Iterative Solution for Rigid-Body Point-Based Registration with Anisotropic Weighting

Ramya Balachandran¹ and J. Michael Fitzpatrick²

¹Department of Otolaryngology, Vanderbilt University Medical Center
²Department of Electrical Engineering and Computer Science, Vanderbilt University
Nashville, TN 37212
ramya.balachandran@vanderbilt.edu, j.michael.fitzpatrick@vanderbilt.edu

ABSTRACT

Rigid-body, point-based registration is commonly used for image-guided systems. Fiducial markers that can be localized in image and physical space are attached to patient anatomy. The fiducial marker locations in the two spaces are used to obtain the physical-to-image registration. It is a common practice to obtain physical positions via optical systems, whose localization error is anisotropic. Furthermore, the positions are often reckoned relative to a coordinate reference frame (CRF) that is rigidly attached to the patient. The use of a CRF enables patient movement relative to the tracking system, but it tends to exacerbate the anisotropy. It is common practice to ignore the localization anisotropy and employ a closed-form solution, which is available for isotropic weighting but not for anisotropic weighting. Iterative methods are available for anisotropic weighting but are quite complex. We present a new iterative algorithm for anisotropic weighting that is simple, intuitive, and has only one adjustable parameter. We show using simulations that our algorithm is more accurate than the isotropic solution for anisotropic localization error. In particular, we show that the new algorithm reduces target registration error when anisotropic localization error is present. When all the localization errors are isotropic, the new algorithm performs as well as the closed-form solution.

KEYWORDS: rigid registration, fiducial, anisotropic weighting, image guidance.

1. INTRODUCTION

Point-based rigid registration is commonly used for image-guided systems. One set of point is to be registered to another set of corresponding points by means of a rigid transformation of the first set. Surgical guidance systems based on preoperative images, such as CT or MR, typically employ a tracking system to map points from image to the physical space of the operating room. For neurosurgery and ear surgery, because of the rigidity of the skull, the point mapping is typically a rigid transformation. The transformation is usually based on fiducial markers that are attached to the head before imaging and remain attached until the procedure begins. A fiducial point set is obtained by localizing each fiducial marker both in the image and in the operating room. Then, a least-squares problem is solved to register the image points to their corresponding physical points, and the result is the rigid transformation. Fiducial localization error (FLE) causes registration error, and the least-squares approach is used to obtain the transformation that minimizes this error in fiducial alignment. The least-squares solution has a closed form when FLE is isotropic [1], [2], but there is no known closed form for the case of anisotropic FLE, for which anisotropic weighting is required. With fiducials that are larger than a voxel size, FLE in the image space can be made highly isotropic, but FLE in physical space may suffer from anisotropy due to tracking [3]. For example, physical positions are often acquired via optical systems, whose localization error is anisotropic. Furthermore, the positions are often reckoned relative to a coordinate reference frame (CRF) that is rigidly attached to the patient. The use of a CRF enables patient movement relative to the tracking system during the procedure, but it tends to exacerbate the anisotropy [3]. Thus, a solution that allows for anisotropy in one space is of value. A few approximate solutions are available [4]-[8], but they tend to be complicated. We present in this paper an alternative approach that is considerably simpler and more intuitive. We show that a substantial improvement in the target registration error (TRE) is achieved when the new algorithm is used in place of the commonly used closed-form solutions when FLE is anisotropic.
2. METHOD

The problem is to find the rigid transformation that minimizes the anisotropically weighted fiducial registration error (FRE), which is square root of the mean of the sum of squares of weighted individual fiducial registration errors. We let $X = \{x_i\}, i = 1...N$ be the set of 3-by-1 fiducial points to be transformed (e.g., from image space), which we call the “moving” fiducials and $Y = \{y_i\}, i = 1...N$ be the set of corresponding fiducial points to which they are to be registered (e.g., in physical space), which we call the “stationary” fiducials. Our goal is to find the 3-by-3 rotation matrix $R$ and 3-by-1 translation vector $t$ that minimize

$$FRE^2 = \frac{1}{N} \sum_{i=1}^{N} \left| W_i (R x_i + t - y_i) \right|^2,$$

where $W_i$ is a 3-by-3 matrix of weights for fiducial $i$. The problem is made difficult by the nonlinear constraint on the rotation matrix, namely, $R^t R = I$, where $I$ is the identity.

The weights account for the variation in the localization accuracy among the fiducials, a variation with respect to the direction and with respect to the markers. We will refer to the former variation as “anisotropy” and the latter as “inhomogeneity”. If $W_i = W_j$ for all $i, j$, then FLE is homogeneous. We assume that FLE for each fiducial is normally distributed with zero mean and can be resolved into three uncorrelated components along a set of orthogonal principal axes, with standard deviations $\sigma_{i\alpha}, \alpha = 1,2,3$. If $\sigma_{i\alpha} = \sigma_{i\beta}$, the problem reduces to the isotropic problem, which is easily solved by closed-form methods, both for the homogeneous and the inhomogeneous cases [9]. We refer to the anisotropic case for which there is no known closed-form solution. This is the situation that arises in surgical navigation, in which the localization error of the physical tracking system in the operating room suffers from a relatively larger standard deviation in the direction from the camera to the fiducial than in the perpendicular directions. The weighting matrix has the form

$$W_i = V_i^t \text{diag}(\sigma_{11}^{-1}, \sigma_{12}^{-1}, \sigma_{13}^{-1}) V_i$$

where $V_i V_i = I$, the 3x3 identity matrix. The columns of $V_i$ are the principal axes of the FLE for fiducial $i$. For many tracking systems, the distance from the camera to the fiducials is much larger than distances between the fiducials. In this case, the principle axes are approximately the same for all fiducials. Thus, $V_i \approx V$. We will for simplicity treat that case. With that assumption, we may, without loss of generality, assume that $V = I$. We can accomplish that by reorienting our coordinate system to that of the camera, which results in the replacement of each $x_i$ with $V x_i$, and each $y_i$ with $V y_i$. We will henceforth work in that coordinate system.

2.1 Algorithm

Our method is iterative, and our strategy is to replace, at each iterative step, the exact, nonlinear problem with a simple, linear problem, which can be solved exactly by linear algebra. The technique of solving a nonlinear problem by solving a simpler linear problem is common [10], [11]. The simplification involves replacing the rotation matrix by an approximation to the rotation matrix that is subject only to a linear constraint. Thus, the exact solution to the linear algebra problem gives an approximate solution to the registration problem. However, we then replace the approximate matrix with the rotation matrix that is closest in the least-squares sense to the approximate matrix, and we apply that rotation matrix to the moving fiducials to bring them incrementally closer to the stationary fiducials. In summary, we repeatedly (1) solve the linearized problem, (2) find the closest rigid transformation to the linear solution, and (3) apply the rigid transformation to the moving fiducials, stopping when the fiducial movement is below a threshold.

At each stage of the iteration, a set of points is transformed to a new set of points. At iteration stage $n$, the points $x_i^{(n-1)}$ are transformed to the points $x_i^{(n)}$. Before the iteration begins, we perform initialization by solving Eq. (1) for isotropic, homogeneous FLE, i.e. omitting $W_i$. The solution is effected by means of the closed-form solution given by Algorithm...
8.1 of [9]. The resulting rotation matrix and translation vector are labeled $R^{(0)}$ and $t^{(0)}$. We calculate $x_i^{(0)} = R^{(0)}x_i + t^{(0)}$ for $i = 1, ..., N$. The rotation and translation of our solution are then initialized as $R = R^{(0)}$ and $t = t^{(0)}$.

After initialization, each stage $n$ of the iteration comprises the following steps:

1. In Eq. (1), $x_i$ is replaced by $x_i^{(n-1)}$, and $R$ is replaced by an approximate rotation operator, $I + \Delta \Theta^{(n)}$, where $\Delta \Theta^{(n)} = -\Delta \Theta^{(n)}$. We then minimize the following approximation of Eq. (1):

$$FRE^2 = \frac{1}{N} \sum_{i=1}^{N} \left( (I + \Delta \Theta^{(n)})x_i^{(n-1)} + \Delta t^{(n)} - y_i \right)^T W_i^2 \left( (I + \Delta \Theta^{(n)})x_i^{(n-1)} + \Delta t^{(n)} - y_i \right).$$

(3)

The $\Delta \Theta^{(n)}$ and translation vector, $\Delta t^{(n)}$, that together minimize (3), can be found exactly. The solution method is given below in Sec 2.2.

2. $\Delta \Theta = \left( \Delta \Theta^{(n-1)} + \Delta \Theta^{(n)} \right)/2$ and $\Delta t = \left( \Delta t^{(n-1)} + \Delta t^{(n)} \right)/2$. (Step 2 is skipped for $n = 1$.)

3. Perform singular value decomposition of $I + \Delta \Theta$: $U \Lambda V^T = I + \Delta \Theta$, where $U$ and $V$ are rotation matrices and $\Lambda$ is a diagonal matrix with non-negative elements.

4. Set $R^{(n)} = UV^T$. (As is shown below in Sec 2.3, this matrix is the proper rotation that is closest in the least-squares sense to $I + \Delta \Theta$.)

5. An updated rotation matrix is calculated: $R = R^{(n)}R$.

6. An updated translation vector is calculated: $t = R^{(n)}t + \Delta t$.

7. An updated point set is calculated: $x_i^{(n)} = Rx_i + t$, for $i = 1, ..., N$.

8. The relative change in point configuration is calculated:

$$\Delta \chi^2 = \frac{1}{N} \sum_{i=1}^{N} \left( x_i^{(n)} - x_i^{(n-1)} \right)^2 / \sum_{i=1}^{N} \left( x_i^{(n)} - \bar{x}^{(n)} \right)^2,$$

where $\bar{x}^{(n)}$ is the centroid of the point configuration $\{x_i^{(n)}\}$.

9. If $\Delta \chi > \text{threshold}$, $n$ is set equal to $n + 1$ and we return to Step 1.

### 2.2 Solution of the approximate equation

In Step 1 of our iterative algorithm, we must find $\Delta \Theta^{(n)}$ and $\Delta t^{(n)}$ to minimize the expression for $FRE^2$ in Eq. (3). We note that this minimization is equivalent to finding the least-squares solution to the set of $3N$ equations:

$$W_i \Delta \Theta^{(n)} x_i^{(n-1)} + W_i \Delta t^{(n)} = W_i \Delta x_i^{(n-1)},$$

(4)

where $\Delta x_i^{(n-1)} = y_i - x_i^{(n-1)}$. The unknowns in these equations are the six nonzero elements of the 3-by-3 antisymmetric matrix $\Delta \Theta^{(n)}$ and the three elements of the vector $\Delta (n)$. We note that for an antisymmetric matrix the diagonal elements all equal zero and that among the off-diagonal elements $\Delta \Theta_{ij}^{(n)} = -\Delta \Theta_{ji}^{(n)}$. As a result of these restrictions, there are only three independent unknowns in $\Delta \Theta^{(n)}$: namely, $\Delta \Theta_{32}^{(n)}$, $\Delta \Theta_{13}^{(n)}$, and $\Delta \Theta_{21}^{(n)}$. The meaning of these three elements can be understood by recalling that $I + \Delta \Theta^{(n)}$ is an approximation of a rotation $R$. If the angle of rotation of $R$ about its axis is small, then the movement, $Rx_i^{(n)} - x_i^{(n)}$, can be approximated as a cross product between the axis of rotation of $R$ and the vector $x_i^{(n)}$. If we define the vector $\Delta \Theta$, whose elements are $\Delta \Theta_1 = \Delta \Theta_{32}^{(n)}$, $\Delta \Theta_2 = \Delta \Theta_{13}^{(n)}$, and $\Delta \Theta_3 = \Delta \Theta_{21}^{(n)}$, then for small rotations, the axis of rotation lies approximately along $\Delta \Theta$, and the angle of rotation is approximately equal to the length of $\Delta \Theta$, and

$$Rx_i^{(n)} - x_i^{(n)} = \Delta \Theta \times x_i^{(n)}.$$
We can use $\Delta \theta$ to transform Eqs. (4) into the canonical form

$$Cq = e,$$

(6)

where $C$ is a $3N$-by-6 matrix, $q$ is a 6-by-1 vector of unknowns, and $e$ is a $3N$-by-1 column vector. The elements of $q$ are as follows:

$$q_1 = \Delta \theta_1, q_2 = \Delta \theta_2, q_3 = \Delta \theta_3, q_4 = \Delta t_4, q_5 = \Delta t_2, q_6 = \Delta t_3.$$  

(7)

To give the elements of $C$ and $e$, we need to specify the $jk$ element of $W$, which we write with three subscripts: $W_{jk,i}^\sigma$, separating the third subscript with a comma to emphasize that it is not a matrix index, but a fiducial index. Additionally, we need to specify element $j$ of $\Lambda_i^{(n)}$, which we write with two subscripts: $\Lambda_{jk,i}^{(n)}$, here separating the second subscript with a comma to emphasize that it is not a vector index, but a fiducial index. With this notation established, we find, by means of a detailed inspection of Eq. (4), that

$$C_{3(i-1)+j,1} = -W_{j2,i}^\sigma x_{3i}^{(n)} + W_{j3,i}^\sigma x_{2i}^{(n)},$$

$$C_{3(i-1)+j,2} = W_{j2,i}^\sigma x_{3i}^{(n)} - W_{j3,i}^\sigma x_{1i}^{(n)},$$

$$C_{3(i-1)+j,3} = -W_{j1,i}^\sigma x_{2i}^{(n)} + W_{j2,i}^\sigma x_{1i}^{(n)},$$

$$C_{3(i-1)+j,4} = W_{j1,i},$$

$$C_{3(i-1)+j,5} = W_{j2,i},$$

$$C_{3(i-1)+j,6} = W_{j3,i},$$

(8)

and

$$e_{3(i-1)+j,1} = W_{j1,i} \Lambda_{1,i}^{(n)} + W_{j2,i} \Lambda_{2,i}^{(n)} + W_{j3,i} \Lambda_{3,i}^{(n)}.$$  

(9)

With these definitions, the solution to Eq. (3) is found by solving Eq. (6) for $q$ by using any appropriate numerical method, and then setting

$$\Delta \Theta = \begin{bmatrix} 0 & -q_3 & q_2 \\ q_3 & 0 & -q_1 \\ -q_2 & q_1 & 0 \end{bmatrix},$$

(10)

and

$$\Delta t = [q_4 \ q_5 \ q_6]^T.$$  

(11)

Any appropriate numerical method may be employed to solve Eq. (6), such as the pseudo-inverse or singular-value decomposition of $C$. For our simulations, we used the “backslash operator” in Matlab (The Mathworks, Natick, MA): $q = C \backslash e$. In the appendix the entire Matlab program is included.

2.3 Finding the closest rotation matrix

In Step 4 of our iterative algorithm, we set $R^{(n)} = UV'$, where $U$ and $V$ are rotation matrices obtained from the singular value decomposition of the approximate rotation matrix: $U \Lambda V' = I + \Delta \Theta$. We can prove that $R^{(n)}$ found in Step 3 of our iterative algorithm is the closest rotation matrix to $I + \Delta \Theta$ in the least squares sense as follows: We let $M = I + \Delta \Theta$. Then we let $R$ be the closest rotation matrix to $M$ and form the difference $E = M - R$. We let $|E|^2$ equal the sum of squares of the elements of $E$. Thus,
\[ |E|^2 = \sum_{i,j} (M_{ij} - R_{ij})^2 \]

\[ = \text{trace} (M - R)^T (M - R) \]

\[ = \text{trace} M' M - 2 \text{trace} M' R + \text{trace} (I). \] (12)

The \( R \) we seek is the one that minimizes \( |E|^2 \). From the last line of Eq. (12), we see that \( |E|^2 \) is minimized when \( \text{trace} (M' R) \) is maximized. To maximize this trace, we employ the singular value decomposition of \( M: U \Lambda V^T = M \), where \( U \) and \( V \) are rotation matrices and \( \Lambda \) is diagonal and has nonnegative elements. Using this formulation for \( M \) in \( \text{trace} (M' R) \), we have that

\[ \text{trace} (M' R) = \text{trace} (V \Lambda U^T R) \]

\[ = \text{trace} (\Lambda U^T RV) \]

\[ = \text{trace} (\Lambda Z) \]

\[ = \sum_{i=1}^{3} \Lambda_{ii} Z_{ii}, \] (13)

where \( Z = U^T RV \). In going from the first line to the second in Eq. (13), we used the property of the trace that \( \text{trace} (AB) = \text{trace} (BA) \). We note that, by virtue of its being a product of rotation matrices, \( Z \) is also a rotation matrix. The maximum value for any element of a rotation matrix is 1. Since each \( \Lambda_{ii} \) is nonnegative, the maximum of the sum in the last line of Eq. (13) is reached when \( Z_{ii} = 1 \) for \( i = 1, 2, 3 \). Thus, \( Z \) is the 3-by-3 identity matrix \( I \). Therefore, \( I = U^T RV \). By multiplying this equation on the left by \( U \) and on the right by \( V' \), we find that \( R = UV' \), which completes the proof.

We can prove that \( R \) is proper as follows. A proper rotation has a nonnegative determinant. Thus, we need to show that the determinant of \( R \) is nonnegative. We use of this property of determinants: For all products \( AB \), \( \det (AB) = \det (A) \det (B) \). From this property we have that

\[ \det (I + \Delta \Theta) = \det (U) \det (\Lambda) \det (V') \]

\[ = \det (\Lambda) \det (UV') \]

\[ = \det (\Lambda) \det (R). \] (14)

We note that the determinant of a diagonal is the product of its diagonal elements, all of which are nonnegative for \( \Lambda \) by virtue of the properties of singular value decomposition. Thus, the sign of the determinant of \( R \) is equal to the sign of \( \det (I + \Delta \Theta) \). We find an expression for this latter determinant directly:

\[ \det (I + \Delta \Theta) = \begin{vmatrix}
1 & -\Delta \theta_3 & \Delta \theta_2 \\
\Delta \theta_3 & 1 & -\Delta \theta_1 \\
-\Delta \theta_2 & \Delta \theta_1 & 1
\end{vmatrix} \]

\[ = 1 + \Delta \theta_1^2 + \Delta \theta_2^2 + \Delta \theta_3^2, \] (15)

which is nonnegative. Thus, the determinant of \( R \) is also nonnegative, which completes the proof.
3. SIMULATION

We tested our algorithm by performing computer simulations using Matlab. Four values for $N$ were chosen for the tests: $N = 3, 4, 5, 10$. For each value of $N$, we randomly chose that number of fiducial locations for $X$ from a cube of edge 200 mm with center at the origin. The other corresponding fiducial set, $Y$, was obtained by rotating and translating the $X$ fiducial configuration arbitrarily—10, −20, and 30 degrees rotation about the $x$, $y$, and $z$ axes, and a translation of 40, 10, and 100 mm along these axes—then randomly perturbing each position. In this way we simplify the problem by combining the isotropic localization error in $X$ space into the anisotropic $Y$ space error. $X$ and $Y$ were then registered. Three different registration methods were compared: (1) the closed-form solution with weighting equal to 1 for all the fiducials and directions, (2) the closed-form solution with weighting for fiducial $i$ set to $\left(\sum_{a=1}^{3} \sigma_{ia}^{2}\right)^{-1/2}$, and (3) the proposed iterative algorithm with $W_i$ defined according to Eq. (2) and a threshold of $10^{-6}$ used for $\Delta X$. (The Matlab implementation of the iteration algorithm—“anisotropic point register”—is provided in the appendix along with other necessary functions.) Three random targets were chosen inside a cube of edge 400 mm centered at the origin, and the target registration error (TRE) was computed for each registration method. 100,000 iterations of perturbing $X$ to get $Y$, computing new registration transformations, and computing TRE values were performed to come up with an overall root-mean-square (RMS) TRE value.

Our algorithm can handle different FLEs for each fiducial and for each direction. Different experiments were performed to study the effect on TRE of using the different algorithms for different cases.

Experiment 1 (homogeneous, anisotropic FLE): A fiducial system with equal FLE in two directions and a different FLE in the third direction for all fiducials was used for this experiment. FLE in the $x$ and $y$ directions for each fiducial were chosen randomly from $\mathcal{N}(0,0.2)$. The FLE in the $z$ direction for all the fiducials were drawn from $\mathcal{N}(0,\text{FLE}_z)$, where $\text{FLE}_z$ varied from 0.1 to 2 mm in steps of 0.2 mm.

Experiment 2 (inhomogeneous, anisotropic FLE): A fiducial system with equal FLE in all directions except for the FLE in one of the direction of certain fiducials was used for this experiment. FLE in the $x$ and $y$ directions for each fiducial were chosen randomly from $\mathcal{N}(0,0.2)$. The FLE in the $z$ direction for each fiducial was chosen randomly from $\mathcal{N}(0,\text{FLE}_z)$ except for the FLE in the $z$ direction for the first and second fiducial which are drawn from $\mathcal{N}(0,\text{FLE}_z)$, where $\text{FLE}_z$ varied from 0.1 to 2 mm in steps of 0.2 mm.

Experiment 3 (homogeneous, anisotropic FLE): A fiducial system with different FLE in the three directions, but same for all fiducials was used for this experiment. FLE in the $x$ and $y$ directions for each fiducial were chosen randomly from $\mathcal{N}(0,0.1)$ and $\mathcal{N}(0,0.2)$ respectively. The FLE in the $z$ direction for all the fiducials were drawn from $\mathcal{N}(0,\text{FLE}_z)$, where $\text{FLE}_z$ varied from 0.1 to 2 mm in steps of 0.2 mm.

Experiment 4 (inhomogeneous, anisotropic FLE): A fiducial system with different FLE in all the three directions, but same for certain fiducials was used for this experiment. FLE in the $x$ and $y$ directions for each fiducial were chosen randomly from $\mathcal{N}(0,0.1)$ and $\mathcal{N}(0,0.2)$ respectively. The FLE in the $z$ direction for each fiducial was chosen randomly from $\mathcal{N}(0,0.3)$, except for the FLE in the $z$ direction for the first and second fiducial which were drawn from $\mathcal{N}(0,\text{FLE}_z)$, where $\text{FLE}_z$ varied from 0.1 to 2 mm in steps of 0.2 mm.

4. RESULTS AND DISCUSSION

Figure 1 shows the results of the simulations of the experiments 1-4 for $N = 3, 4, 5, 10$. Each plot compares the RMS TRE values calculated using the three different registration methods for different values of FLE in the $z$ direction. The method based on singular value decomposition as described in Algorithm 8.1 of [9] was used for the first two registration methods. When the fiducial system is homogeneous, meaning FLE in a given direction is same for all fiducials, the results of the first two registration methods are the same. Thus, for experiments 1 and 3, the results of the first two registration methods are the same and they are labeled as SVD in the plots to represent the solution based on singular value decomposition. For experiments 1 and 2, when $\text{FLE}_z$ is close to 0.2, which is the FLE for other directions.
**Figure 1.** RMS TRE versus FLE, using different registrations for experiments 1-4 when (a)-(d) $N = 3$, (e)-(h) $N = 4$, (i)-(l) $N = 5$, and (m)-(p) $N = 10$. Vertical dashed lines show the homogeneous and isotropic cases.
and fiducials, the registration problem reduces to the isotropically weighted problem and the RMS TRE matches for all the methods (shown by vertical dashed line in the plots).

We note that the use of proper weighting and the new simple iterative approach improves the TRE in the presence of anisotropic error. We see that the SVD solution with inhomogeneous weighting (registration method 2) performs better than the SVD solution with equal weighting (registration method 1), but the new iterative method is superior to both. The TRE improvement depends on the inhomogeneous and anisotropic nature of the system. In our experiments, increasing \( N \) reduces the inhomogeneous and anisotropic nature of the system. So the TRE improvement also reduces relatively.

The number of iterations required for \( \Delta X \) to meet the threshold criterion decreased monotonically with increasing \( N \). The mean number of iterations in our simulations decreased from 19 iterations for \( N = 3 \) to 6 iterations for \( N = 10 \). The maximum time taken by the new algorithm for computing the registration transformation was 18 ms, which makes it possible to use this algorithm for real-time applications.

5. CONCLUSION

We presented in this paper a new simple iterative algorithm for solving the point-based registration problem when the FLE is anisotropic. In contrast to the existing approximate solutions, the new algorithm is more intuitive, easy to understand, and implement. It has only one adjustable parameter—the threshold for the stopping criterion. Through simulations we have shown a substantial improvement in TRE using the new algorithm in place of the commonly used closed-form solutions for systems with anisotropic localization error. When the error is completely isotropic, the new algorithm produces the same result as the closed-form solution, as would be expected.

Image-guidance systems often employ a CRF to enable patient movement relative to the tracking system. It is known that anisotropic localization error is introduced while localizing points relative to CRF [3]. Thus, these results show that this new algorithm can be expected to improve the accuracy of these systems as well.

ACKNOWLEDGEMENTS

This work was supported by the grant R21 EB006044-01A1 from the National Institute of Biomedical Imaging and Bioengineering.

APPENDIX

The following are Matlab functions that implement the algorithm presented in this paper:

```matlab
function [R,t,FRE,n] = anisotropic_point_register(X,Y,W,threshold)
% X is the moving set, which is registered to the static set Y. Both are 3
% by N, where N is the number of fiducials. W is a 3-by-3-by-N array, with
% each page containing the weighting matrix for the Nth pair of points.
% THRESHOLD is the size of the change to the moving set above which the
% iteration continues.
%
% Creation:
% R. Balachandran and J. M. Fitzpatrick
% December 2008
if nargin<3,error('X and Y must be given as input');end
if size(X,1)~=3 && size(Y,1)~=3,error('X and Y must be 3 by N.');end
N = size(X,2);
if size(Y,2)~=N,error('X and Y must have the same number of points.');end
% Initial estimate of the transformation assumes anisotropy:
[R,t,FRE] = point_register(X,Y);
```
if nargin<3 % if W not given, then give the isotropic solution
    n = 0;
    return
end
if nargin<4,threshold = 1e-6;end
n = 0; % iteration index = 0;
config_change = Inf;
Xold = R*X+repmat(t,1,N);
while (config_change>threshold)
    n = n+1;
    C = C_maker(Xold,W);
    e = e_maker(Xold,Y,W);
    q = C\e;
    if n > 1,q = (q + oldq)/2; end %damps oscillations
    oldq = q;
    delta_t = [q(4) q(5) q(6)]';
    delta_theta = [1 -q(3) q(2); q(3) 1 -q(1); -q(2) q(1) 1];
    [U,Lambda,V] = svd(delta_theta);
    delta_R = U*V';
    R = delta_R*R; % update rotation
    t = delta_R*t+delta_t; % update translation
    Xnew = R*X+repmat(t,1,N); % update moving points
    config_change = sqrt(sum(sum((Xnew-Xold).^2))/... 
    sum(sum((Xold-repmat(mean(Xold,2),1,N)).^2)));
    Xold = Xnew;
end
for ii = 1:N
    D = W(:,:,ii)*(Xnew(:,ii)-Y(:,ii));
    FRE(ii) = D'*D;
end
FRE = sqrt(mean(FRE));

function C = C_maker(X,W)
W1 = W(:,1,:); W2 = W(:,2,:); W3 = W(:,3,:);
X1 = X(1,:); X2 = X(2,:); X3 = X(3,:);
X = reshape([repmat(X1,3,1);repmat(X2,3,1);repmat(X3,3,1)],size(W));
X1 = X(:,1,:); X2 = X(:,2,:); X3 = X(:,3,:);
C = [W2.*X3-W3.*X2, -W1.*X3+W3.*X1, -W1.*X2+W2.*X1, W1,  W2, W3];
C = permute(C,[1,3,2]);
C = reshape(C,[],6);

function e = e_maker(X,Y,W)
W1 = W(:,1,:); W2 = W(:,2,:); W3 = W(:,3,:);
D = Y-X;
D1 = D(1,:); D2 = D(2,:); D3 = D(3,:);
D = reshape([repmat(D1,3,1);repmat(D2,3,1);repmat(D3,3,1)],size(W));
D1 = D(:,1,:); D2 = D(:,2,:); D3 = D(:,3,:);
e = [W1.*D1 + W2.*D2 + W3.*D3];
e = e(:);

function [R, t, FRE] = point_register(X, Y)
% This function performs point-based rigid body registration with equal
% weighting for all fiducials. It returns a rotation matrix, translation
% vector and the FRE.
if nargin < 2
    error('At least two input arguments are required.');
end
[Ncoords Npoints] = size(X);
[Ncoords_Y Npoints_Y] = size(Y);
if Ncoords ~= 3 | Ncoords_Y ~= 3
    error('Each argument must have exactly three rows.');
elseif (Ncoords ~= Ncoords_Y) | (Npoints ~= Npoints_Y)
    error('X and Y must have the same number of columns.');
elseif Npoints < 3
    error('X and Y must each have 3 or more columns.');
end
Xbar = mean(X,2);  % X centroid
Ybar = mean(Y,2);  % Y centroid
Xtilde = X-repmat(Xbar,1,Npoints); % X relative to centroid
Ytilde = Y-repmat(Ybar,1,Npoints); % Y relative to centroid
H = Xtilde*Ytilde';  % cross covariance matrix
[U S V] = svd(H);    % U*S*V' = H
R = V*diag([1, 1, det(V*U)])*U';
t = Ybar - R*Xbar;
FREvect = R*X + repmat(t,1,Npoints) - Y;
FRE = sqrt(mean(sum(FREvect.^2,1)));

REFERENCES