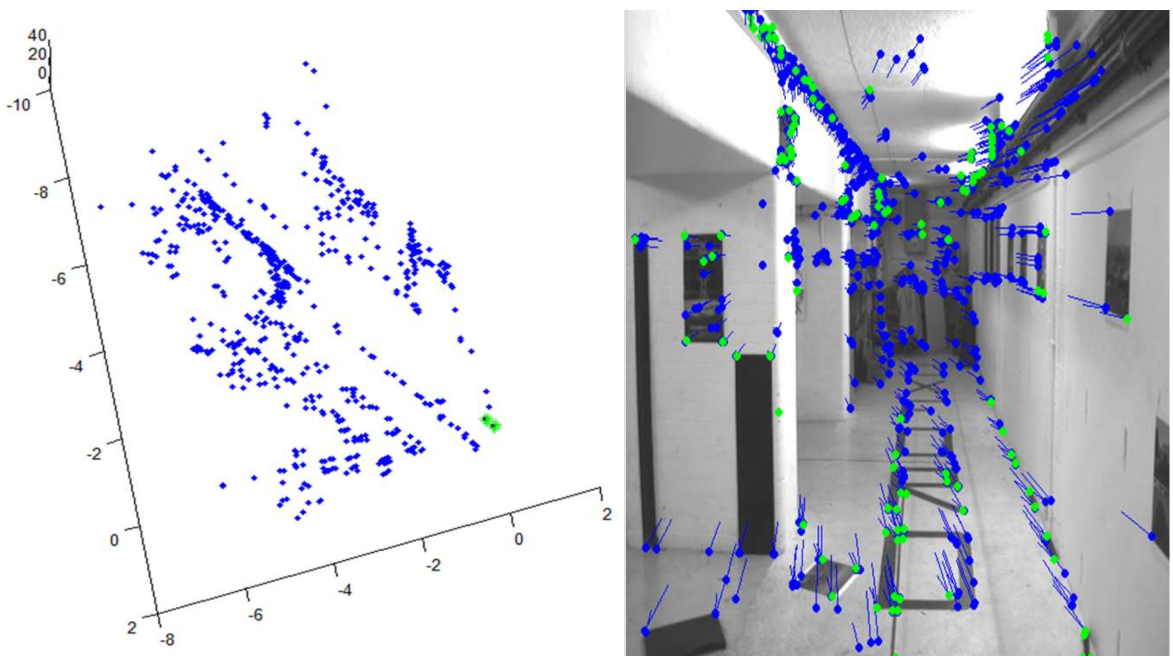

Figure 8. Reconstruction from the first two frames of the "corridor" sequence[2]. The two camera locations are shown as green spots on the left.

7. Scene Reconstruction from the Essential Matrix: Where's the Hack?

There have been many authors who presented good to high quality results in terms of recovering camera pose and scene structure from the essential matrix. To name a few renowned researchers, Pollefeys (Pollefeys, Van Gool et al. 2004), Nister (Nistér 2004), Zisserman and Hartley (Hartley and Zisserman 2003) have presented remarkable scene

---

[2] Images retrieved from http://www.robots.ox.ac.uk/~vgg/data/data-mview.html

reconstructions with their specialized algorithms for the computation of the fundamental matrix (and subsequently, of the essential matrix through known camera intrinsics). To the best of my knowledge, with the exception of Nister's algorithm[3], what these methods have in common is that they do not directly address the two primary constraints associated with the essential matrix: a) It has exactly two singular values and, b) These singular values are equal. Typically, the aforementioned algorithms enforce these constraints after the optimization. In my opinion, the ramifications of this strategy can be unpredictable, depending on the level of noise in the data. Enforcing two equal singular values is a brutal way of imposing constraints and can potentially alter the relative pose estimate to an extent at which no iterative refinement can recover from. As proven in theorem 1, matrix E is an essential matrix if and only if it can be written as the sum of two tensor products,

$$E = v_1 u_1^T + v_2 u_2^T \tag{56}$$

where $v_1, v_2, u_1, u_2 \in \mathbb{R}^3$ are unit vectors such that, $v_1^T v_2 = u_1^T u_2 = 0$. An alternative approach would be to use the standard formula for the essential matrix given in equation (8) and impose orthonormality constraints on the columns of the rotation matrix and a unit-norm constraint on the baseline. Either way, a properly constrained optimization involves a Lagrangian (or some parametrized expression) with the standard 8-point algorithm cost function and 5 Lagrange multipliers (2 for orthogonality and 3 for unit norms). The Karush-Kuhn-Tucker (KKT) conditions will eventually lead to a 4th degree polynomial system in the components of v_1, v_2, u_1, u_2. This system is relatively hard to obtain analytically and it requires a special category of algorithms known as Groebner basis solvers (Lazard 1983) to solve it. To the best of my knowledge, the parametrization of equation (3.32) is not frequently met in literature.

Nister's solution (Nistér 2004) deserves special reference in this section for being the only method (to the best of the author's knowledge) that actually solves for the essential matrix while abiding by the orthogonality constraints. The idea is to recover the essential matrix from the 4-dimensional null space of the data matrix. The constraints yield a polynomial system containing reasonably-sized expressions (only 4 unknowns up to arbitrary scale) and can be solved in a relatively uncomplicated manner. Of course, the problem with this method is that it does not generalize to the most usual formulation of the problem which is an overdetermined system[4], in which case the null space of the data matrix is rarely non-empty. For completeness, I would like to mention a parametrization by Vincent Lui and Tom Drummond for an iterative solution of the 5-point problem estimation (Lui and Drummond 2007). Other solutions for the 5-point problem involve Hongdong's (Li and Hartley 2006) and Kukelova's (Kukelova, Bujnak et al. 2008) methods.

[3] Of course, many variations of the 5-point algorithm have been proposed since Nister, but they do not essentially add something new to the concept.

[4] Nister mentions in his paper that the method generalizes to the overdetermined case if the 4 eigenvectors of the data matrix corresponding to the 4 smallest singular values are taken instead of the 4 null-space basis vectors used in the 5-point case. I believe that this is an arbitrary assertion. Clearly, one is free to employ the singular vectors to diagonalize the data matrix, but there is no justification about why the constrained minimum should be in the space of the smallest 4 singular values. It is a sane conjecture from a greedy point of view, but there is no proof and moreover, there is no justification about why 4 vectors should be used instead of, for instance, 5 which is the number of DOF of the essential matrix.

References

[1] H. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections," *Readings in computer vision: issues, problems, principles, and paradigms*, p. 61, 1987.

[2] R. Hartley, A. Zisserman, and I. ebrary, *Multiple view geometry in computer vision* vol. 2: Cambridge Univ Press, 2003.

[3] E. Trucco and A. Verri, *Introductory techniques for 3-D computer vision* vol. 93: Prentice Hall New Jersey, 1998.

[4] O. Faugeras, *Three-dimensional computer vision: a geometric viewpoint*: the MIT Press, 1993.

[5] O. Faugeras, Q.-T. Luong, and T. Papadopoulo, *The geometry of multiple images: the laws that govern the formation of multiple images of a scene and some of their applications*: the MIT Press, 2004.

[6] J. Y. Bouguet, "Pyramidal implementation of the affine lucas kanade feature tracker—description of the algorithm," Technical report). Intel Corporation2001.

[7] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, pp. 185-203, 1981.

[8] M. Z. Brown, D. Burschka, and G. D. Hager, "Advances in computational stereo," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 25, pp. 993-1008, 2003.

[9] Q. T. Luong and O. D. Faugeras, "The fundamental matrix: Theory, algorithms, and stability analysis," *International Journal of Computer Vision*, vol. 17, pp. 43-75, 1996.

[10] W. Chojnacki and M. J. Brooks, "On the consistency of the normalized eight-point algorithm," *Journal of Mathematical Imaging and Vision*, vol. 28, pp. 19-27, 2007.

[11] R. I. Hartley, "In defense of the eight-point algorithm," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 19, pp. 580-593, 1997.

[12] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry, *An invitation to 3-d vision: from images to geometric models* vol. 26: springer, 2003.

[13] B. K. P. Horn, "Recovering baseline and orientation from essential matrix," *J. Optical Society of America*, 1990.

[14] A. OpenGL, M. Woo, J. Neider, and T. Davis, "OpenGL programming guide," ed: Addison-Wesley, 1999.

[15] R. Hartley, "Ambiguous configurations for 3-view projective reconstruction," *Computer Vision-ECCV 2000*, pp. 922-935, 2000.

[16] A. Hejlsberg, S. Wiltamuth, and P. Golde, *C# language specification*: Addison-Wesley Longman Publishing Co., Inc., 2003.

[17] D. Fay, "An architecture for distributed applications on the Internet: overview of Microsoft's. NET platform," in *Parallel and Distributed Processing Symposium, 2003. Proceedings. International*, 2003, p. 7 pp.

[18] G. Bradski, "The opencv library," *Doctor Dobbs Journal*, vol. 25, pp. 120-126, 2000.

[19] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, pp. 381-395, 1981.