Optimal Ray Intersection For Computing 3D Points From N-View Correspondences

Greg Slabaugh, Ron Schafer, Mark Livingston

October 2, 2001

Introduction

Often in computer vision, one must triangulate corresponding points between two images in order to compute depth. Due to errors in quantization, camera calibration, and correpondence, rays back-projected from the images into three-dimensional space rarely intersect. As a result, one must find a point that is optimally close to the two rays. This report generalizes this problem for the case of multiple images; i.e., how to compute the optimal position of three-dimensional point given rays emanating from corresponding pixels in N images, where $N \geq 2$. We develop a simple linear technique and provide an example to demonstrate the method.

Notation

In this report, points in space are given capital letters, and the point coordinates are given in parentheses, such as $P = (x, y, z)$. Vectors are given a bold font. A normalized vector is denoted with a hat. Vectors are expressed as a linear combination of the unit vectors $\hat{\mathbf{x}}$, $\hat{\mathbf{y}}$, and $\hat{\mathbf{z}}$. For example, $\hat{\mathbf{d}}_\mathbf{1} = a_1\hat{\mathbf{x}} + b_1\hat{\mathbf{y}} + c_1\hat{\mathbf{z}}$ is a normalized vector. Rays are denoted with the $\vec{\ }$ symbol, such as $\vec{\ell}$.

Computing Depth Using Two Views

We begin by considering a common method [4] for stereo triangulation. Here, we assume that the camera parameters are known, and that the projections, P_1 and P_2 of a 3D point into two images, I_1 and I_2 are known, as shown in Figure 1. Points P_1 and P_2 are called a *correspondence*, as they both correspond to the same point P in 3D space. Our task is to compute the location of point P, given P_1, P_2, and the camera parameters.

Figure 1 shows that for each camera, we can back-project a ray from the camera center C_i, through the image pixel of the correspondence, forming a ray $\hat{\mathbf{d}}_i$ into 3D space. Ideally, these rays would intersect exactly at the same 3D point. However, since the camera parameters and correspondence locations in image space are known only approximately, the rays will not actually intersect in 3D space. So instead, we seek to find a 3D point that has minimal

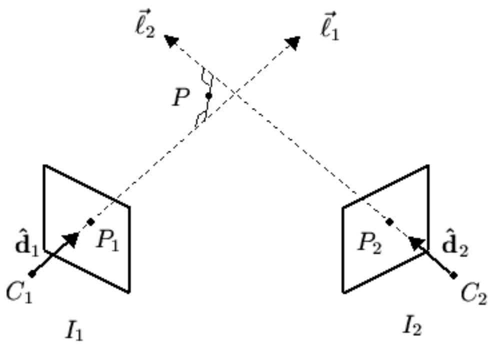


Figure 1: A common method for stereo triangulation.

distance from both rays. This point will be located on a line segment that is orthogonal to the rays, as shown in Figure 1. A standard approach then, first computes the endpoints of this line segment. From these, one computes the midpoint P of the line segment. Point P is the point in 3D space that is optimally close to the two non-intersecting rays.

This approach works well for stereo triangulation. However, what if one has a correspondence visible in N views, for $N > 2$? One could triangulate the correspondence between M different pairs of views, possibly all $\begin{pmatrix} N \\ 2 \end{pmatrix}$ pairs, resulting in M estimates of the point P. These estimates could then be combined to produce a single value for the point P. While this would result in a reasonable solution, there are a few drawbacks to this approach. First, the triangulation algorithm would be executed M times, once for each pair of views chosen. If all pairs of views are chosen, this requires executing the triangulation algorithm $O(N^2)$ times. For large N this can be computationally intensive. Perhaps even more significant is that the combination of these pairwise results does not guarantee a solution that is optimal in the sense of having minimum distance to all rays.

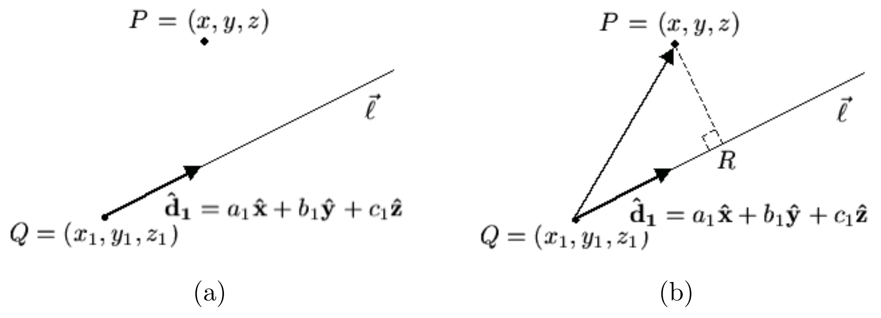
In the next section, we describe a simple technique that executes in $O(N)$ time and additionally finds a point P that is optimally close to all N rays.

Computing Depth Using N Views

The Distance Between a Point and a Ray

In this subsection we derive the equation for the distance between a point and a ray.

Consider Figure 2(a), which shows a point $P = (x, y, z)$ and a ray $\vec{\ell}$ that starts at point $Q = (x_1, y_1, z_1)$ and has a normalized direction $\hat{\mathbf{d}}_\mathbf{1} = a_1\hat{\mathbf{x}} + b_1\hat{\mathbf{y}} + c_1\hat{\mathbf{z}}$. Our task in this subsection is to derive an equation for the distance between the point P and the ray $\vec{\ell}$. Intuitively, this distance is along a line that goes through P and is orthogonal to $\vec{\ell}$. In Figure 2(b), this distance is represented by the length of vector $\mathbf{RP}$.



$P = (x, y, z)$
$\vec{\ell}$
$\mathbf{\hat{d}_1} = a_1\mathbf{\hat{x}} + b_1\mathbf{\hat{y}} + c_1\mathbf{\hat{z}}$
$Q = (x_1, y_1, z_1)$
(a)
$P = (x, y, z)$
$\vec{\ell}$
R
$\mathbf{\hat{d}_1} = a_1\mathbf{\hat{x}} + b_1\mathbf{\hat{y}} + c_1\mathbf{\hat{z}}$
$Q = (x_1, y_1, z_1)$
(b)

Figure 2: Finding the shortest distance between a point and a ray.

The vector **QR** is the projection of **QP** onto the ray $\vec{\ell}$. From the figure, we note that

$$\mathbf{QP} = (x - x_1)\mathbf{\hat{x}} + (y - y_1)\mathbf{\hat{y}} + (z - z_1)\mathbf{\hat{z}}, \tag{1}$$

and this vector has a length

$$||\mathbf{QP}|| = \sqrt{(x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2}. \tag{2}$$

Thus, **QR**, the projection of **QP** onto $\mathbf{\hat{d}_1}$ is

$$\begin{aligned} \mathbf{QR} &= \left(\mathbf{QP} \cdot \mathbf{\hat{d}_1}\right)\mathbf{\hat{d}_1} & (3) \\ &= [a_1(x - x_1) + b_1(y - y_1) + c_1(z - z_1)]\,\mathbf{\hat{d}_1} & (4) \end{aligned}$$

Since $\mathbf{\hat{d}_1}$ has unit magnitude, the length of **QR** is then

$$||\mathbf{QR}|| = a_1(x - x_1) + b_1(y - y_1) + c_1(z - z_1). \tag{5}$$

Our goal is find the length of the vector **RP**. Since the points PQR form a right triangle, we can invoke the Pythagorean thereom,

$$||\mathbf{QP}||^2 = ||\mathbf{QR}||^2 + ||\mathbf{RP}||^2, \tag{6}$$

or

$$||\mathbf{RP}||^2 = ||\mathbf{QP}||^2 - ||\mathbf{QR}||^2 \tag{7}$$

Substituting in values gives

$$||\mathbf{RP}||^2 = (x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2 - [a_1(x - x_1) + b_1(y - y_1) + c_1(z - z_1)]^2 \tag{8}$$

Thus,

$$||\mathbf{RP}|| = \sqrt{(x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2 - [a_1(x - x_1) + b_1(y - y_1) + c_1(z - z_1)]^2} \tag{9}$$

Equation 9 is the distance between a point $P = (x, y, z)$ and a ray $\vec{\ell}$ that starts at point $Q = (x_1, y_1, z_1)$ and has a normalized direction $\mathbf{\hat{d}_1} = a_1\mathbf{\hat{x}} + b_1\mathbf{\hat{y}} + c_1\mathbf{\hat{z}}$.

Minimizing the Sum of Squared Distance

To compute the total distance, $D(x, y, z)$ between a point and N rays, Equation 9 is evaluated for each ray i and the results are summed. This yields an equation of the form

$$D(x,y,z) = \sum_{i=1}^{N} \sqrt{(x-x_i)^2 + (y-y_i)^2 + (z-z_i)^2 - [a_i(x-x_i) + b_i(y-y_i) + c_i(z-z_i)]^2}. \tag{10}$$

One might try to analytically optimize $D(x, y, z)$. However, due to the square root, the solution is non-linear. So instead, we optimize the sum of squared distances,

$$E(x,y,z) = \sum_{i=1}^{N} (x-x_i)^2 + (y-y_i)^2 + (z-z_i)^2 - [a_i(x-x_i) + b_i(y-y_i) + c_i(z-z_i)]^2, \tag{11}$$

which is a sum of terms similar to Equation 8, one for each ray. Thus, our goal is to find a point that globally minimizes $E(x, y, z)$. Such a point will be optimal in the sense of minimizing the sum of squared distance.

Derivation of the Optimal Point

To find the optimal point, we differentiate $E(x, y, z)$, set the partial derivatives to zero, and evaluate the critical points.

$$\frac{\partial E(x,y,z)}{\partial x} = \sum_{i=1}^{N} \{2(x-x_i) - 2a_i[a_i(x-x_i) + b_i(y-y_i) + c_i(z-z_i)]\} = 0 \tag{12}$$

$$\frac{\partial E(x,y,z)}{\partial y} = \sum_{i=1}^{N} \{2(y-y_i) - 2b_i[a_i(x-x_i) + b_i(y-y_i) + c_i(z-z_i)]\} = 0 \tag{13}$$

$$\frac{\partial E(x,y,z)}{\partial z} = \sum_{i=1}^{N} \{2(z-z_i) - 2c_i[a_i(x-x_i) + b_i(y-y_i) + c_i(z-z_i)]\} = 0 \tag{14}$$

We seek the optimal point (x, y, z) that satisfies the above equations. Expanding the terms and dividing by 2 yields

$$\sum_{i=1}^{N} \left[x - x_i - a_i^2 x + a_i^2 x_i - a_i b_i y + a_i b_i y_i - a_i c_i z + a_i c_i z_i\right] = 0 \tag{15}$$

$$\sum_{i=1}^{N} \left[y - y_i - a_i b_i x + a_i b_i x_i - b_i^2 y + b_i^2 y_i - b_i c_i z + b_i c_i z_i\right] = 0 \tag{16}$$

$$\sum_{i=1}^{N} \left[z - z_i - a_i c_i x + a_i c_i x_i - b_i c_i y + b_i c_i y_i - c_i^2 z + c_i^2 z_i\right] = 0 \tag{17}$$

Placing the terms involving (x_i, y_i, z_i) on the right side of the equation gives

$$\sum_{i=1}^{N} \left[(1-a_i^2)x - a_i b_i y - a_i c_i z\right] = \sum_{i=1}^{N} \left[(1-a_i^2)x_i - a_i b_i y_i - a_i c_i z_i\right] \tag{18}$$

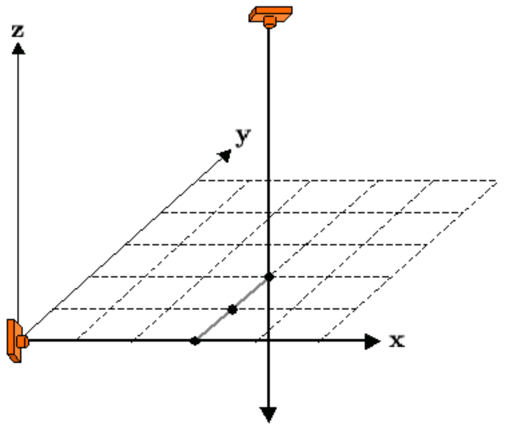


Figure 3: An example.

$$\sum_{i=1}^{N}\left[-a_ib_ix+(1-b_i^2)y-b_ic_iz\right] = \sum_{i=1}^{N}\left[-a_ib_ix_i+(1-b_i^2)y_i-b_ic_iz_i\right] \tag{19}$$

$$\sum_{i=1}^{N}\left[-a_ic_ix-b_ic_iy+(1-c_i^2)z\right] = \sum_{i=1}^{N}\left[-a_ic_ix_i-b_ic_iy_i+(1-c_i^2)z_i\right] \tag{20}$$

Next, we write this in matrix form

$$\begin{bmatrix} \sum_i(1-a_i^2) & -\sum_i a_ib_i & -\sum_i a_ic_i \\ -\sum_i a_ib_i & \sum_i(1-b_i^2) & -\sum_i b_ic_i \\ -\sum_i a_ic_i & -\sum_i b_ic_i & \sum_i(1-c_i^2) \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \sum_i\left[(1-a_i^2)x_i-a_ib_iy_i-a_ic_iz_i\right] \\ \sum_i\left[-a_ib_ix_i+(1-b_i^2)y_i-b_ic_iz_i\right] \\ \sum_i\left[-a_ic_ix_i-b_ic_iy_i+(1-c_i^2)z_i\right] \end{bmatrix} \tag{21}$$

This expression is of the form $A\mathbf{x} = \mathbf{b}$. For each ray i, the starting point of the ray (x_i, y_i, z_i) and the ray direction $\hat{\mathbf{d}}_i = a_i\hat{\mathbf{x}} + b_i\hat{\mathbf{y}} + c_i\hat{\mathbf{z}}$ are known. Thus, all the terms in the matrix A and the vector $\mathbf{b}$ are known. We compute these matrices, and solve for $\mathbf{x}$,

$$\mathbf{x} = A^{-1}\mathbf{b}. \tag{22}$$

The point $\mathbf{x}$ then, is the point that is is closest to all of the rays in the sense of minimizing the sum of squared distance.

Example

In this section, we consider a simple example that shows how this approach works for triangulation. While this example computes the optimal point using two images, the approach works for an arbitrary number of images.

Suppose we have two cameras as shown in Figure 3. Camera 1 is centered at the origin, so $(x_1, y_1, z_1) = (0, 0, 0)$. Camera 2 is centered at the point $(x_2, y_2, z_2) = (3, 2, 5)$. A correspondence is found in the two images, and in each image a ray is back-projected from the camera center through the pixel in the image plane, as shown in the figure. The ray from camera 1 has a direction $(a_1, b_1, c_1) = (1, 0, 0)$, parallel to the x-axis, and the ray from camera 2 has a direction $(a_2, b_2, c_2) = (0, 0, -1)$, parallel to the negative z-axis. Our goal is to find the point in 3D space that is closest to both rays. By inspection, we expect the solution to this problem to be $(3, 1, 0)$.

Using the matrix equation in the previous section, we get

$$\begin{bmatrix} 1-a_1^2+1-a_2^2 & -a_1b_1-a_2b_2 & -a_1c_1-a_2c_2 \\ -a_1b_1-a_2b_2 & 1-b_1^2+1-b_2^2 & -b_1c_1-b_2c_2 \\ -a_1c_1-a_2c_2 & -b_1c_1-b_2c_2 & 1-c_1^2+1-c_2^2 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} =$$

$$\begin{bmatrix} (1-a_1^2)x_1 - a_1b_1y_1 - a_1c_1z_1 + (1-a_2^2)x_2 - a_2b_2y_2 - a_2c_2z_2 \\ -a_1b_1x_1 + (1-b_1^2)y_1 - b_1c_1z_1 - a_2b_2x_2 + (1-b_2^2)y_2 - b_2c_2z_2 \\ -a_1c_1x_1 - b_1c_1y_1 + (1-c_1^2)z_1 - a_1c_1x_2 - b_1c_1y_2 + (1-c_1^2)z_2 \end{bmatrix} \tag{23}$$

Plugging in known values gives

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 4 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 3 \\ 2 \\ 0 \end{bmatrix} \tag{24}$$

which yields the desired solution of

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \\ 0 \end{bmatrix} \tag{25}$$

Analysis

When Is a Unique Solution Not Possible?

Equation 22 shows that if the matrix A is invertible, then a unique solution exists. In this subsection we determine under what circumstances A becomes singular, resulting in no unique solution. To do this, we first compute the determinant of A.

$$|A| = \begin{vmatrix} \sum_i(1-a_i^2) & -\sum_i a_ib_i & -\sum_i a_ic_i \\ -\sum_i a_ib_i & \sum_i(1-b_i^2) & -\sum_i b_ic_i \\ -\sum_i a_ic_i & -\sum_i b_ic_i & \sum_i(1-c_i^2) \end{vmatrix} \tag{26}$$

Each element of A is a sum from $i = 1 \cdots N$. Evaluating and simplifying the terms in this determinant for general N is rather challenging. Using the constraint $a_i^2 + b_i^2 + c_i^2 = 1$, with some work it is possible to show that this determinant can be rewritten as

$$|A| \quad = \quad \frac{1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N}\sum_{\substack{k=1\\ k\neq i,j}}^{N}(b_ic_j - b_jc_i)^2(1-a_k^2) +$$

$$\begin{aligned}
&\frac{1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\j\neq i}}^{N}\sum_{\substack{k=1\\k\neq i,j}}^{N}(a_ic_j - a_jc_i)^2(1-b_k^2) + \\
&\frac{1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\j\neq i}}^{N}\sum_{\substack{k=1\\k\neq i,j}}^{N}(a_ib_j - a_jb_i)^2(1-c_k^2) + \\
&\sum_{i=1}^{N}\sum_{\substack{j=1\\j\neq i}}^{N}\sum_{\substack{k=1\\k\neq i,j}}^{N}(a_ib_jc_k - a_kb_ic_j)^2 \qquad (27)
\end{aligned}$$

A unique solution to this problem cannot be found when Equation 27 equals zero. $|A|$ is the sum of non-negative expressions, as each ray direction is normalized. Thus, the only way Equation 27 can equal zero is if each expression evaulates to zero. This fact leads to the following theorem:

Theorem 1 *A unique solution exists except when all the rays are either parallel or anti-parallel. That is,*

$$\hat{\mathbf{d}}_i = \begin{cases} a\hat{\mathbf{x}} + b\hat{\mathbf{y}} + c\hat{\mathbf{z}} \\ \quad\quad or \\ -a\hat{\mathbf{x}} - b\hat{\mathbf{y}} - c\hat{\mathbf{z}} \end{cases} \qquad (28)$$

Proof: To prove Theorem 1, we must show that each term in Equation 27 equals zero only when all the rays are parallel or anti-parallel. The only way the expression $(1 - a_k^2)$ can be zero is if each normalized ray direction is either $(1, 0, 0)$ or $(-1, 0, 0)$, i.e., all the rays are parallel or anti-parallel. A similar result holds for the $(1 - b_k^2)$ and $(1 - c_k^2)$ expressions in Equation 27. For the expressions $(b_ic_j - b_jc_i)$, $(a_ic_j - a_jc_i)$, and $(a_ib_j - a_jb_i)$ to be zero, we must have

$$\begin{aligned}
b_ic_j &= b_jc_i \qquad (29)\\
a_ic_j &= a_jc_i \qquad (30)\\
a_ib_j &= a_jb_i \qquad (31)
\end{aligned}$$

Using the fact that $a_i^2 + b_i^2 + c_i^2 = 1$, we can rewrite Equation 31 as

$$\begin{aligned}
a_ib_j &= a_jb_i \qquad (32)\\
a_i\sqrt{1 - a_j^2 - c_j^2} &= a_j\sqrt{1 - a_i^2 - c_i^2} \qquad (33)\\
a_i^2(1 - a_j^2 - c_j^2) &= a_j^2(1 - a_i^2 - c_i^2) \qquad (34)\\
a_i^2 - a_i^2c_j^2 &= a_j^2 - a_j^2c_i^2 \qquad (35)
\end{aligned}$$

Since $(a_ic_j - a_jc_i)^2 = 0$, we know that $a_i^2c_j^2 = 2a_ia_jc_ic_j - a_j^2c_i^2$. We substitute this into Equation 35, yielding

$$\begin{aligned}
a_i^2 - 2a_ia_jc_ic_j + a_j^2c_i^2 &= a_j^2 - a_j^2c_i^2 \qquad (36)\\
a_i^2 + 2a_j^2c_i^2 &= a_j^2 + 2a_ia_jc_ic_j \qquad (37)
\end{aligned}$$

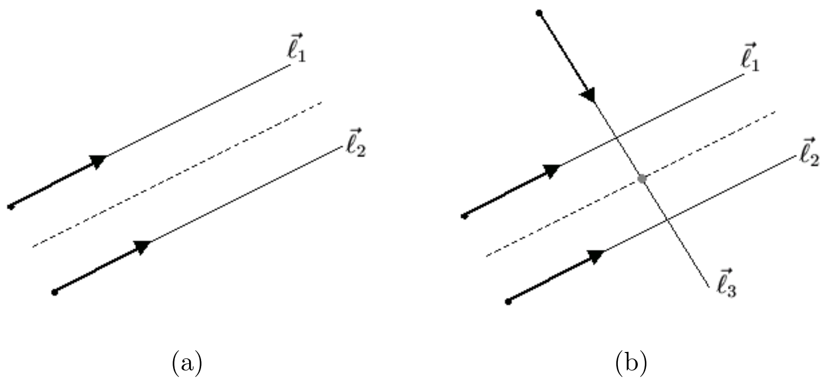


Figure 4: No unique solution exists when all rays are parallel or anti-parallel. In (a), any point that is along the line that is equidistant from the parallel rays will be optimally close to the two rays. However, if just one ray is not parallel, as in (b), a unique solution exists, depicted with a gray dot.

Using Equation 30, we get

$$a_i^2 = a_j^2 \tag{38}$$

Similar results hold for b and c, resulting in

$$\begin{aligned} a_i &= \pm a_j && (39)\\ b_i &= \pm b_j && (40)\\ c_i &= \pm c_j && (41)\end{aligned}$$

In order for Equations 29 to 31 to be satisfied, the signs on the above three equations must either all be positive or negative. Thus, the first three terms of Equation 27 are zero if and only if all the rays are parallel or anti-parallel.

The fourth term of Equation 27 can be re-expressed to look one of the Equations 29 to 31. For example, using the fact that $a_i b_j = a_j b_i$,

$$\begin{aligned} (a_i b_j)c_k &= a_k b_i c_j && (42)\\ (a_j b_i)c_k &= a_k b_i c_j && (43)\\ a_j c_k &= a_k c_j && (44)\end{aligned}$$

Thus, the fourth term of Equation 27 is zero when the other three terms of Equation 27 are zero, namely when all the rays are parallel or anti-parallel. This concludes the proof. $\triangle$

Theorem 1 is consistent with one's intuition. When all the rays are parallel (or anti-parallel), one would not expect that a unique solution exists, as shown in Figure 4 (a). In this case, any point along a line that is equidistant from the parallel rays will be a valid solution. This equidistant line is depicted with a dotted line. However, if at least one of the rays is not parallel to the others, then a unique solution exists, as shown in in Figure 4 (b).

Discussion of Extremum

In the derivation above, we found an extremal point of the sum of squared distance $E(x, y, z)$. By nature of the problem, one can assume that this extremal point is a minimum and not a maximum. For the finicky reader, we prove in this subsection that the extremal point is indeed a minimum.

Theorem 2 *The extremal point found by this method minimizes the sum of squared distance.*

Proof: To show that the extremal point is indeed a minimum, we must analyze the matrix of second partial derivatives [1],

$$D = \begin{bmatrix} \frac{\partial^2 E}{\partial x^2} & \frac{\partial^2 E}{\partial x \partial y} & \frac{\partial^2 E}{\partial x \partial z} \\ \frac{\partial^2 E}{\partial x \partial y} & \frac{\partial^2 E}{\partial y^2} & \frac{\partial^2 E}{\partial y \partial z} \\ \frac{\partial^2 E}{\partial x \partial z} & \frac{\partial^2 E}{\partial y \partial z} & \frac{\partial^2 E}{\partial z^2} \end{bmatrix} = \begin{bmatrix} 2\sum_i (1-a_i^2) & -2\sum_i a_i b_i & -2\sum_i a_i c_i \\ -2\sum_i a_i b_i & 2\sum_i (1-b_i^2) & -2\sum_i b_i c_i \\ -2\sum_i a_i c_i & -2\sum_i b_i c_i & 2\sum_i (1-c_i^2) \end{bmatrix} \tag{45}$$

and show that the determinants of the upper-left 1x1 submatrix, upper-left 2x2 submatrix, and D are all strictly positive.

First, we compute the determinant of D_1, the upper-left 1x1 submatrix,

$$|D_1| = 2\sum_i (1-a_i^2) \tag{46}$$

Since a_i is a component from a normalized vector, the value of each $(1-a_i^2)$ term must be non-negative. Thus, D_1 must be non-negative, since it is a sum of non-negative terms. The only situation for which D_1 could be zero is if each $a_i = \pm 1$. In that case, all rays would be parallel or anti-parallel, with directions $(a_i, b_i, c_i) = (1, 0, 0)$ or $(a_i, b_i, c_i) = (-1, 0, 0)$. As shown earlier, we cannot expect a unique solution for this case, so we do not care about the extremal point. Thus, D_1 is strictly positive except when the all rays are parallel or anti-parallel.

Next, we compute the determinant of D_2, the upper-left 2x2 submatrix,

$$|D_2| = \begin{vmatrix} 2\sum_i (1-a_i^2) & -2\sum_i a_i b_i \\ -2\sum_i a_i b_i & 2\sum_i (1-b_i^2) \end{vmatrix} \tag{47}$$

Using the fact that $a_i^2 + b_i^2 + c_i^2 = 1$, this determinant can be expressed as

$$|D_2| = 4N\sum_{i=1}^{N}(c_i^2) + 2\sum_{i=1}^{N}\sum_{\substack{j=1 \\ j\neq i}}^{N}(a_i b_j - a_j b_i)^2 \tag{48}$$

Since D_2 is a sum of non-negative terms, it must also be non-negative. The only way D_2 can be zero is if all $c_i = 0$, and each $a_i b_j = a_j b_i$ for $i \neq j$. This can only happen when all the rays are parallel or anti-parallel, since

$$a_i b_j = a_j b_i \tag{49}$$

$$\begin{aligned}
\sqrt{1-b_i^2-c_i^2}\,b_j &= \sqrt{1-b_j^2-c_j^2}\,b_i && (50)\\
\sqrt{1-b_i^2}\,b_j &= \sqrt{1-b_j^2}\,b_i && (51)\\
(1-b_i^2)b_j^2 &= (1-b_j^2)b_i^2 && (52)\\
b_j^2-b_i^2b_j^2 &= b_i^2-b_i^2b_j^2 && (53)\\
b_i^2 &= b_j^2 && (54)
\end{aligned}$$

Likewise, D_2 can only be zero for $a_i^2 = a_j^2$. This equation, along with Equations 49 and 54 are true only when the rays are parallel or anti-parallel. Thus, D_2 is strictly positive except when all rays are parallel or anti-parallel.

Finally, we must show that the determinant of D is non-negative. Comparing equations 26 and 45, we note that $|D| = 8|A|$. In the previous section, we showed that the determinant of A was strictly positive except when all rays were parallel or anti-parallel. Therefore, this result also applies to the determinant of D. This concludes the proof. $\triangle$

Thus, the extremal point P found by our optimal ray intersection algorithm minimizes the sum of squared distance to each ray.

Source Code

Source code that computes the optimal ray intersection in the sense of minimizing the sum of squared distance is available on the web. Please visit
http://www.ece.gatech.edu/users/slabaugh/code/opray/ for source code in C and Matlab.

Conclusion

In this report we have developed a simple linear algorithm that finds a point P in 3D space given a correspondence across N images, where $N \geq 2$. Point P minimizes the sum of squared distance between itself and N rays back-projected from the images into 3D space. Our algorithm runs in O(N) time.

Future Work

One future direction for this research is to incorporate a senstivity analysis. We are interested in exploring how errors in image space (due to quantization or correspondence) and camera calibration affect the certainty of the 3D point computed by our algorithm. Furthermore, we are interested in investigating ways to incorporate weighting factors to downweight viewpoints that have small baselines.

Acknowledgements

We thank Greg Turk for use and redistribution of his invert.c routine. Additionally, we thank Jordan Rosenthal for his help in evaluating the determinant of the matrix A, and

John Glotzbach for his assistance in proving that the extremal point P is a minimum. Support for this research came through Hewlett-Packard.

References

[1] R. Bartle, *The Elements of Real Analysis*, Second Edition, John Wiley and Sons, New York, 1976, pp. 397 - 401.

[2] J. Marsden and A. Weinstein, *Calculus III*, Springer-Verlag, New York, 1985, pp. 670 - 671.

[3] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon, "Bundle Adjustment – A Modern Synthesis", *Vision Algorithms: Theory and Practice,* Proceedings of the International Workshop on Vision Algorithms, Corfu, September 21-22, 1999, pp. 298 - 372.

[4] E. Trucco and A. Verri, *Introductory Techniques for 3-D Computer Vision*, Prentice-Hall, New Jersey, 1998, pp. 162 - 163.