Texture Cues in Multiple Views

Ryan White

December 14, 2004

Abstract

In this project, we address the problem of computing a transformation between two cameras that image the same set of points and normals. In contrast to the traditional setting, where only points are used, we have more information and can provide an un-ambiguous solution. However, this comes at a computational cost and requires the use of a mixing constant, a combinatorial seach and gradient descent. We demonstrate the algorithm on images of cloth.

1 Introduction

This project addresses the computer vision problem of reconstructing the shape of a scene from a dense set of points and normals in two cameras. An example of this problem is shown in figure 1, where two images are taken of the same cloth. We wish to reconstruct the depth of the cloth at the center of each square.

Using standard computer vision techniques, we can accurately estimate the location of each point. In addition, we can estimate the correspondence of the patches between the two images and, to two fold ambiguity, we can estimate the direction of the normal to the surface. A further description of the normal ambiguity can be found in [4], but is sumarized in figure 2.

Our goal is to reconstruct the depth of each correspondence. Given information about the location and direction of each camera, this computation is simple. However, we wish to compute this information from the images (or, subsequently, the points and normals). As discussed in [6], without normal information, this reconstruction is ambiguous – however, by estimating the normals, we will accurately compute the cameras.

2 Problem Statement

For each patch in each image, there is a an (x,y) location, an (x,y,z) normal and an unknown sign bit that signifies the two-fold ambiguity of the normal. We will adopt the following notation for patch i in image k: the location is $(p^i_{k,x}, p^i_{k,y})$, the (unit) normal is $(n^i_{k,x}, n^i_{k,y}, n^i_{k,z})$ and the sign bit is $s^i_k \in \{-1, 1\}$. We will denote the transformation between the two cameras as the matrix Q and the relative scale of the two cameras as K. Since we assume that our cameras are orthographic, we know that Q should be orthonormal.

Using this notation, we can state our problem as find a Q that satisfies the following two equations:

$$\begin{bmatrix} p^1_{1,x} & \cdots & p^n_{1,x} \\ p^1_{1,x} & \cdots & p^n_{1,x} \\ \times & \cdots & \times \end{bmatrix} = K \begin{bmatrix} & & \\ & Q & \\ & & \end{bmatrix} \begin{bmatrix} p^1_{2,x} & \cdots & p^n_{2,x} \\ p^1_{2,x} & \cdots & p^n_{2,x} \\ \times & \cdots & \times \end{bmatrix}$$

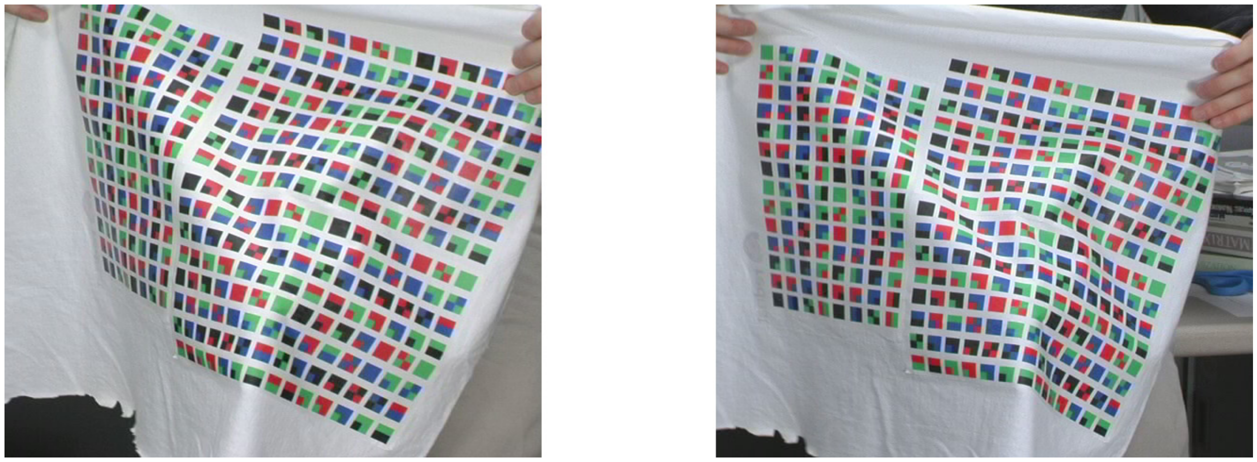

Figure 1: A sample image of the two views of the cloth. We reconstruct the shape by determining the locations and normals for each of the squares. From the two views, we can reconstruct the locations of the camera and subsequently determine the shape of the cloth in 3D.

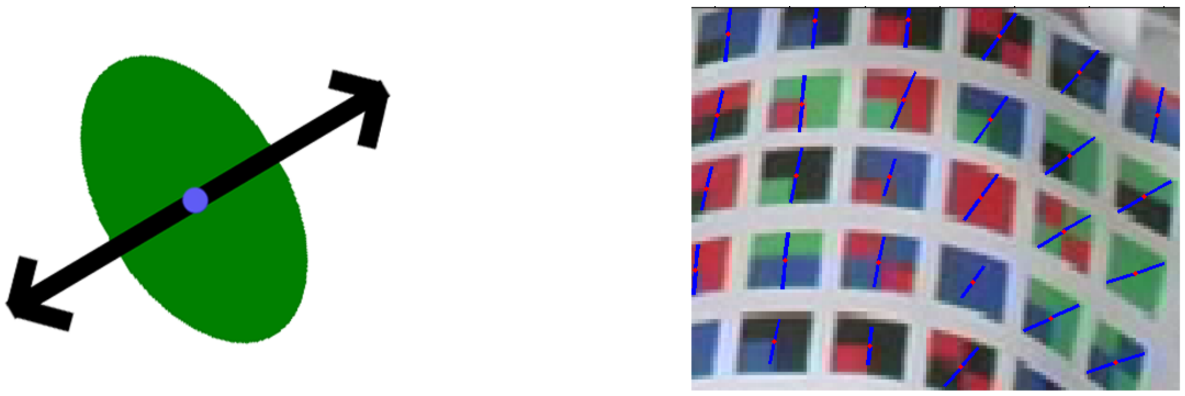

Figure 2: On the left, an oblique view of a circular texture element. Since we assume that we know that the frontal image of the texture element is circular, we can estimate the normal up to a two-fold ambiuity. The ambiguous normals are shown as the black errors and cannot easily be distinguished from a single view. However, with the solution provided in this paper will allows us to determine which normal is correct. On the left is this ambiguity show on real data.

$$\begin{bmatrix} s_1^1 \cdot n_{1,x}^1 & \dots & s_1^n \cdot n_{1,x}^n \\ s_1^1 \cdot n_{1,y}^1 & \dots & s_1^n \cdot n_{1,y}^n \\ n_{1,z}^1 & \dots & n_{1,z}^n \end{bmatrix} = \begin{bmatrix} & & \\ & Q & \\ & & \end{bmatrix} \begin{bmatrix} s_2^1 \cdot n_{2,x}^1 & \dots & s_1^n \cdot n_{2,x}^n \\ s_2^1 \cdot n_{2,y}^1 & \dots & s_1^n \cdot n_{2,y}^n \\ n_{2,z}^1 & \dots & n_{2,z}^n \end{bmatrix}$$

Here, the points p_k^i and the normals n_k^i are known and the signs s_k^i and orthonormal matrix Q are unknown. In some cases we will assume that the signs s_k^i are known and use the following simplified notation:

$$\left[\begin{array}{ccc|c} & P_1 & & N_1 \\ \times & \dots & \times & \end{array}\right] = \begin{bmatrix} & Q & \end{bmatrix} \left[\begin{array}{ccc|c} & K \cdot P_2 & & N_2 \\ \times & \dots & \times & \end{array}\right] \tag{1}$$

For each of these problems, ideally, we can solve for the matrix Q, the sign bits s_k^i and the corresponding depths $p_{k,z}^i$ which were represented above as $\times$. We will assume from this point on that the system of equations is overconstrained. Exact solutions are, in general, not possible and we will focus on methods that minimize a cost function that charges for deviation from equality. In the following sections, we will explore different cost functions and corresponding solution methods.

3 Problem Characterization

Our problem, as defined above is actually ambiguous – even with normal information. Suppose that we find a solution to the problem in the form:

$$\begin{aligned} Q &= \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{21} & q_{22} & q_{23} \\ q_{31} & q_{32} & q_{33} \end{bmatrix} \\ s_k^i &= \hat{s}_k^i \end{aligned}$$

Then, we can compose an equally valid solution using:

$$\begin{aligned} Q &= \begin{bmatrix} q_{11} & q_{12} & -q_{13} \\ q_{21} & q_{22} & -q_{23} \\ -q_{31} & -q_{32} & q_{33} \end{bmatrix} \\ s_k^i &= -\hat{s}_k^i \end{aligned}$$

Using the notation from before:

$$\begin{bmatrix} & P_1 & \end{bmatrix} = \begin{bmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{bmatrix} \begin{bmatrix} & K \cdot P_2 & \end{bmatrix}$$

and

$$\begin{aligned}
\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \\ N_1 \\ \\ \end{bmatrix} &= \begin{bmatrix} \\ Q \\ \\ \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \\ N_2 \\ \\ \end{bmatrix} \\
\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \\ N_1 \\ \\ \end{bmatrix} &= \begin{bmatrix} q_{11} & q_{12} & -q_{13} \\ q_{21} & q_{22} & -q_{23} \\ -q_{31} & -q_{32} & q_{33} \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \\ N_2 \\ \\ \end{bmatrix} \\
\begin{bmatrix} -\hat{s}_1^1 \cdot n_{1,x}^1 & \dots & -\hat{s}_1^n \cdot n_{1,x}^n \\ -\hat{s}_1^1 \cdot n_{1,y}^1 & \dots & -\hat{s}_1^n \cdot n_{1,y}^n \\ n_{1,z}^1 & \dots & n_{1,z}^n \end{bmatrix} &= \begin{bmatrix} q_{11} & q_{12} & -q_{13} \\ q_{21} & q_{22} & -q_{23} \\ -q_{31} & -q_{32} & q_{33} \end{bmatrix} \begin{bmatrix} -\hat{s}_2^1 \cdot n_{2,x}^1 & \dots & -\hat{s}_1^n \cdot n_{2,x}^n \\ -\hat{s}_2^1 \cdot n_{2,y}^1 & \dots & -\hat{s}_1^n \cdot n_{2,y}^n \\ n_{2,z}^1 & \dots & n_{2,z}^n \end{bmatrix}
\end{aligned}$$

This ambiguity is a failure of the points and normals representation of the information in the scene, and can only be resolved using other vision cues. For the purpose of this paper, we will assume $s_1^1 = 1$.

4 Background

Several methods exist to compute the camera, or an equivalent transformation given a subset of the information. These methods are sumarized in the following subsections, but will not provide our final solution.

4.1 Solution using Points

Omitting the normals and just solving for the points results in a system with a degree of ambiguity. (The basis for this approach was developed in [5]) Ideally, a solution would provide some form of trade-off between the accuracy in the points and the accuracy in the normals, while resolving any ambiguity in unknowns by using the normals.

Our cost for the points is based on the reconstruction error of the points. We will define the following quatities: the point data matrix $\mathcal{D}$, the camera matrix $\mathcal{C}$ and the 3D point location $\mathcal{P}$:

$$\mathcal{D} = \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} \tag{2}$$

$$\mathcal{C} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ q_{11} & q_{12} & q_{13} \\ q_{21} & q_{22} & q_{23} \end{bmatrix} \tag{3}$$

$$\mathcal{D} = \mathcal{C} \cdot \mathcal{P} \tag{4}$$

We solve for the point locations in 3D using the psuedo inverse of the camera matrix. $\mathcal{P} = \mathcal{C}^\dagger \cdot \mathcal{D}$ We now compute the cost as:

$$C_p = \|\mathcal{C}\mathcal{C}^\dagger\mathcal{D} - \mathcal{D}\|$$

Without the constraint that $\mathcal{C}$ satisfy 4, we can compute the optimal $\mathcal{C}$ using the singular value decomposition on $\mathcal{D}$:

$$
\begin{aligned}
\mathcal{D} &= U \cdot \Sigma \cdot V \\
\mathcal{D} &= U \cdot U_{1..3,1..3}^{-1} \cdot U_{1..3,1..3} \cdot \Sigma \cdot V \\
\mathcal{D} &= U \cdot U_{1..3,1..3}^{-1} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ a & b & c \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\frac{a}{c} & -\frac{b}{c} & \frac{1}{c} \end{bmatrix} \cdot U_{1..3,1..3} \cdot \Sigma \cdot V
\end{aligned}
$$

Now, for a certain set of the values (a, b, c) we can compute the camera matrix $\mathcal{C}$ and the 3D point locations $\mathcal{P}$ as:

$$
\begin{aligned}
\mathcal{C} &= U \cdot U_{1..3,1..3}^{-1} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ a & b & c \end{bmatrix} \\
\mathcal{P} &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\frac{a}{c} & -\frac{b}{c} & \frac{1}{c} \end{bmatrix} \cdot U_{1..3,1..3} \cdot \Sigma \cdot V
\end{aligned}
$$

We omit the exact details of solving for (a, b, c), the requirements on U such that (a, b, c) exist and the corresponding ambiguity in $\mathcal{C}$ in this case, since it solves only a subset of our problem. Note that we can reconstruct the rotation matrix Q from the camera matrix $\mathcal{C}$ up to a sign ambiguity in the third row of Q.

4.2 Solution using Normals

Alternatively, one could solve for Q by omitting the points ($p_{k,x}^i$ and $p_{k,y}^i$) and using only the normals ($n_{k,x}^i, n_{k,y}^i, n_{k,z}^i$). This results in a metric solution to the problem (minus the single ambiguity presented in section 3). However, these solutions are significantly less accurate, because, in general, the accuracy of the normals is less than the accuracy of the point locations. We use the method of Orthogonal Procrustes to solve this problem:

$$
\begin{bmatrix} & & \\ & N_1 & \\ & & \end{bmatrix} = \begin{bmatrix} & & \\ & Q & \\ & & \end{bmatrix} \begin{bmatrix} & & \\ & N_2 & \\ & & \end{bmatrix}
$$

We can formulate the cost using the frobenius norm and create the following maximization problem:

$$
\begin{aligned}
Q &= \operatorname{argmax} \|N_1 - Q \cdot N_2\|_F \\
& \quad s.t.\ Q^T Q = I
\end{aligned}
$$

The optimal solution can be found using the singular value decomposition:

$$\begin{aligned}
Q &= \text{argmax } \|N_1 - QN_2\|_F \\
&= \text{argmax tr}((N_1 - QN_2)^T(N_1 - QN_2)) \\
&= \text{argmax tr}(N_1^T N_1 - N_2^T Q^T N_1 - N_1^T Q N_2 + N_2^T Q^T Q N_2) \\
&= \text{argmax tr}(N_1^T N_1) + 2\text{tr}(QN_2N_1^T) + \text{tr}(N_2^T N_2) \\
&= \text{argmax tr}(QN_2N_1^T) \\
N_2N_1^T &= U \cdot \Sigma \cdot V
\end{aligned}$$

As stated in [1], the optimal solution for Q is $U \cdot V$.

5 Method

We solve this problem in two parts: first, we present the solution if all of the signs (s_k^i) are known. Then, we'll present a hueristic method for searching for the correct signs.

5.1 Known Signs

We start by solving for the matrix Q assuming that we know the sign bits. This allows us to simplify our problem to a nolinear optimization problem in three variables. To do this, we define two cost functions: one for the points and one for the normals. Following the work from previous sections, these costs are:

$$\begin{aligned}
C_p &= \|\mathcal{C}\mathcal{C}^\dagger \mathcal{D} - \mathcal{D}\| \\
C_n &= -\text{tr}(N_1^T \cdot Q \cdot N_2) \\
C &= C_p + \lambda C_n
\end{aligned}$$

The normal cost is the dot product of the normal in the first frame with the transformed normal from the second frame. We will parameterize the rotation as a composition of three Given's rotations:

$$Q = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\phi & 0 & \sin\phi \\ 0 & 1 & 0 \\ -\sin\phi & 0 & \cos\phi \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & \sin\psi \\ 0 & -\sin\psi & \cos\psi \end{bmatrix}$$

One can easily verify that this matrix is orthogonal, and Euler's Rotation Theorem states that it defines the entire class of rotations in $\Re^3$.

Now, with a cost function that can be computed as a function of three variables, we perform gradient descent using the built in optimization toolbox in MATLAB.

5.2 Searching for Signs

An arbitrary system requires a search over the $(2^N)^2$ possible sign bits. However, an overconstrained system that satisfies the equality constraints can be solved in a search over a constant number of sign flips.

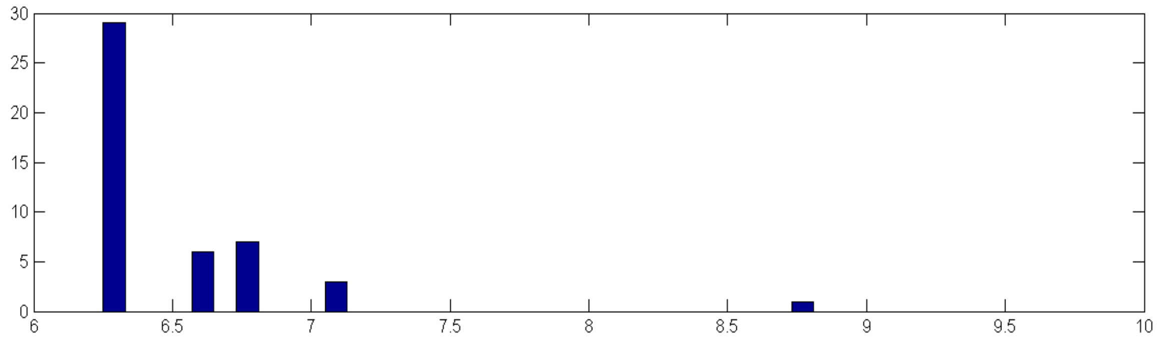

Figure 3: A histogram of 100 minimization scores used in computing the rotation matrix Q. Each sample was computed using a single run of the algorithm in section 5.2. The dominating value at 6.3 indicates that we have found the minimum for this problem given the noise in the system. We use a biased the cost function so that 0 corresponds to a perfect solution to the problem posed in equation 1.

If we denote the number of points as N_p, our system has $3+N_p$ unknowns (3 for the rotation matrix Q and N_p for the depth of each points) and $6 \cdot N_p$ equations (3 for each point and 3 for each normal). Technically, that means that any patch should be enough to solve for the transformation matrix Q. Therefore, if the coefficients in the system of equations had no errors, we could compute the rotation matrix Q from a single match (with a search over a single normal ambiguity in each view) and then compute all subsequent normal ambiguities. We compute the two sign bits using the following combinatorial minimization over 4 possibilities:

$$\min_{(s_1^i, s_2^i) \in \{-1,1\}} \begin{bmatrix} s_1^i \cdot n_{1,x}^i \\ s_1^i \cdot n_{1,y}^i \\ n_{1,z}^i \end{bmatrix}^T \cdot Q \cdot \begin{bmatrix} s_2^i \cdot n_{2,x}^i \\ s_2^i \cdot n_{2,y}^i \\ n_{2,z}^i \end{bmatrix}$$

Since we have errors in our coefficients, we will robustly search for a camera by picking N_m matches at random and combinatorially search over $(2_m^N)^2$ minimization problems as defined in the previous section. Once we have selected the rotation matrix Q, we disambiguate the rest of the matches, then recompute a more accurate Q using all of the matches. Since any pair of matches may be inaccurate, we repeat this process many times and pick the lowest score in the set.

6 Results

We ran the algorithm described in the previous section on a real vision problem involving images of cloth. As shown in figure 4, our reconstruction is fairly accurate, implying that the algorithm worked. We look at the specifics of the hueristic search for normals in figure 3 and notice that the lowest score in the set contains more votes than any other score. At a rough level, this implies that our search over a smaller set is reasonable, since we often find the same minimum score.

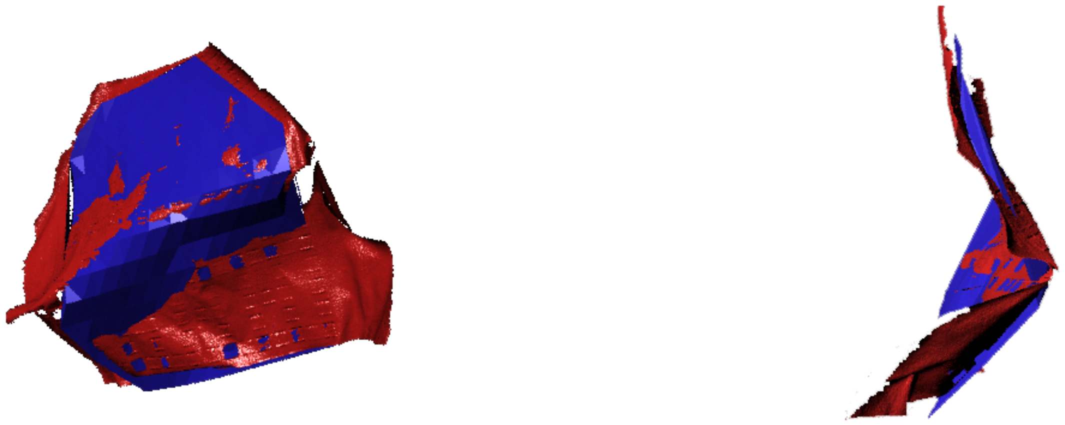

Figure 4: A match between ground truth (red) and our reconstruction (blue). Ground truth was obtained using a 3D depth scan of the scene.

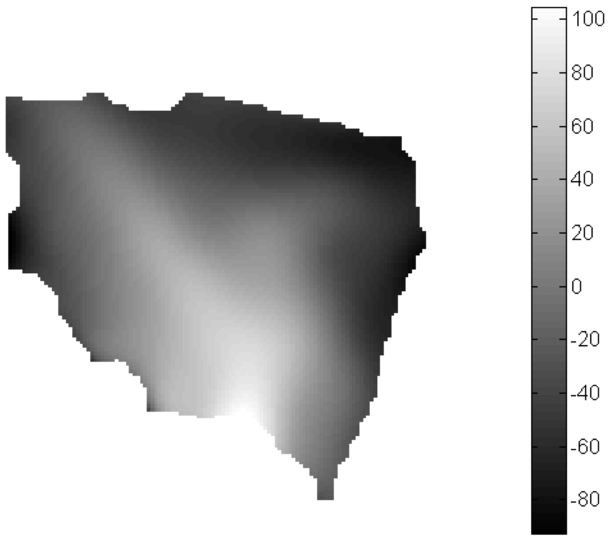

Figure 5: A reconstructed depth based on the alorithm presented in this paper from the images in 1.

7 Discussion

Ideally a solution to this problem would come with a guarantee of optimality and a estimate of run time. As with many computer vision problems, we justify the weaknesses of our algorithm with empirical evidense that it runs quickly (2.35 seconds per iteration in MATLAB) and robustly (we used this on several examples not included in this paper).

References

[1] G.H. Golub and C.F. Van Loan. *Matrix Computations.* John Hopkins University Press, 1996.

[2] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2000.

[3] B.K.P Horn, H.M. Hilden, and S. Negahdaripour. Closed- form solution of absolute orientation using orthonormal matrices. *Journal Optical Society America*, 5(7):1127–1135, 1988.

[4] Anthony Lobay and D.A. Forsyth. Recoverig shape and irradiance maps from rich dense texton fields. In *Computer Vision and Pattern Recognition*, 2004.

[5] Carlo Tomasi and Takeo Kanade. Shape and Motion from Image Streams: a Factorization Method, Full Report on the Orthographic Case. Technical Report CMU-CS-92-104, Carnegie Mellon, March 1992.

[6] S. Ullman. *Interpretation of Visual Motion.* MIT Press, 1979.