Recovering Camera Location, Camera Orientation, and World Marker Locations From Photographs

Jack Goldfeather
February 24, 2010

1 Least Square Fits for Quadratics

Let

\[ F(x_1, ..., x_n) = \sum_{i=1}^{k} (a_{0i} + a_{1i}x_1 + ... a_{ni}x_n)^2 \]

That is, \( F(x_1, ..., x_n) \) is the sum of squares of linear functions in \( n \) variables. This function is never negative, and a common task in what follows is to find \( X = (x_1, ..., x_n) \) that makes \( F(x_1, ..., x_n) \) as close to 0 as possible. There are two versions of this which we consider separately.

1.1 Method A: \( a_{0i} = 0 \) for all \( i \) and \( X \) constrained to be a unit vector

1. Square everything out and collect terms to write

\[ F(x_1, ..., x_n) = \sum_{i=1}^{n} b_{ii}x_i^2 + \sum_{i=1}^{n} \sum_{j=i+1}^{n} b_{ij}x_i x_j \]

2. Create the symmetric matrix \( C = (c_{ij}) \) from these coefficients by defining \( c_{ii} = b_{ii}, \ c_{ij} = b_{ij}/2 \) for \( i < j \), and \( c_{ij} = c_{ji} \) for \( j < i \). For example if \( F(x_1, x_2) = b_{11}x_1^2 + b_{22}x_2^2 + b_{12}x_1x_2 \) then

\[ C = \begin{pmatrix} b_{11} & b_{12}/2 \\ b_{12}/2 & b_{22} \end{pmatrix} \]

Since \( C \) is a symmetric matrix, by a theorem in linear algebra, the eigenvalues of \( C \) are all real, and the associated eigenvectors are orthogonal to each other. Suppose the eigenvalues are \( \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n \) and the associated unit eigenvectors are \( v_1, ..., v_n \). If \( P \) is the matrix whose columns are these eigenvectors and \( D \) is the diagonal matrix of eigenvalues, the Diagonalization Theorem in linear algebra says that:

\[ C = PDP^{-1} \]

3. It can be proved from this that \( X = v_1 \), the unit eigenvector associated to the \textit{smallest} eigenvalue, minimizes \( F \) as described above.

1.2 Method B: No conditions on \( a_{0i} \) or \( X \)

The minimum of the multivariable function \( F \) occurs where

\[ \nabla F(X) = \left( \frac{\partial F}{\partial x_1}, ..., \frac{\partial F}{\partial x_n} \right) = 0 \]

Since \( F(X) \) is quadratic, \( \nabla F(X) = 0 \) is a system of \( n \) linear equations in \( n \) unknowns which can be solved using any linear system solver. There should be only one solution since \( F \) has no maximum value.
2 Rotation and Translation

Let $R_j$ be the (unknown) rotation matrix from world to camera $j$ coordinates and let $T_j$ be the (unknown) translation vector from the world origin to camera $j$ origin. We can find $R_j$ and $T_j$ provided:

1. We can identify several vectors in the cameras view that are known to be parallel to the world coordinate axes. In a room, if we fix the world origin to be a corner of the room, we can use lines along walls and windows, etc. Let $X = \{X_1, ..., X_{n_1}\}$, $Y = \{Y_1, ..., Y_{n_2}\}$, and $Z = \{Z_1, ..., Z_{n_3}\}$ be the sets of these vectors parallel to the three coordinates axes, respectively. This alone allows us to compute $R_j$.

2. We have measured the world positions of a set of points $P = \{P_1, ..., P_s\}$ and can identify them in at least 2 camera images. In this case, we can find $T_j$ by solving a linear system of 3 equations in 3 unknowns.

3. We can only identify the set of points $P = \{P_1, ..., P_s\}$ in at least 2 camera images. In this case, we can find $T_j$ and the coordinates of all the $P_i$ by solving a large system of linear equations. This is likely to be less accurate than the previous method.

2.1 Finding $R_j$

In Figure 1, uppercase letters indicate a vector expressed in world coordinates and lowercase letters indicate a vector expressed in camera coordinates. Vector $V$ is one of the vectors parallel to a world coordinate axis. We will assume in what follows that $V = X_1$, a vector parallel to the x-axis, since the other cases are similar. The vector $\vec{vc}$ is the image of this vector in the camera and its endpoints $(p_1, p_2)$ and $(q_1, q_2)$ are the measured (i.e. known) camera pixel coordinates. If the focal length of the camera is $F$, then

$$\vec{pc} = (p_1, p_2, F) \quad \vec{qc} = (q_1, q_2, F)$$

Then the normal to the plane passing through the camera origin and containing these two vectors can be computed:

$$\vec{nc} = \vec{pc} \times \vec{qc}$$

Hence to each vector in $X_i \in X$, we can associate a normal vector $m_i = \vec{nc}$ to the plane through the camera origin and $X_i$. We convert $X_i$ into camera coordinates by multiplying it by the yet unknown rotation matrix $R_j$. That is $R_jX_i$ is perpendicular to $\vec{m}_i$ so

$$m_i \circ (R_jX_i) = 0$$  \hspace{1cm} (1)
Now let
\[ R_j = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \]  

Any rotation matrix has the property that its columns are unit vectors. Since \( X_i = (s_i, 0, 0) \), if we let \( \bar{m}_i = (a_i, b_i, c_i) \) and plug into equation (1) we obtain:

\[ (a_i r_{11} + b_i r_{21} + c_i r_{31}) = 0 \]  

Given that we have measured the vectors \( \bar{m}_i \), we won’t, in general, be able to find a simultaneous solution \((r_{11}, r_{21}, r_{31})\) for all \( i \). Instead we try to find a solution that minimizes

\[ \sum_{i=1}^{n_1} (a_i r_{11} + b_i r_{21} + c_i r_{31})^2 \]  

Method A from Section 1 can be used to find a best estimate of the first column of \( R_j \). Repeating this process for the vectors in \( Y \) and \( Z \) gives us best estimates for the 2nd and 3rd columns of \( R_j \).

### 2.2 Finding \( T_j \)

In Figure 1, observe that the vectors \( T - P \) and \( T - Q \) also lie in the plane containing \( V \) and the camera origin. If we let \( P = P_i \) and \( Q = P_k \), two of the measured world points, then, as in the previous section, we can find a vector \( \bar{m}_{ik} \) normal to the plane containing the camera origin and the vector connecting \( P_i \) to \( P_k \). We convert these vectors into camera coordinates using the (now known) rotation matrix \( R_j \). So we know that

\[ \bar{m}_{ik} \circ R_j(T_j - P_i) = 0 \quad m_{ik} \circ R_j(T_j - P_k) = 0 \]  

Again, we try to minimize

\[ \sum_{i \neq k} (\bar{m}_{ik} \circ R_j(T_j - P_i))^2 + (m_{ik} \circ R_j(T_j - P_k))^2 \]  

Since \( m_{ik}, R_j, P_i, \) and \( P_k \) are all known and \( T_j = (t_1, t_2, t_3) \), equation (6) reduces to a quadratic expression in \( t_1, t_2, \) and \( t_3 \). That is, we need to minimize a function of the form

\[ f(t_1, t_2, t_3) = c_1 t_1^2 + c_2 t_2^2 + c_3 t_3^2 + c_4 t_1 t_2 + c_5 t_1 t_3 + c_6 t_2 t_3 + c_7 t_1 + c_8 t_2 + c_9 t_3 + c_{10} \]  

A best estimate can be found by using Method B from Section 1.

If the coordinates of the points \( P_i \) are not known, then equation (6) still is quadratic in the coordinate variables \( U = (t_1, t_2, t_3, P_{11}, P_{12}, P_{13}, ..., P_{31}, P_{32}, P_{33}) \), so we can still solve a linear system in \( 3s + 3 \) unknowns. In this case we will get a solution of the form \( s_j U_j \). That is, we will only find a solution up to a scale factor.

### 3 Finding the World Coordinates of Markers

Once the camera parameters for each camera are computed (focal length \( F_j \), rotation matrix \( R_j \), translation vector \( T_j \)) we can find the world coordinates of any marker by knowing its pixel coordinates in at least 2 cameras.

Let \( p_j = (x_j, y_j, F_j) \) where \( (x_j, y_j) \) is the pixel address of the marker in camera \( j \) and \( F_j \) is the focal length of camera \( j \). Then \( X_j = R_j^{-1} p_j \) is the vector \( p_j \) expressed in world coordinates. Since \( T_j \) is the world coordinate expression for the camera \( j \) origin, the parametric equation of the line passing through the camera origin \( j \) in the direction of \( X_j \) is given by

\[ P_j(t_j) = T_j + t_j X_j. \]

All we know now is that our marker lies somewhere along this line.

If we can find \( t_i \) and \( t_j \) such that \( P_i(t_i) = P_j(t_j) \) for cameras \( i \) and \( j \), this common point is our correct 3D world location of the marker. Of course, due to measurement errors, this is not likely, so instead we minimize the square-sum error of all the lengths of vectors \( P_i(t_i) - P_j(t_j) \). If there are \( N \) cameras that can see the marker, we minimize the function

\[ F(t_1, ..., t_N) = \sum_{i=1}^{N} \sum_{j=i+1}^{N} (|P_i(t_i) - P_j(t_j)|)^2 = \sum_{i=1}^{N} \sum_{j=i+1}^{N} (|T_i - T_j + t_i X_i - t_j X_j|)^2 \]  

which is quadratic in \( (t_1, ..., t_N) \) and so can be minimized using Method B from Section 1.