Linear Least-Squares Optimization for Point-to-Plane ICP Surface Registration

Article · January 2004

1 author:

Kok-Lim Low
National University of Singapore
57 PUBLICATIONS **1,190** CITATIONS

SEE PROFILE

Technical Report TR04-004, Department of Computer Science, University of North Carolina at Chapel Hill, February 2004.

Linear Least-Squares Optimization for Point-to-Plane ICP Surface Registration

Kok-Lim Low

Department of Computer Science
University of North Carolina at Chapel Hill
Email: lowk@cs.unc.edu

ABSTRACT

The Iterative Closest Point (ICP) algorithm that uses the point-to-plane error metric has been shown to converge much faster than one that uses the point-to-point error metric. At each iteration of the ICP algorithm, the change of relative pose that gives the minimal point-to-plane error is usually solved using standard nonlinear least-squares methods, which are often very slow. Fortunately, when the relative orientation between the two input surfaces is small, we can approximate the nonlinear optimization problem with a linear least-squares one that can be solved more efficiently. We detail the derivation of a linear system whose least-squares solution is a good approximation to that obtained from a nonlinear optimization.

1 INTRODUCTION

3D shape alignment is an important part of many applications. It is used for object recognition in which newly acquired shapes in the environment are fitted to model shapes in the database. For reverse engineering and building real-world models for virtual reality, it is used to align multiple partial range scans to form models that are more complete. For autonomous range acquisition, 3D registration is used to accurately localize the range scanner, and to align data from multiple scans for view-planning computation.

Since its introduction by Besl and McKay [Besl92], the ICP (Iterative Closest Point) algorithm has become the most widely used method for aligning three-dimensional shapes (a similar algorithm was also introduced by Chen and Medioni [Chen92]). Rusinkiewicz and Levoy [Rusinkiewicz01] provide a recent survey of the many ICP variants based on the original ICP concept.

In the ICP algorithm described by Besl and McKay [Besl92], each point in one data set is paired with the closest point in the other data set to form correspondence pairs. Then a point-to-point error metric is used in which the sum of the squared distance between points in each correspondence pair is minimized. The process is iterated until the error becomes smaller than a threshold or it stops changing. On the other hand, Chen and Medioni [Chen92] used a point-to-plane error metric in which the object of minimization is the sum of the squared distance between a point and the tangent plane at its correspondence point. Unlike the point-to-point metric, which has a closed-form solution, the point-to-plane metric is usually solved using standard nonlinear least squares methods, such as the Levenberg-Marquardt method [Press92]. Although each iteration of the point-to-plane ICP algorithm is generally slower than the point-to-point version, researchers have observed significantly better convergence rates in the former [Rusinkiewicz01]. A more theoretical explanation of the convergence of the point-to-plane metric is described by Pottmann et al [Pottmann02].

In [Rusinkiewicz01], it was suggested that when the relative orientation (rotation) between the two input surfaces is small, one can approximate the nonlinear least-squares optimization problem with a linear one, so as to speed up the computation. This approximation is simply done by replacing sin θ by θ and cos θ by 1 in the rotation matrix.

In this technical report, we describe in detail the derivation of a system of linear equations to approximate the original nonlinear system, and demonstrate how the least-squares solution to the linear system can be obtained using SVD (singular value decomposition). A 3D rigid-body transformation matrix is then constructed from the linear least-squares solution.

2 POINT-TO-PLANE ICP ALGORITHM

Given a source surface and a destination surface, each iteration of the ICP algorithm first establishes a set of pair-correspondences between points in the source surface and points in the destination surfaces. For example, for each point on the source surface, the nearest point on the destination surface is chosen as its correspondence [Besl92] (see [Rusinkiewicz01] for other approaches to find point correspondences). The output of an ICP iteration is a 3D rigid-body transformation **M** that transforms the source points such that the total error between the corresponding points, under a certain chosen error metric, is minimal.

When the point-to-plane error metric is used, the object of minimization is the sum of the squared distance between each source point and the tangent plane at its corresponding destination point (see Figure 1). More specifically, if $\mathbf{s}_i = (s_{ix}, s_{iy}, s_{iz}, 1)^{\mathrm{T}}$ is a source point, $\mathbf{d}_i = (d_{ix}, d_{iy}, d_{iz}, 1)^{\mathrm{T}}$ is the corresponding destination point, and $\mathbf{n}_i = (n_{ix}, n_{iy}, n_{iz}, 0)^{\mathrm{T}}$ is the unit normal vector at $\mathbf{d}_i$, then the goal of each ICP iteration is to find $\mathbf{M}_{\mathrm{opt}}$ such that

$$\mathbf{M}_{\mathrm{opt}} = \arg\min_{\mathbf{M}} \sum_i \left((\mathbf{M} \cdot \mathbf{s}_i - \mathbf{d}_i) \bullet \mathbf{n}_i \right)^2 \qquad (1)$$

where **M** and $\mathbf{M}_{\mathrm{opt}}$ are 4×4 3D rigid-body transformation matrices.

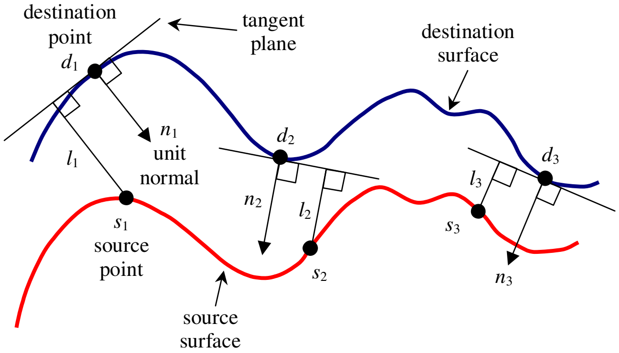

***Figure 1**: Point-to-plane error between two surfaces.*

A 3D rigid-body transformation **M** is composed of a rotation matrix **R**(α, β, γ) and a translation matrix **T**(t_x, t_y, t_z), i.e.

$$\mathbf{M} = \mathbf{T}(t_x, t_y, t_z) \cdot \mathbf{R}(\alpha, \beta, \gamma) \tag{2}$$

where

$$\mathbf{T}(t_x, t_y, t_z) = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3}$$

and

$$\begin{aligned} \mathbf{R}(\alpha, \beta, \gamma) &= \mathbf{R}_z(\gamma) \cdot \mathbf{R}_y(\beta) \cdot \mathbf{R}_x(\alpha) \\ &= \begin{pmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{aligned} \tag{4}$$

with

$$\begin{aligned} r_{11} &= \cos\gamma\cos\beta, \\ r_{12} &= -\sin\gamma\cos\alpha + \cos\gamma\sin\beta\sin\alpha, \\ r_{13} &= \sin\gamma\sin\alpha + \cos\gamma\sin\beta\cos\alpha, \\ r_{21} &= \sin\gamma\cos\beta, \\ r_{22} &= \cos\gamma\cos\alpha + \sin\gamma\sin\beta\sin\alpha, \\ r_{23} &= -\cos\gamma\sin\alpha + \sin\gamma\sin\beta\cos\alpha, \\ r_{31} &= -\sin\beta, \\ r_{32} &= \cos\beta\sin\alpha, \\ r_{33} &= \cos\beta\cos\alpha. \end{aligned}$$

$\mathbf{R}_x(\alpha)$, $\mathbf{R}_y(\beta)$ and $\mathbf{R}_z(\gamma)$ are rotations of α, β, and γ radians about the x-axis, y-axis and z-axis, respectively.

Equation (1) is essentially a least-squares optimization problem, and solving it requires the determination of only the values of the six parameters α, β, γ, t_x, t_y, and t_z. However, since α, β, and γ are arguments of nonlinear trigonometric functions in the rotation matrix **R**, efficient linear least-squares techniques cannot be applied to obtain the solution. In the next section, we present how this nonlinear least-squares problem can be approximated by a linear one, so that a linear least-squares technique can be applied.

3 LINEAR APPROXIMATION

When an angle $\theta \approx 0$, we can use the approximations $\sin\theta \approx \theta$ and $\cos\theta \approx 1$. Therefore, when $\alpha, \beta, \gamma \approx 0$,

$$\begin{aligned} \mathbf{R}(\alpha, \beta, \gamma) &\approx \begin{pmatrix} 1 & \alpha\beta - \gamma & \alpha\gamma + \beta & 0 \\ \gamma & \alpha\beta\gamma + 1 & \beta\gamma - \alpha & 0 \\ -\beta & \alpha & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ &\approx \begin{pmatrix} 1 & -\gamma & \beta & 0 \\ \gamma & 1 & -\alpha & 0 \\ -\beta & \alpha & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \hat{\mathbf{R}}(\alpha, \beta, \gamma). \end{aligned} \tag{5}$$

Then, **M** is approximated by

$$\begin{aligned} \hat{\mathbf{M}} &= \mathbf{T}(t_x, t_y, t_z) \cdot \hat{\mathbf{R}}(\alpha, \beta, \gamma) \\ &= \begin{pmatrix} 1 & -\gamma & \beta & t_x \\ \gamma & 1 & -\alpha & t_y \\ -\beta & \alpha & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}. \end{aligned} \tag{6}$$

We can now rewrite Equation (1) as

$$\hat{\mathbf{M}}_{\text{opt}} = \arg\min_{\hat{\mathbf{M}}} \sum_i \left(\left(\hat{\mathbf{M}} \cdot \mathbf{s}_i - \mathbf{d}_i \right) \bullet \mathbf{n}_i \right)^2. \tag{7}$$

Each $\left(\hat{\mathbf{M}} \cdot \mathbf{s}_i - \mathbf{d}_i \right) \bullet \mathbf{n}_i$ in (7) can be written as a linear expression of the six parameters α, β, γ, t_x, t_y, and t_z:

$$\begin{aligned} \left(\hat{\mathbf{M}} \cdot \mathbf{s}_i - \mathbf{d}_i \right) \bullet \mathbf{n}_i &= \left(\hat{\mathbf{M}} \cdot \begin{pmatrix} s_{ix} \\ s_{iy} \\ s_{iz} \\ 1 \end{pmatrix} - \begin{pmatrix} d_{ix} \\ d_{iy} \\ d_{iz} \\ 1 \end{pmatrix} \right) \bullet \begin{pmatrix} n_{ix} \\ n_{iy} \\ n_{iz} \\ 0 \end{pmatrix} \\ &= [(n_{iz}s_{iy} - n_{iy}s_{iz})\alpha + (n_{ix}s_{iz} - n_{iz}s_{ix})\beta + (n_{iy}s_{ix} - n_{ix}s_{iy})\gamma + \\ &\quad n_{ix}t_x + n_{iy}t_y + n_{iz}t_z] - \\ &\quad [n_{ix}d_{ix} + n_{iy}d_{iy} + n_{iz}d_{iz} - n_{ix}s_{ix} - n_{iy}s_{iy} - n_{iz}s_{iz}]. \end{aligned}$$

Given N pairs of point correspondences, we can arrange all $\left(\hat{\mathbf{M}} \cdot \mathbf{s}_i - \mathbf{d}_i \right) \bullet \mathbf{n}_i$, $1 \le i \le N$, into a matrix expression

$$\mathbf{Ax} - \mathbf{b}$$

where

$$\mathbf{b} = \begin{pmatrix} n_{1x}d_{1x} + n_{1y}d_{1y} + n_{1z}d_{1z} - n_{1x}s_{1x} - n_{1y}s_{1y} - n_{1z}s_{1z} \\ n_{2x}d_{2x} + n_{2y}d_{2y} + n_{2z}d_{2z} - n_{2x}s_{2x} - n_{2y}s_{2y} - n_{2z}s_{2z} \\ \vdots \\ n_{Nx}d_{Nx} + n_{Ny}d_{Ny} + n_{Nz}d_{Nz} - n_{Nx}s_{Nx} - n_{Ny}s_{Ny} - n_{Nz}s_{Nz} \end{pmatrix}. \tag{8}$$

$$\mathbf{x} = \begin{pmatrix} \alpha & \beta & \gamma & t_x & t_y & t_z \end{pmatrix}^{\mathrm{T}} \tag{9}$$

and

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & n_{1x} & n_{1y} & n_{1z} \\ a_{21} & a_{22} & a_{23} & n_{2x} & n_{2y} & n_{2z} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{N1} & a_{N2} & a_{N3} & n_{Nx} & n_{Ny} & n_{Nz} \end{pmatrix} \tag{10}$$

with

$$\begin{aligned} a_{i1} &= n_{iz}s_{iy} - n_{iy}s_{iz}, \\ a_{i2} &= n_{ix}s_{iz} - n_{iz}s_{ix}, \\ a_{i3} &= n_{iy}s_{ix} - n_{ix}s_{iy}. \end{aligned}$$

Note that

$$\min_{\hat{\mathbf{M}}} \sum_i \left(\left(\hat{\mathbf{M}} \cdot \mathbf{s}_i - \mathbf{d}_i \right) \bullet \mathbf{n}_i \right)^2 = \min_{\mathbf{x}} \left| \mathbf{Ax} - \mathbf{b} \right|^2. \tag{11}$$

Therefore, we can obtain $\hat{\mathbf{M}}_{\text{opt}}$ by first solving for

$$\mathbf{x}_{\text{opt}} = \arg\min_{\mathbf{x}} \left| \mathbf{Ax} - \mathbf{b} \right|^2, \tag{12}$$

which is a standard linear least-squares problem, and can be solved using SVD (singular value decomposition) [Press92]. Let $\mathbf{A} = \mathbf{U\Sigma V}^{\mathrm{T}}$ be the SVD of $\mathbf{A}$. The pseudo-inverse of $\mathbf{A}$ is defined as the matrix $\mathbf{A}^{+} = \mathbf{V\Sigma}^{+}\mathbf{U}^{\mathrm{T}}$, where $\mathbf{\Sigma}^{+}$ is the matrix formed by taking the inverse of the non-zero elements of $\mathbf{\Sigma}$ (and leaving the zero elements unchanged). Then, the solution to the least-squares problem (12) is

$$\mathbf{x}_{\mathrm{opt}} = \mathbf{A}^{+}\mathbf{b}. \tag{13}$$

Suppose the solution $\mathbf{x}_{\mathrm{opt}} = (\alpha_{\mathrm{opt}}, \beta_{\mathrm{opt}}, \gamma_{\mathrm{opt}}, t_{x\mathrm{opt}}, t_{y\mathrm{opt}}, t_{z\mathrm{opt}})$. Note that since $\hat{\mathbf{R}}(\alpha_{\mathrm{opt}}, \beta_{\mathrm{opt}}, \gamma_{\mathrm{opt}})$ may not be a valid rotation matrix, we should not use the result $\mathbf{T}(t_{x\mathrm{opt}}, t_{y\mathrm{opt}}, t_{z\mathrm{opt}}) \cdot \hat{\mathbf{R}}(\alpha_{\mathrm{opt}}, \beta_{\mathrm{opt}}, \gamma_{\mathrm{opt}})$. Instead, we should use $\mathbf{T}(t_{x\mathrm{opt}}, t_{y\mathrm{opt}}, t_{z\mathrm{opt}}) \cdot \mathbf{R}(\alpha_{\mathrm{opt}}, \beta_{\mathrm{opt}}, \gamma_{\mathrm{opt}})$, even though it is not equal to $\hat{\mathbf{M}}_{\mathrm{opt}}$ as defined in (7).

4 DISCUSSION

In practice, the linear approximation method can be used even when the relative orientation between the two input surfaces is quite large, sometimes as large as 30°, which we have observed. However, this is very dependent on the geometry and the amount of overlap between the two input surfaces. As the relative orientation decreases after each ICP iteration, the linear approximation becomes more accurate in the next.

To improve the numerical stability of the computation, it is important to use a unit of distance that is comparable in magnitude with the rotation angles. The simplest way is to rescale and move the two input surfaces so that they are bounded within a unit sphere or cube centered at the origin.

ACKNOWLEDGEMENTS

Thanks to Anselmo Lastra for proof-reading this article.

REFERENCES

[Besl92] Paul J. Besl, and Neil D. McKay. *A Method for Registration of 3-D Shapes*. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 14(2), pp. 239–256, 1992.

[Chen92] Yang Chen, and Gerard Medioni. *Object Modeling by Registration of Multiple Range Images*. International Journal of Image and Vision Computing, 10(3), pp. 145–155, 1992.

[Pottmann02] H. Pottmann, S. Leopoldseder, M. Hofer. *Registration without ICP*. Technical Report No. 91, Institute of Geometry, Vienna University of Technology, February 2002.

[Press92] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing, Second Edition*, Cambridge University Press, 1992.

[Rusinkiewicz01] Szymon Rusinkiewicz and Marc Levoy. *Efficient Variants of the ICP Algorithm*. Proceedings of the International Conference on 3-D Digital Imaging and Modeling (3DIM), pp. 145–152, 2001.