A Method for Registration of 3-D Shapes

Paul J. Besl, *Member, IEEE*, and Neil D. McKay

***Abstract*—This paper describes a general-purpose, representation-independent method for the accurate and computationally efficient registration of 3-D shapes including free-form curves and surfaces. The method handles the full six degrees of freedom and is based on the iterative closest point (ICP) algorithm, which requires only a procedure to find the closest point on a geometric entity to a given point. The ICP algorithm always converges monotonically to the nearest local minimum of a mean-square distance metric, and experience shows that the rate of convergence is rapid during the first few iterations. Therefore, given an adequate set of initial rotations and translations for a particular class of objects with a certain level of "shape complexity," one can globally minimize the mean-square distance metric over all six degrees of freedom by testing each initial registration. For example, a given "model" shape and a sensed "data" shape that represents a major portion of the model shape can be registered in minutes by testing one initial translation and a relatively small set of rotations to allow for the given level of model complexity. One important application of this method is to register sensed data from unfixtured rigid objects with an ideal geometric model prior to shape inspection. The described method is also useful for deciding fundamental issues such as the congruence (shape equivalence) of different geometric representations as well as for estimating the motion between point sets where the correspondences are not known. Experimental results show the capabilities of the registration algorithm on point sets, curves, and surfaces.**

***Index Terms*— Free-form curve matching, free-form surface matching, motion estimation, pose estimation, quaternions, 3-D registration.**

I. Introduction

GLOBAL AND local shape matching metrics for free-form curves and surfaces as well as point sets were described in [3] in an attempt to formalize and unify the description of a key problem in computer vision: Given 3-D data in a sensor coordinate system, which describes a data shape that may correspond to a model shape, and given a model shape in a model coordinate system in a different geometric shape representation, estimate the optimal rotation and translation that aligns, or registers, the model shape and the data shape minimizing the distance between the shapes and thereby allowing determination of the equivalence of the shapes via a mean-square distance metric. Of key interest to many applications is the following question: Does a segmented region from a range image match a subset of B-spline surfaces on a computer-aided-design (CAD) model? This paper provides a solution to this free-form surface matching problem as defined in [3] and [5] as a special case of a simple, general, unified approach, which generalizes to n dimensions and provides solutions to 1) the point-set matching problem without correspondence and 2) the free-form curve matching problem. The algorithm requires no extracted features, no curve or surface derivatives, and no preprocessing of 3-D data, except for the removal of statistical outliers.

The main application of the proposed method as described here is to register digitized data from unfixtured rigid objects with an idealized geometric model prior to shape inspection. When inspecting shapes using high-accuracy noncontact measurement devices [4] over a shallow depth of field, the uncertainty in different sensed points does not vary by much. Therefore, for purposes of simplicity and relevance to inspection applications based on thousands of digitized points, the case of unequal uncertainty among points is not considered. Similarly, the removal of statistical outliers is considered a preprocessing step, is probably best implemented as such, and will also not be addressed. In the context of inspection applications, the assumption that a high-accuracy noncontact measurement device does not generate bad data is reasonable since some sensors have the ability to reject highly uncertain measurements.

The proposed shape registration algorithm can be used with the following representations of geometric data:

1) Point sets
2) line segment sets (polylines)
3) implicit curves: $\vec{g}(x, y, z) = 0$
4) parametric curves: $(x(u), y(u), z(u))$
5) triangle sets (faceted surfaces)
6) implicit surfaces: $g(x, y, z) = 0$
7) parametric surfaces: $(x(u, v), y(u, v), z(u, v))$.

This covers most applications that would utilize a method to register 3-D shapes. Other representations are handled by providing a procedure for evaluating the closest point on the given shape to a given digitized point.

This paper is structured as follows: Several relevant papers from the literature are first reviewed. Next, the mathematical preliminaries of computing the closest point on a shape to a given point are covered for the geometric representations mentioned above. Then, the iterative closest point (ICP) algorithm is stated, and a theorem is proven concerning its monotonic convergence property. The issue of the initial registration states is addressed next. Finally, experimental results for point sets, curves, and surfaces are presented to demonstrate the capabilities of the ICP registration algorithm.

II. Literature Review

Relatively little work has been published in the area of registration (pose estimation, alignment, motion estimation) of 3-D

Manuscript received October 30, 1990; revised May 6, 1991.

The authors are with the Computer Science Department, General Motors Research Laboratories, Warren, MI 48090-9055.

IEEE Log Number 9102686.

free-form shapes. Most of the existing literature addressing global shape matching or registration have addressed limited classes of shapes, namely, 1) polyhedral models, 2) piecewise-(super)quadric models [2], [27], and 3) point sets with known correspondence. The reader may consult [6] and [14] for pre-1985 work in these areas. For a sampling of other more recent related work not discussed below, see [8], [10], [12], [13], [19], [20], [24], [26], [34], [35], [37], [39], [44], [46], [48], [53], [58], [59].

Historically, free-form shape matching using 3-D data was done earliest by Faugeras and his group at INRIA [18], where they demonstrated effective matching with a Renault auto part (steering knuckle) in the early 1980's. This work popularized the use of quaternions for least squares registration of corresponding 3-D point sets in the computer vision community. The alternative use of the singular value decomposition (SVD) algorithm [23], [1], [49] was not as widely known in this time frame. The primary limitation of this work was that it relied on the probable existence of reasonably large planar regions within a free-form shape.

Schwartz and Sharir [50] developed a solution to the free-form space curve matching problem without feature extraction in late 1985. They used a nonquaternion approach to computing the least squares rotation matrix. The method works well with reasonable quality curve data but has difficulty with very noisy curves because the method uses arclength sampling of the curves to obtain corresponding point sets.

Haralick *et al.* [28] addressed the 3-D point-set pose estimation problem using robust methods combined with the least squares SVD registration approach, which provided a robust statistical alternative to the least squares quaternion or SVD point set matching. This algorithm is able to handle statistical outliers and could theoretically be substituted for our quaternion-based algorithm as long as the determinant of the orthonormal matrix is strictly a positive one. A recent conference proceedings [47] contains new contributions on this subject.

Horn [31] derived an alternative formulation of Faugeras's method [18] of least squares quaternion matching that uses the maximum eigenvalue of a 4×4 matrix instead of the minimum eigenvalue. Horn [30] and Brou [11] also developed the extended Gaussian image (EGI) methods allowing the matching of convex and restricted sets of nonconvex shapes based on surface normal histograms.

Taubin [55] has done some interesting work in the area of implicit algebraic nonplanar 3-D curve and surface estimation with applications to position estimation without feature extraction. He describes a method of approximating data points with implicit algebraic forms up to the tenth degree using an approximate distance metric. Global shapes (not occluded shapes) can be identified based on generalized eigenvalues, and the registration transformation can be recovered. The method is shown to be useful for complete planar curve and space curve shapes, but it is unclear that the effectiveness generalizes well to more complicated surfaces, such as terrain data or a human face. Taubin has stated that the numerical methods of the approximate distance fit tend to break down above the tenth degree. He later [56] extended his work in shape description by investigating shape matching based on generalized shape polynomials. This demonstrated some interesting theoretical results but remains to be demonstrated for practical use on complex surfaces.

Szeliski [54] also describes a method for estimating motion from sparse range data without correspondence between the points and without feature extraction. His primary goal was to create a method for estimating the motion of the observer between two range image frames of the same terrain. Given the set of points from one frame, he applies a smoothness assumption to create a smoothing spline approximation of the points. Then, a conventional steepest descent algorithm is used to rotate and translate the second data set so that it minimizes the sum of the covariance-weighted z differences between the points and the surface. His approach is based on a regular xy-grid structure, and true 3-D point-to-surface distances are not computed. The steepest-descent approach is a slower alternative to reaching the local minima than our proposed ICP algorithm described below. Szeliski uses optimal Bayesian mathematics to allow him to downweight noisier values at longer ranges from a simulated range finder. For navigation range imaging sensors, the uncertainty in data points vary significantly from the foreground to the background. For high-accuracy sensors with shallow depths of field, the uncertainty variation between points is orders of magnitude less and is of much less concern. Szeliski provides experimental results for synthetic terrain data and a block. The terrain data motion test was a simple translation along one axis: a 1-D correlation problem. His block test did involve six degrees of freedom, but the block is a very simple shape. Overall, this work presents some interesting ideas, but the experimental results are unconvincing for applications.

Horn and Harris [33] also addressed the problem of estimating the exact rigid-body motion of the observer given sequentially digitized range image frames of the same terrain. They describe a range rate constraint equation and an elevation rate constraint equation. The result is a noniterative least squares method that provides a six-degree-of-freedom motion estimate as long as the motion between frames of data is relatively small. This method is much quicker than the one proposed by Szeliski, but it is not clear that this method generalizes to arbitrary rotations and translations of a shape.

Kamgar-Parsi *et al.* [36] also describe a method for the registration of multiple overlapping range images without distinctive feature extraction. This method works very well using the level sets of 2.5-D range data but is essentially restricted to the three degrees of freedom in the plane since the work was addressed toward piecing together terrain map data.

Li [38] addressed free-form surface matching with arbitrary rotations and translations. His method forms an attributed relational graph of fundamental surface regions for data and model shapes and then performs graph matching using an inexact approach that allows for variability in attributes as well as in graph adjacency relationships. This seems to be a reasonable approach but relies on extraction of derivative-based quantities. Experimental results are shown for a coffee cup and the Renault auto part; see also Wong *et al.* [60] for other related work using attributed graphs for 3-D matching.

The work of Gilbert and Foo [21] and Gilbert *et al.* [22] is related in that it addresses the computation of distance between two object shapes. Such methods could be the basis for similar shape matching techniques as are described below. The major inconvenience with their method, though, is that object shapes must be decomposed into convex subbodies, which is a problem that, in general, is not trivial for arbitrary CAD models or for digitized 3-D data.

III. Mathematical Preliminaries

In this section, methods for computing the closest point to a given point on the various geometric representations listed above are described. First, the basic geometric entities are covered, followed by parametric entities, and, finally, implicit entities. The reader might consult Mortenson [42] on some of the items below for additional information.

The Euclidean distance $d(\vec{r}_1, \vec{r}_2)$ between the two points $\vec{r}_1 = (x_1, y_1, z_1)$ and $\vec{r}_2 = (x_2, y_2, z_2)$ is $d(\vec{r}_1, \vec{r}_2) = \|\vec{r}_1 - \vec{r}_2\| = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$. Let A be a point set with N_a points denoted $\vec{a}_i$: $A = \{\vec{a}_i\}$ for $i = 1, \ldots, N_a$. The distance between the point $\vec{p}$ and the point set A is

$$d(\vec{p}, A) = \min_{i \in \{1, \ldots, N_a\}} d(\vec{p}, \vec{a}_i). \tag{1}$$

The closest point $\vec{a}_j$ of A satisfies the equality $d(\vec{p}, \vec{a}_j) = d(\vec{p}, A)$.

Let l be the line segment connecting the two points $\vec{r}_1$ and $\vec{r}_2$. The distance between the point $\vec{r}$ and the line segment l is

$$d(\vec{p}, l) = \min_{u+v=1} \|u\vec{r}_1 + v\vec{r}_2 - \vec{p}\| \tag{2}$$

where $u \in [0, 1]$ and $v \in [0, 1]$. The required closed-form computations are straightforward. Let L be the set of N_l line segments denoted l_i, and let $L = \{l_i\}$ for $i = 1, \ldots, N_l$. The distance between the point $\vec{p}$ and the line segment set L is

$$d(\vec{p}, L) = \min_{i \in \{1, \ldots, N_l\}} d(\vec{p}, l_i). \tag{3}$$

The closest point $\vec{y}_j$ on the line segment set L satisfies the equality $d(\vec{p}, \vec{y}_j) = d(\vec{p}, L)$.

Let t be the triangle defined by the three points $\vec{r}_1 = (x_1, y_1, z_1)$, $\vec{r}_2 = (x_2, y_2, z_2)$, and $\vec{r}_3 = (x_3, y_3, z_3)$. The distance between the point $\vec{p}$ and the triangle t is

$$d(\vec{p}, t) = \min_{u+v+w=1} \|u\vec{r}_1 + v\vec{r}_2 + w\vec{r}_3 - \vec{p}\| \tag{4}$$

where $u \in [0, 1]$, $v \in [0, 1]$, and $w \in [0, 1]$. The required closed-form computations are again straightforward. Let T be the set of N_t triangles denoted t_i, and let $T = \{t_i\}$ for $i = 1, \ldots, N_t$. The distance between the point $\vec{p}$ and the triangle set T is given by

$$d(\vec{p}, T) = \min_{i \in \{1, \ldots, N_t\}} d(\vec{p}, t_i). \tag{5}$$

The closest point $\vec{y}_j$ on the triangle set T satisfies the equality $d(\vec{p}, \vec{y}_j) = d(\vec{p}, T)$.

A. Point to Parametric Entity Distance

In this section, a parametric curve and a parametric surface are treated as a single parametric entity $\vec{r}(\vec{u})$, where $\vec{u} = u \in \mathcal{R}^1$ should be substituted for parametric curves, and $\vec{u} = (u, v) \in \mathcal{R}^2$ should be substituted for parametric surfaces ($\mathcal{R}$ denotes the real line). The evaluation domain for a curve is an interval, but the evaluation domain for a surface can be any arbitrary closed-connected region in the plane. For more information on parametric entities, such as Bezier and B-spline curves and surfaces, see [9], [15]–[17], [42], [52].

The distance from a given point $\vec{p}$ to a parametric entity E is

$$d(\vec{p}, E) = \min_{\vec{r}(\vec{u}) \in E} d(\vec{p}, \vec{r}(\vec{u})). \tag{6}$$

The computations to compute the distance are not closed form and are relatively involved. One method for computing point-to-curve and point-to-surface distances is described below. Sets of parametric entities are again straightforward once the distance metric for an individual entity is implemented. Let F be the set of N_e parametric entities denoted E_i, and let $F = \{E_i\}$ for $i = 1, N_e$. The distance between a point $\vec{p}$ and the parametric entity set F is

$$d(\vec{p}, F) = \min_{i \in \{1, \ldots, N_e\}} d(\vec{p}, \vec{E}_i). \tag{7}$$

The closest point $\vec{y}_j$ on the parametric entity set F satisfies the equality $d(\vec{p}, \vec{y}_j) = d(\vec{p}, F)$.

Our first step towards computing the distance from a point to a parametric entity is creating a simplex-based approximation (line segments or triangles). For a parametric space curve $C = \{\vec{r}(u)\}$, one can compute a polyline $L(C, \delta)$ such that the piecewise-linear approximation never deviates from the space curve by more than a prespecified distance δ. By tagging each point of the polyline with its corresponding u argument values of the parametric curve, one can obtain an estimate of the u_a argument value of the closest point from the line segment set.

Similarly, for a parametric surface $S = \{\vec{r}(u, v)\}$, one can compute a triangle set $T(S, \delta)$ such that the piecewise-triangular approximation never deviates from the surface by more than a prespecified distance δ. By tagging each triangle vertex with the corresponding (u, v) argument values of the parametric surface, one can obtain an estimate (u_a, v_a) of the argument values of the closest point from the triangle set. As a result of these curve and surface procedures, one can assume that an initial value $\vec{u}_a$ is available such that $\vec{r}(\vec{u}_a)$ is very close to the closest point on the parametric entity.

The point-to-parametric-entity distance problem is ideal for employing a pure Newton's minimization approach when a reliable starting point $\vec{u}_a$ is available. The scalar objective function to be minimized is

$$f(\vec{u}) = \|\vec{r}(\vec{u}) - \vec{p}\|^2. \tag{8}$$

Let $\nabla = [\partial/\partial\vec{u}]^t$ be the vector differential gradient operator (where t implies vector transpose). The minimum of f occurs when $\nabla f = 0$. When the parametric entity is a surface, the 2-D gradient vector is $\nabla f = [f_u, f_v]^t$, and the 2-D Hessian matrix

is

$$\nabla\nabla^t(f) = \begin{bmatrix} f_{uu} & f_{uv} \\ f_{uv} & f_{vv} \end{bmatrix} \tag{9}$$

where the partial derivatives of the objective function are given by

$$f_u(\vec{u}) = 2\vec{r}_u^t(\vec{u})(\vec{r}(\vec{u}) - \vec{p}) \tag{10}$$
$$f_v(\vec{u}) = 2\vec{r}_v^t(\vec{u})(\vec{r}(\vec{u}) - \vec{p}) \tag{11}$$
$$f_{uu}(\vec{u}) = 2\vec{r}_{uu}^t(\vec{u})(\vec{r}(\vec{u}) - \vec{p}) + 2\vec{r}_u^t(\vec{u})\vec{r}_u(\vec{u}) \tag{12}$$
$$f_{vv}(\vec{u}) = 2\vec{r}_{vv}^t(\vec{u})(\vec{r}(\vec{u}) - \vec{p}) + 2\vec{r}_v^t(\vec{u})\vec{r}_v(\vec{u}) \tag{13}$$
$$f_{uv}(\vec{u}) = 2\vec{r}_{uv}^t(\vec{u})(\vec{r}(\vec{u}) - \vec{p}) + 2\vec{r}_u^t(\vec{u})\vec{r}_v(\vec{u}). \tag{14}$$

The curve case requires only computation of f_u and f_{uu}. The Newton's update formula for either entity is

$$\vec{u}_{k+1} = \vec{u}_k - [\nabla\nabla^t(f)(\vec{u}_k)]^{-1}\nabla f(\vec{u}_k) \tag{15}$$

where $\vec{u}_0 = \vec{u}_a$. When using the starting point selection method described above based on a simplex approximation with a reasonably small δ, Newton's method for computing the closest point generally converges in one to five iterations and typically in three. The computational cost of Newton's method is very low in contrast with finding good starting points.

B. Point to Implicit Entity Distance

An implicit geometric entity is defined as the zero set of a possibly vector-valued multivariate function $\vec{g}(\vec{r}) = 0$. The distance from a given point $\vec{p}$ to an implicit entity I is

$$d(\vec{p}, I) = \min_{\vec{g}(\vec{r})=0} d(\vec{p}, \vec{r}) = \min_{\vec{g}(\vec{r})=0} \|\vec{r} - \vec{p}\|. \tag{16}$$

The calculations to compute this distance are also not closed form and are relatively involved. One method for computing point-to-curve and point-to-surface distances is outlined below. Sets of implicit entities are straightforward once the distance metric for an individual entity is implemented. Let J be the set of N_I parametric entities denoted I_k and $J = \{I_k\}$ for $k = 1, N_I$. The distance between a point $\vec{p}$ and the implicit entity set J is

$$d(\vec{p}, J) = \min_{k\in\{1,\ldots,N_I\}} d(\vec{p}, \vec{I}_k). \tag{17}$$

The closest point $\vec{y}_j$ on the implicit entity I_j satisfies the equality $d(\vec{p}, \vec{y}_j) = d(\vec{p}, J)$.

Our first step towards computing the distance from a point to an implicit entity is creating a simplex-based approximation (line segments or triangles) as was done for parametric entities [7]. Computing the point-to-line set or point-to-triangle set distance yields an approximate closest point $\vec{r}_a$, which can be used to compute the exact distance.

The implicit entity distance problem is quite different from the parametric entity case where unconstrained optimization suffices. To find the closest point on an implicit entity defined by $\vec{g}(\vec{r}) = 0$ to a given point $\vec{p}$, one must solve a constrained optimization problem to minimize a quadratic objective function subject to a nonlinear constraint

$$\text{Minimize } f(\vec{r}) = \|\vec{r} - \vec{p}\|^2 \text{ where } \vec{g}(\vec{r}) = 0. \tag{18}$$

One approach to this problem is to form the augmented Lagrange multiplier system of equations [40]:

$$\begin{aligned} \nabla f(\vec{r}) + \vec{\lambda}^t\nabla\vec{g}(\vec{r}) &= 0 \\ \vec{g}(\vec{r}) &= 0 \end{aligned} \tag{19}$$

where $\nabla = [\partial/\partial\vec{r}]^t$ and solve this system of nonlinear equations via numerical methods. The number of equations and unknowns for the nonlinear system is three for planar curves, four for surfaces, and five for implicitly defined space curves. Continuation methods [41] can be used to solve this problem for algebraic entities even without a good starting point, but a good starting point will allow the use of faster methods, such as the multidimensional Newton's root finding method. From a numerical point of view, the parametric methods are much easier to deal with. From an applied point of view, no industrial CAD systems store free-form curves or surfaces in implicit form. For this reason, implicit surfaces of interest are dealt with in our implemented system either via special case mathematics (e.g., spheres) or via a parametric form. Of course, if there were an application where it was necessary to handle free-form implicit entities in their implicit form [51], the above algorithm could be implemented.

Taubin [55] uses an approximate distance algorithm that implies a simple update formula for surfaces and planar curves when $g(\vec{r}_0)$ is nearly zero:

$$\vec{r}_{k+1} = \vec{r}_k - \frac{\nabla g(\vec{r}_k) g(\vec{r}_k)}{\|\nabla g(\vec{r}_k)\|^2}. \tag{20}$$

This method is only exact if the infinite line with the direction $\nabla g(\vec{r})$ at the starting point $\vec{r}_0$ intersects the implicit entity at a point where the normal vector has that same direction. This is not true in general, and the approximation is generally worse the further the point is from the implicit entity. Therefore, this result cannot be used if precise distance results are required.

C. Corresponding Point Set Registration

All closest point (minimum distance) algorithms have been mentioned in forms that generalize to n dimensions. One more necessary procedure for yielding the least squares rotation and translation is reviewed. For our purposes, the quaternion-based algorithm is preferred over the singular value decomposition (SVD) method in two and three dimensions since reflections are not desired. The SVD approach, based on the cross-covariance matrix of two point distributions, does, however, generalize easily to n dimensions and would be our method of choice for $n > 3$ in any n-dimensional applications. The basic solution of Horn [31] is described below, although the method of Faugeras [18] is equivalent. Our summary stresses the role of the SVD cross-covariance matrix, which is an important relationship not discussed in other work.

The unit quaternion is a four vector $\vec{q}_R = [q_0 q_1 q_2 q_3]^t$, where $q_0 \geq 0$, and $q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$. The 3×3 rotation matrix generated by a unit rotation quaternion is found at the bottom of this page. Let $\vec{q}_T = [q_4 q_5 q_6]^t$ be a translation vector. The complete registration state vector $\vec{q}$ is denoted $\vec{q} = [\vec{q}_R | \vec{q}_T]^t$. Let $P = \{\vec{p}_i\}$ be a measured data point set to be aligned with a model point set $X = \{\vec{x}_i\}$, where $N_x = N_p$ and where each point $\vec{p}_i$ corresponds to the point $\vec{x}_i$ with the same index. The mean square objective function to be minimized is

$$f(\vec{q}) = \frac{1}{N_p} \sum_{i=1}^{N_p} \|\vec{x}_i - \boldsymbol{R}(\vec{q}_R)\vec{p}_i - \vec{q}_T\|^2. \tag{22}$$

The "center of mass" $\vec{\mu}_p$ of the measured point set P and the center of mass $\vec{\mu}_x$ for the X point set are given by

$$\vec{\mu}_p = \frac{1}{N_p} \sum_{i=1}^{N_p} \vec{p}_i \quad \text{and} \quad \vec{\mu}_x = \frac{1}{N_x} \sum_{i=1}^{N_x} \vec{x}_i. \tag{23}$$

The cross-covariance matrix Σ_{px} of the sets P and X is given by

$$\Sigma_{px} = \frac{1}{N_p} \sum_{i=1}^{N_p} [(\vec{p}_i - \vec{\mu}_p)(\vec{x}_i - \vec{\mu}_x)^t] = \frac{1}{N_p} \sum_{i=1}^{N_p} [\vec{p}_i \vec{x}_i^t] - \vec{\mu}_p \vec{\mu}_x^t. \tag{24}$$

The cyclic components of the anti-symmetric matrix $A_{ij} = (\Sigma_{px} - \Sigma_{px}^T)_{ij}$ are used to form the column vector $\Delta = [A_{23} \quad A_{31} \quad A_{12}]^T$. This vector is then used to form the symmetric 4×4 matrix $Q(\Sigma_{px})$

$$Q(\Sigma_{px}) = \begin{bmatrix} \operatorname{tr}(\Sigma_{px}) & \Delta^T \\ \Delta & \Sigma_{px} + \Sigma_{px}^T - \operatorname{tr}(\Sigma_{px}) \boldsymbol{I}_3 \end{bmatrix} \tag{25}$$

where $\boldsymbol{I}_3$ is the 3×3 identity matrix. The unit eigenvector $\vec{q}_R = [q_0 \quad q_1 \quad q_2 \quad q_3]^t$ corresponding to the maximum eigenvalue of the matrix $Q(\Sigma_{px})$ is selected as the optimal rotation. The optimal translation vector is given by

$$\vec{q}_T = \vec{\mu}_x - \boldsymbol{R}(\vec{q}_R)\vec{\mu}_p. \tag{26}$$

This least squares quaternion operation is $O(N_p)$ and is denoted as

$$(\vec{q}, d_{ms}) = \mathcal{Q}(P, X) \tag{27}$$

where d_{ms} is the mean square point matching error. The notation $\vec{q}(P)$ is used to denote the point set P after transformation by the registration vector $\vec{q}$.

IV. The Iterative Closest Point Algorithm

Now that the methods for computing the closest point on a geometric shape to a given point and for computing a least squares registration vector have been outlined, the ICP algorithm can be described in terms of an abstract geometric shape X whose internal representation must be known to execute the algorithm but is not of concern for this discussion. Thus, all that follows applies equally well to 1) sets of points, 2) sets of line segments, 3) sets of parametric curves, 4) sets of implicit curves, 5) sets of triangles, 6) sets of parametric surfaces, and 7) sets of implicit surfaces.

In the description of the algorithm, a "data" shape P is moved (registered, positioned) to be in best alignment with a "model" shape X. The data and the model shape may be represented in any of the allowable forms. For our purposes, the data shape must be decomposed into a point set if it is not already in point set form. Fortunately, this is easy; the points to be used for triangle and line sets are the vertices and the endpoints, and if the data shape comes in a surface or curve form, then the vertices and endpoints of the triangle/line approximation (as described above) are used. The number of points in the data shape will be denoted N_p. Let N_x be the number of points, line segments, or triangles involved in the model shape. As described above, the curve and surface closest-point evaluators implemented in our system require a framework of lines or triangles to yield the initial parameter values for the Newton's iteration; therefore, the number N_x is still relevant for these smooth entities but varies according to the accuracy of the approximation.

The distance metric d between an individual data point $\vec{p}$ and a model shape X will be denoted

$$d(\vec{p}, X) = \min_{\vec{x} \in X} \|\vec{x} - \vec{p}\|. \tag{28}$$

The closest point in X that yields the minimum distance is denoted $\vec{y}$ such that $d(\vec{p}, \vec{y}) = d(\vec{p}, X)$, where $\vec{y} \in X$. Note that computing the closest point is $O(N_x)$ worst case with expected cost $\log(N_x)$. When the closest point computation (from $\vec{p}$ to X) is performed for each point in P, that process is worst case $O(N_p N_x)$. Let Y denote the resulting set of closest points, and let $\mathcal{C}$ be the closest point operator:

$$Y = \mathcal{C}(P, X). \tag{29}$$

Given the resultant corresponding point set Y, the least squares registration is computed as described above:

$$(\vec{q}, d) = \mathcal{Q}(P, Y). \tag{30}$$

The positions of the data shape point set are then updated via $P = \vec{q}(P)$.

$$\boldsymbol{R} = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 - q_0 q_3) & 2(q_1 q_3 + q_0 q_2) \\ 2(q_1 q_2 + q_0 q_3) & q_0^2 + q_2^2 - q_1^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_2 q_3 + q_0 q_1) & q_0^2 + q_3^2 - q_1^2 - q_2^2 \end{bmatrix} \tag{21}$$

A. ICP Algorithm Statement

The ICP algorithm can now be stated:

- The point set P with N_p points $\{\vec{p}_i\}$ from the data shape and the model shape X (with N_x supporting geometric primitives: points, lines, or triangles) are given.
- The iteration is initialized by setting $P_0 = P$, $\vec{q}_0 = [1, 0, 0, 0, 0, 0, 0]^t$ and $k = 0$. The registration vectors are defined relative to the initial data set P_0 so that the final registration represents the complete transformation. Steps 1, 2, 3, and 4 are applied until convergence within a tolerance τ. The computational cost of each operation is given in brackets.
    a. Compute the closest points: $Y_k = \mathcal{C}(P_k, X)$ (cost: $0(N_p N_x)$ worst case, $0(N_p \log N_x)$ average).
    b. Compute the registration: $(\vec{q}_k, d_k) = \mathcal{Q}(P_0, Y_k)$ (cost: $O(N_p)$).
    c. Apply the registration: $P_{k+1} = \vec{q}_k(P_0)$ (cost: $O(N_p)$).
    d. Terminate the iteration when the change in mean-square error falls below a preset threshold $\tau > 0$ specifying the desired precision of the registration: $d_k - d_{k+1} < \tau$.

If a dimensionless threshold is desired, one can replace τ with $\tau\sqrt{tr(\Sigma_x)}$, where the square root of the trace of the covariance of the model shape indicates the rough size of the model shape.

B. Convergence Theorem

A convergence theorem for the ICP algorithm is now stated and proved. The key ideas are that 1) least squares registration generically reduces the average distance between corresponding points during each iteration, whereas 2) the closest point determination generically reduces the distance for each point individually. Of course, this individual distance reduction also reduces the average distance because the average of a set of smaller positive numbers is smaller. We offer a more elaborate explanation in the proof below.

Theorem: The iterative closest point algorithm always converges monotonically to a local minimum with respect to the mean-square distance objective function.

Proof: Given $P_k = \{\vec{p}_{ik}\} = \vec{q}_k(P_0)$ and X, compute the set of closest points $Y_k = \{\vec{y}_{ik}\}$ as prescribed above given the internal geometric representation of X. The mean squared error e_k of that correspondence is given by

$$e_k = \frac{1}{N_p}\sum_{i=1}^{N_p} \|\vec{y}_{ik} - \vec{p}_{ik}\|^2. \tag{31}$$

The $\mathcal{Q}$ operator is applied to get $\vec{q}_k$ and d_k from that correspondence:

$$d_k = \frac{1}{N_p}\sum_{i=1}^{N_p} \|\vec{y}_{ik} - \boldsymbol{R}(\vec{q}_{kR})\vec{p}_{i0} - \vec{q}_{kT}\|^2. \tag{32}$$

It is always the case that $d_k \leq e_k$. Suppose that $d_k > e_k$. If this were so, then the identity transformation on the point set would yield a smaller mean square error than the least squares registration, which cannot possibly be the case. Next, let the least squares registration $\vec{q}_k$ be applied to the point set P_0, yielding the point set P_{k+1}. If the previous correspondence to the set of points Y_k were maintained, then the mean square error is still d_k, that is

$$d_k = \frac{1}{N_p}\sum_{i=1}^{N_p} \|\vec{y}_{ik} - \vec{p}_{i,k+1}\|^2. \tag{33}$$

However, during the application of the subsequent closest point operator, a new point set Y_{k+1} is obtained: $Y_{k+1} = \mathcal{C}(P_{k+1}, X)$. It is clear that

$$\|\vec{y}_{i,k+1} - \vec{p}_{i,k+1}\| \leq \|\vec{y}_{ik} - \vec{p}_{i,k+1}\| \text{ for each } i = 1, N_p \tag{34}$$

because the point $\vec{y}_{ik}$ was the closest point prior to transformation by $\vec{q}_k$ and resides at some new distance relative to $\vec{p}_{i,k+1}$. If $\vec{y}_{i,k+1}$ were further from $\vec{p}_{i,k+1}$ than $\vec{y}_{ik}$, then this would directly contradict the basic operation of the $\mathcal{C}$ closest point operator. Therefore, the mean square errors e_k and d_k must obey the following inequality:

$$0 \leq d_{k+1} \leq e_{k+1} \leq d_k \leq e_k \text{ for all } k. \tag{35}$$

The lower bound occurs, of course, since mean-square errors cannot be negative. Because the mean-square error sequence is nonincreasing and bounded below, the algorithm as stated above must converge monotonically to a minimum value. Q.E.D.

Experimentally, we find fast convergence during the first few iterations that slows down as it approaches the local minimum. Even at this slow pace, somewhere between 30 and 50 iterations yields excellent results: $d_k \approx 0.1\%$ of model shape size. The convergence can be accelerated using a simple additional operation described in the next section.

C. An Accelerated ICP Algorithm

The accelerated ICP algorithm uses a minor variation on the basic line search methods of multivariate unconstrained minimization [45]. As the iterative closest point algorithm proceeds, a sequence of registration vectors is generated: $\vec{q}_1$, $\vec{q}_2$, $\vec{q}_3$, $\vec{q}_4$, $\vec{q}_5$, $\vec{q}_6$, ..., which traces out a path in the registration state space from the identity transformation toward a locally optimal shape match. Consider the difference vector sequence defined by

$$\Delta\vec{q}_k = \vec{q}_k - \vec{q}_{k-1} \tag{36}$$

which defines a direction in the registration state space. Let the angle in 7 space between the two last directions be denoted

$$\theta_k = \cos^{-1}\left(\frac{\Delta\vec{q}_k{}^t \Delta\vec{q}_{k-1}}{\|\Delta\vec{q}_k\|\|\Delta\vec{q}_{k-1}\|}\right) \tag{37}$$

and let $\delta\theta$ be a sufficiently small angular tolerance (e.g., 10°). If

$$\theta_k < \delta\theta \text{ and } \theta_{k-1} < \delta\theta \tag{38}$$

then there is good direction alignment for the last three registration state vectors: $\vec{q}_k$, $\vec{q}_{k-1}$, and $\vec{q}_{k-2}$. Let d_k, d_{k-1},

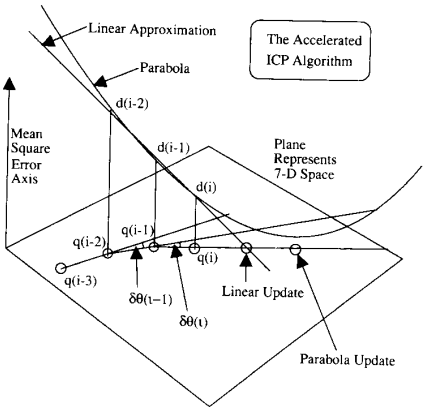


Fig. 1. Consistent direction allows acceleration of the ICP algorithm.

and d_{k-2} be the associated mean square errors, and let v_k, v_{k-1}, and v_{k-2} be associated approximate arc length argument values:

$$v_k = 0, v_{k-1} = -\|\Delta\vec{q}_k\|, v_{k-2} = -\|\Delta\vec{q}_{k-1}\| + v_{k-1}. \quad (39)$$

See Fig. 1 for a picture of the situation. Next, a linear approximation and a parabolic interpolant to the last three data points are computed:

$$d_1(v) = c_1 v + b_1 \; d_2(v) = a_2 v^2 + b_2 v + c_2 \quad (40)$$

which gives us a possible linear update, based on the zero crossing of the line, and a possible parabola update, based on the extremum point of the parabola:

$$v_1 = -b_1/a_1 > 0 \; v_2 = -b_2/2a_2. \quad (41)$$

To be on the safe side, we adopt a maximum allowable value $v_{\max}$. The following logic is used to perform an attempted update:

1) If $0 < v_2 < v_1 < v_{\max}$ or $0 < v_2 < v_{\max} < v_1$, use the parabola-based updated registration vector: $\vec{q}'_k = \vec{q}_k + v_2\Delta\vec{q}_k/\|\Delta\vec{q}_k\|$ instead of the usual vector $\vec{q}_k$ when performing the update on the point set, i.e., $P_{k+1} = \vec{q}'_k(P_0)$.
2) If $0 < v_1 < v_2 < v_{\max}$ or $0 < v_1 < v_{\max} < v_2$ or $v_2 < 0$ and $0 < v_1 < v_{\max}$, use the line-based updated registration vector $\vec{q}'_k = \vec{q}_k + v_1\Delta\vec{q}_k/\|\Delta\vec{q}_k\|$ instead of the usual vector $\vec{q}_k$.
3) If both $v_1 > v_{\max}$ and $v_2 > v_{\max}$, use the maximum allowable update $\vec{q}'_k = \vec{q}_k + v_{\max}\Delta\vec{q}_k/\|\Delta\vec{q}_k\|$ instead of the usual vector $\vec{q}_k$.

We have found experimentally that setting $v_{\max} = 25\|\Delta\vec{q}_k\|$ adaptively has provided a good sanity check on the updates allowing the iterative closest point algorithm to move to the local minimum with a given degree of precision in many fewer steps. A nominal run of more than 50 basic ICP iterations for a given value of τ is typically accelerated to 15 or 20 iterations.

If the updated registration vector were somehow to overshoot the minimum enough to yield a worse mean square error, it would be advantageous to construct a new parabola using the new registration with the last two steps and move to the appropriate minimum. This has not been necessary in our experience. To be rigorous, one can simply ignore the suggested update if it causes a worse mean square error.

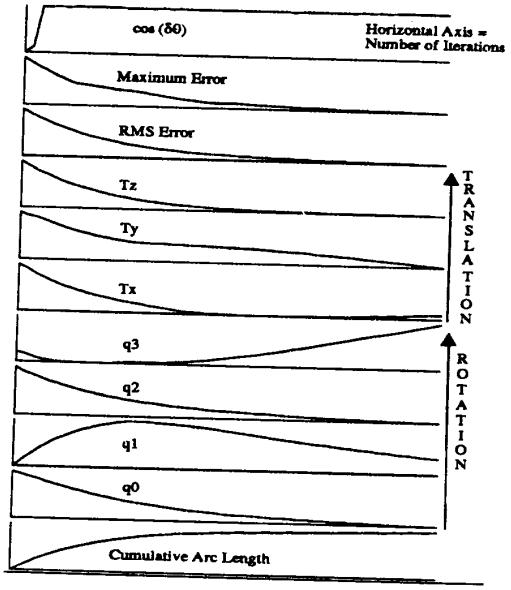


Fig. 2. Various quantities plotted against iteration count for the basic nonaccelerated ICP algorithm.

To give a quantitative example comparison, the registration values, RMS error, maximum error, angular change, and cumulative arc length values were recorded during 50 iterations of both the basic and accelerated ICP algorithms during the same free-form surface matching test. The results for the basic ICP algorithm are shown in Fig. 2. Note the smooth character of all the curves. The most important feature is that the $\cos(\delta\theta)$ plot indicates a consistent direction of updates for all but the first few iterations. In contrast, the accelerated ICP algorithm shows the desirable jumpy behavior as seen in Fig. 3. In addition, note how most quantities get close to their final values after the first acceleration step and very close after two. The acceleration steps occur whenever a V-shaped dip occurs in the plot of $\cos(\delta\theta)$ versus the iteration count.

D. Alternative Minimization Approaches

The ICP algorithm allows us to move from a given starting point to a local minima in 7 space relatively quickly in comparison with other possible alternatives. Each iteration requires only one evaluation of the closest point operator: the most expensive computation. Any optimization method that does not use explicit vector gradient estimates, such as Powell's direction set method, the Nelder-Mead downhill simplex method, or simulated annealling, requires literally *hundreds* to *tens of thousands* of closest point evaluations. These numbers are based on tests done to simulate the action of the least squares registration step involved in one ICP iteration but using instead Powell's direction set method and the Nelder-Mead method from [45].

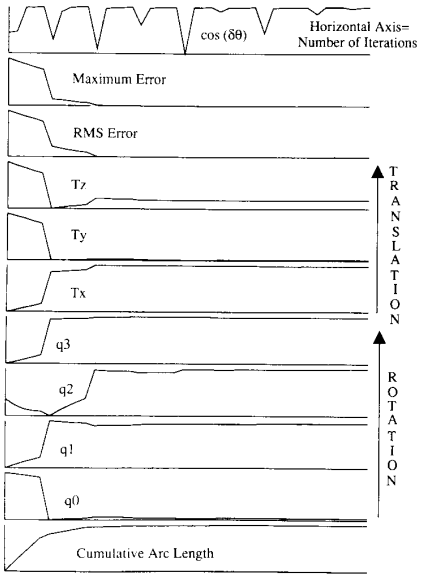


Fig. 3. Various quantities plotted against iteration count for the accelerated ICP algorithm.

Any optimization method that uses explicit vector gradients, such as steepest descent, conjugate gradient, and variable metric schemes, will require at least seven closest point evaluations for each numerical gradient evaluation. Therefore, such a method would have to converge in three or four iterations to be competitive with the accelerated ICP method. Such generic methods generally require many more than three iterations with the number of required closest point evaluations running well over 100 even in ideal circumstances. If a pure numerical Hessian-based Newton's method were used, the numerical gradient and Hessian computations would require at least 13 closest point evaluations per iteration, implying that the iteration would have to converge in two iterations to be competitive with the accelerated ICP algorithm. A pure Newton's method might require only three iterations if the initial point were already well into the region of attraction surrounding a local minimum, but the initial iterations would not be handled well by Newton's method.

Whenever an accelerated parabolic update takes place after three basic ICP steps, we can get nearly quadratic convergence for less than steepest descent cost. This is an interesting accomplishment for a function where derivatives cannot be evaluated. Note that the steepest descent gradient direction is not deliberately computed; we merely observe when a consistent direction is being followed.

Other problems involved with using general-purpose optimization methods are the following: 1) If any angles are used as in [54], angular cycles across 360° must be handled correctly, and 2) if a unit quaternion becomes a nonunit quaternion, as would be expected taking arbitrary direction steps in 4 space, the quaternion must be renormalized somewhere. Unfortunately, if the objective function evaluator changes the values in the state vector during the optimization iteration, this has a bad effect on most nonlinear optimization algorithms.

To summarize, any method that allows one to move from an initial state to its corresponding local minimum could theoretically be used in place of the ICP algorithm. For example, consider Szeliski's [54] work with steepest descent and three rotation angles. However, the arguments above indicate that one would have a hard time trying to find an algorithm that was only ten times slower on average. The key benefit of the ICP algorithm is that the convergence is fast and monotonic. No expensive closest point evaluations are spent on registration vectors that have worse mean square errors than the current state. Because of the ICP convergence theorem, one does not have to "feel around" in the multidimensional space to determine the direction in which to move.

V. The Set of Initial Registrations

Even though the ICP algorithm must converge monotonically to a local minimum from any given rotation and translation of the data point set, it may or may not converge on the desired global minimum. The only way to be sure is to find the minimum of all the local minima. The problem with reaching the desired global minimum with certainty is that it is difficult to precisely characterize in general the partitioning of the registration state space into local minima wells (regions of attraction) because this partitioning is potentially different for every possible different shape encountered.

To be precise, consider a 6-D state space Ω, where the quaternion component q_0 is determined from the other quaternion components: $q_0 = \sqrt{1-(q_1^2+q_2^2+q_3^2)}$. The actual state space Ω is a subset of the space $\Omega' = [-1,1]^3 \times \mathcal{R}^3$, where $\mathcal{R} = (-\infty, +\infty)$ is the real line. The subset Ω is specified by the "inside or on the unit 3 sphere" constraint that $q_1^2 + q_2^2 + q_3^2 \leq 1$. Therefore, Ω may be viewed as a type of hyper-cylinder in 6 space.

For any given nonpathological shape X that represents a real-world surface or object (e.g., pathological shape descriptions based on $\sin(1/x)$ near zero not allowed) and for any given point set P_{reg} already correctly registered with X, consider that any initial state $\vec{q} \in \Omega$ of the point set $P = \vec{q}(P_{\text{reg}})$ will converge to a local minimum as it is matched to X. There are a finite number of local minima $N_m(X,P)$ after one has fixed X and P. (The shape X is considered pathological if this is not true.) Let $\Psi(X,P)$ be the set of all local minima:

$$\Psi(X,P) = \{\psi_n\}_{n=1}^{N_m}. \tag{42}$$

This induces a natural partitioning of Ω into equivalence classes, labeled Ψ_n, where every value of $\vec{q}$ that converges via the ICP algorithm to ψ_n is a member of the class Ψ_n. This allows us to state that

$$\Omega = \bigcup_{n=1}^{N_m} \Psi_n \text{ and } \Psi_n \cap \Psi_m = \phi \text{ if } n \neq m. \tag{43}$$

Let Ψ_1 be the equivalence class that maps to the correct global minimum ψ_1.

To guarantee that the global minimum is found for a given shape X and a given set P not already registered with X, one

must use an appropriate set of initial states so that transforming P by at least one initial registration will place the point set into the correct equivalence class Ψ_1 of registrations. This allows it to converge to the correct global minimum ψ_1. Two fundamental questions are 1) how to construct an initial set of states for any given object that guarantees a correct global minimum and 2) how to construct an initial set of states that guarantees all shapes in a given class of shapes when those shapes converge to the correct respective global minimum.

By using a sufficiently dense uniform sampling of quaternions on the unit sphere combined with a sufficiently dense sampling of translation vectors occupying the total volume about the shape X, it is possible to determine the complete finite set of local minima with a sufficiently small probability of error for one given object. One could construct a set of initial states to include all these local minima solutions along with the halfway states between the k nearest neighbors (e.g., $k = 12$) to attempt to avoid having a set of initial states that lie on or near the boundaries between the equivalence classes.

A method that could be useful for computing guaranteed initial states for a set of model shapes is to use a 6-D occupancy array to compute hypervoxel-based descriptions of the equivalence classes for each shape of interest and then computing an overall partitioning by refining the first shape's partition by intersecting the equivalence classes of each subsequent shape. Such methods can be very memory intensive; a $20 \times 20 \times 20 \times 20 \times 20 \times 20$ hypercubic-hypervoxel grid of the smallest hypercylinder containing all relevant registrations of all shapes of interest requires an 8-Mbyte (64 Mb) array for a single object. Clearly, this is a test of patience for anyone wishing to construct an initial set of states customized to a given set of objects to yield guaranteed results, but indeed, it is possible and relatively straightforward, except for the high dimension of the problem.

A. Initial States for Global Matching

There are simpler methods of dealing with the initial state problem that are very effective on most shapes one comes across. Let us adopt the following definition of the first two moments of the distribution of geometry in P and X: $\vec{\mu}_p = E[\vec{p}|\vec{p} \in P]$, $\vec{\mu}_x = E[\vec{r}|\vec{r} \in X]$, $\Sigma_p = E[(\vec{p}-\vec{\mu}_p)(\vec{p}-\vec{\mu}_p)^t|\vec{p} \in P]$, and $\Sigma_x = E[(\vec{r}-\vec{\mu}_x)(\vec{r}-\vec{\mu}_x)^t|\vec{r} \in X]$, where $E[\cdot]$ represents the sample expectation (averaging) operator. If the point data set P covers a significant portion of the model shape X such that the condition

$$\alpha_1 \sqrt{tr(\Sigma_x)} \leq \sqrt{tr(\Sigma_p)} \leq \sqrt{tr(\Sigma_x)} \tag{44}$$

holds for a sufficiently large factor, say $\alpha_1 = 1/\sqrt{2} \approx 0.71$, then we have found that it is generally not necessary to use multiple initial translation states, as long as enough rotation states are used. This factor α_1 is the allowable occlusion percentage for global matching. The exact value of α_1 could be computed for any given class of object shapes via exhaustive testing if that is desired.

There are two reasonable options for the initial translation state: 1) Apply the ICP algorithm directly to the point set P using multiple rotation states about its center of mass $\vec{\mu}_p$, or 2) transform it first so that the centers of mass of P and X coincide, and then apply ICP. We have found no differences in the final registration results between 1) not translating and 2) pretranslating the point set when using an adequately large set of initial rotations (e.g., 24). In fact, any translation state suffices because the ICP algorithm is very insensitive to the initial translation state when used for global shape matching. We have observed that pretranslating the data point set's center of mass to the model shape's center of mass generally saves a few iterations (e.g., 2 to 4) out of the usual 20 or so total.

A further simplification in the global shape matching algorithm can be accomplished for most generic shapes, where principal moments demonstrate some level of distinctness. Let $p_x \geq p_y \geq p_z$ be the square roots of the eigenvalues of Σ_p, and let $r_x \geq r_y \geq r_z$ be the square roots of the eigenvalues of Σ_x. If the following sets of conditions hold:

$$p_y \leq \alpha_2 p_x \qquad p_z \leq \alpha_2 p_y \tag{45}$$

$$r_y \leq \alpha_2 r_x \qquad r_z \leq \alpha_2 r_y \tag{46}$$

for a specified value of α_2, e.g., $\alpha_2 = 1/\sqrt{2} \approx 0.71$, one can reliably match the basic shape structure of data and model using only the eigenvectors of the matrices Σ_p and Σ_x. Again, the exact value of α_2 could be computed for any given set of objects and any given level of sensor noise via exhaustive testing if needed. In this case of eigenvalue distinctness, the identity transformation and the 180° rotations about the eigenvector axes corresponding to r_x, r_y, and r_z provide a very good set of only four initial rotations that will yield the correct global minimum for a wide class of model shapes.

If two of the three eigenvalues are approximately equal but significantly different from the third for both data and model shapes, the number of initial states need only be expanded for rotations about the nonambiguous axis, thereby reducing the total number of initial rotational states.

If neither of the above cases for global matching hold true (i.e., $p_x \approx p_y \approx p_z$ and $r_x \approx r_y \approx r_z$), then one must use a fine sampling of quaternion states that cover the entire surface of the northern hemisphere of the unit 4 sphere uniformly.

The rotation groups of the regular polyhedra, which have been well known to crystallographers since the 1800's [29], provide a convenient set of uniformly sampled initial rotations: (a) 12 tetrahedral group states, (b) 24 octahedral/hexahedral group states, and (c) 60 icosahedral/dodecahedral group states. The tetrahedral states are a proper subset (subgroup) of the octahedral/hexahedral states and the icosahedral/dodecahedral states. The octahedral/hexahedral states are not properly contained in the icosahedral/dodecahedral states. For a convenient listing of these rotations in quaternion form, see Appendix A of [32].

From an implementation point of view, one has the option of using precomputed lists or nested loops. For the nested loop case, all normalized combinations of $q_0 = \{1, 0\}$, $q_1 = \{+1, 0, -1\}$, $q_2 = \{+1, 0, -1\}$, and $q_3 = \{+1, 0, -1\}$ provide an easy-to-compute set of 40 rotation states that includes the tetrahedral and the octahedral/hexahedral groups. (One must ensure that the first nonzero quaternion component is positive to avoid duplication of states.) For a really complicated set

of shapes, all normalized combinations of $q_0 = \{1, 0.5, 0\}$, $q_1 = \{+1, 0.5, 0, -0.5, -1\}$, $q_2 = \{+1, 0.5, 0, -0.5, -1\}$, and $q_3 = \{+1, 0.5, 0, -0.5, -1\}$ provides another easy-to-compute set of 312 initial rotation states. Another scheme for a very dense sampling of states is to refine the 60 states of the icosahedral group by subdividing triangles using a 1 to 4 split and using an increased number of rotations about each axis specified by each vertex of the refined icosahedron.

The general rule of thumb is the more complicated the object, the more initial states required. Any method of getting a sufficiently dense, uniform distribution of quaternions over the northern hemisphere of the unit 4 sphere (or over the full interior and surface of the unit 3 sphere) is adequate. In general, every application may want to use a customized set of initial quaternions that will maximize the probability of choosing a good starting point early for the shapes of interest.

B. A Counter Example

Although the above methods for global shape matching will work very well for many shapes with a small probability of error, we can also state categorically that for any given fixed set of initial rotation states, one can construct a shape X that cannot be correctly registered by the algorithm. Begin with a sphere of radius R. Add a thin spike of length S to the surface of the sphere for each specified axis and for each rotation about that axis as indicated by the fixed set of initial rotation states. Next, add one or more spikes of length $S + \epsilon$ anywhere that the extra spikes will fit. If the extras will not fit, make the original spikes thinner. Then, sample points on the surface of this shape with any desired scheme so that there is at least one point per spike, including one at the tip of each spike. The ICP algorithm combined with the given fixed set of initial rotation states will not be able to register a generic repositioning of this point set with the original object in such a way that the longer (or shorter) spikes are correctly registered with each other. It can safely be predicted that the proposed registration algorithm will have difficulty correctly registering "sea urchins" and "planets." These shapes are characterized as having almost exactly equal eigenvalues of the covariance matrix Σ_x and as having small shape features at a fine scale relative to the overall shape. Of course, for any given fixed set of object shapes, the set of initial rotations can be increased to guarantee correct registration.

C. Local Shape Matching

The proposed registration algorithm is definitely *not* useful if only a subset of the data point shape P corresponds to the model shape X or a subset of the model shape X, that is, the data point set includes a significant number of data points that do not correspond to any points on the model shape. Unfortunately, this is a trait of the majority of the shape matching algorithms that have ever been implemented. Moreover, this is a common problem in computer vision since data segmentation algorithms often misgroup one set of data points with another distinct group of points that should not be grouped together.

The ICP algorithm is still useful for local matching problems where the entire set of data points P matches a subset of the model shape X. The drawback is that more than one initial translation must be used, which increases the cost of computing the correct registration. If N_t is the number of initial translation states, N_q is the number of initial rotation states, and each closest point evaluation of P relative to X costs $O(N_p N_x)$, in the worst case, the total cost of local matching is $O(N_t N_q N_x N_p)$ as opposed to the cost of global matching $O(N_q N_x N_p)$. The number of initial translations is also dependent on the relative size of the data point set P compared with the model shape X. By defining a quantity η as the ratio of the "sizes" of the data and model shapes

$$\eta = \frac{m(X)}{m(P)} \geq 1 \tag{47}$$

where $m(\cdot)$ is a general measure that measures approximate 1) arc length if X is a curve, 2) area if X is a surface, and 3) volume if X is a volumetric point configuration, then one can estimate the basic qualitative behavior of the required number of translation states as $N_t \approx c(X)\eta$ for most shapes, where $c(X)$ is an approximate proportionality factor that depends on the complexity of the shape X being matched.

Computing such an estimate $m(X)$ is straightforward for the shape X, but evaluating $m(P)$ is more difficult because P may simply be a point set, and the corresponding length, area, and volume on the shape model X is unknown. One can use a convex hull, surface Delaunay, or closest point connection algorithm to get accurate measures for volumes, areas, and arc lengths, but it is more likely that one would design an algorithm to tolerate up to a particular percentage of occlusion, and a fixed set of translations would be computed to handle all objects in a given class of objects with up to that level of occlusion. Examples of local shape matching are demonstrated in the next section.

VI. Experimental Results

This section is divided into three sections: 1) point set matching, 2) curve matching, and 3) surface matching. All programs were written in C. Any quoted approximate times are given for execution on a single-processor computer rated at 1.6 Mflops on the 100×100 double-precision Linpack benchmark.

A. Point Set Matching

In this section, we demonstrate the ability of the ICP algorithm to perform local point set matching without correspondence. Table I lists a point set with eight points that is to be matched against a set of 11 points. Fig. 4 shows the two point sets prior to registration. Fig. 5 shows the two point sets after registration by the ICP algorithm after six iterations, which took less than 1 s. The computed registration is

```
   Translation: -48.078 6.65685 119.479
 Rotation Axis: (0.0321865 0.998188 -0.0508331)
Rotation Angle: 55.7188 degs
     RMS Error: 0.437608 mm
```

The algorithm does not pay attention to the extra points or to the ordering of the points because it always pairs a given

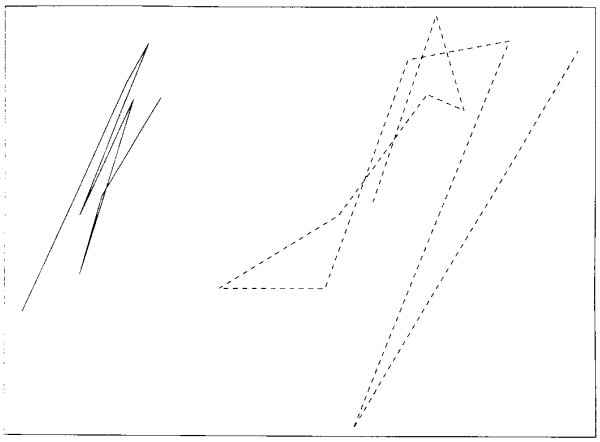


Fig. 4. Set 1 and set 2 prior to registration.

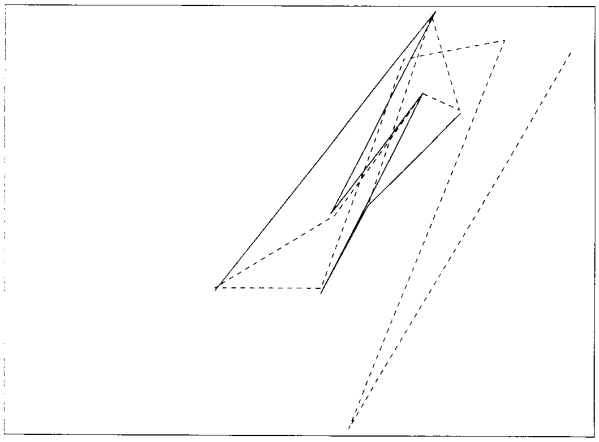


Fig. 5. Set 1 and set 2 after ICP registration.

TABLE I
TWO 3-D POINT SETS: SET 1 IS A SUBSET OF SET 2

$x1$	$y1$	$z1$	$x2$	$y2$	$z2$
43.89	-5.88	106.99	72.78	7.12	146.10
42.02	20.52	112.52	70.19	24.80	148.67
42.01	25.39	113.25	76.21	18.28	147.20
44.95	4.69	112.60	72.71	17.69	148.09
44.12	17.96	115.15	70.67	4.62	145.95
48.26	-1.37	113.59	64.38	-5.40	143.59
46.28	7.03	114.58	72.47	-1.16	143.85
47.00	18.52	117.65	69.82	19.81	148.32
			77.00	25.00	150.00
			80.00	-10.0	140.00
			83.00	30.00	145.00

point on the first set to the closest point on the other set. Only one initial rotation state and one initial translation state were used in this example of local matching. In general, one must use an initial set of rotations combined with an initial set of translations to achieve local matching. Compared with basic point set matching, which requires the same number of points listed in direct correspondence, we are essentially trading off additional CPU time for local matching capability and point order insensitivity.

1) Computational Issues: If one were to use a brute-force tree search (testing every possible correspondence and choosing the best one) in order to find the best match, this type of registration would require $\frac{N_x!}{(N_x-N_p)!}$ operations of the basic least squares quaternion match. For the simple example of local matching above with $N_p = 8$ and $N_x = 11$, a brute force test would require 6 652 800 quaternion registration operations. The above registration required only six operations since the initial state was already a member of the equivalence class of the global minimum. Even with 24 initial translation states and 60 initial rotation states allowing ten iterations for each combined initial state, we would require only 14 400 operations to provide an exhaustive and very capable matching ability. Moreover, it only takes a few minutes for these types of small point sets. The computation reduction ratio of the ICP algorithm compared with brute-force testing for this simple case is 462:1. Of course, other alternatives are possible, but we see that considering 1440 initial states is not unreasonable when the ICP algorithm is used to move from initial state to local minimum.

In the African mask example in the surface section below, we accurately registered a point set with N_p=2500 points to a point set with N_x=4200 points using 60 initial rotation states in less than 8 min. The amount of brute-force enumeration testing required for this case is ridiculously large; more than $1700^{2500} (> 10^{7500})$ operations are required. Even at 1 TeraOp/s (10^{12}) for the age of the universe 10^{18} s, this exact brute-force enumeration of all possible combinations would require well over 10^{250} universe lifetimes!

B. Curve Matching

In this section, the ability of the ICP algorithm to do local free-form curve matching is demonstrated. A 3-D parametric space curve spline was defined as a linear combination of cubic B-splines and control points. A copy of it was translated and rotated to be relatively difficult to match. The rotated and translated curve was converted to a polyline description with 64 points. Each x, y, z component of each point of the polyline was then corrupted by zero-mean Gaussian noise with a standard deviation of $\sigma = 0.1$ (compared with a curve size of $2.3 \times 2 \times 1$ units). The $\pm 3\sigma$ range of 0.6 units is clearly visible compared with the size of the object. We then cut off over half of the noisy polyline leaving a partial noisy curve shape. Fig. 6 shows the two space curves prior to registration. Fig. 7 shows the two space curves after ICP registration using 12 initial rotation states and six initial translation states for a total of 72 initial registration states. If the registration to move the curve away from the original curve is post-multiplied by the registration recovered using the ICP algorithm, we should obtain a matrix close to the identity matrix except for the effects of noise. The match of the partial noisy curve to the original spline curve yielded the following registration matrix,

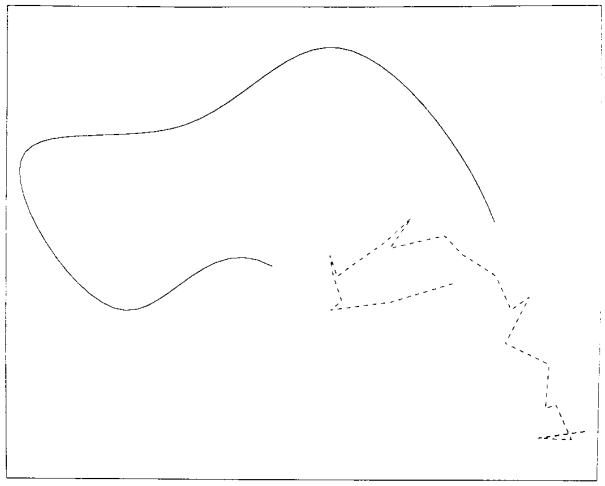


Fig. 6. Ideal and a partial noisy space curve before registration.

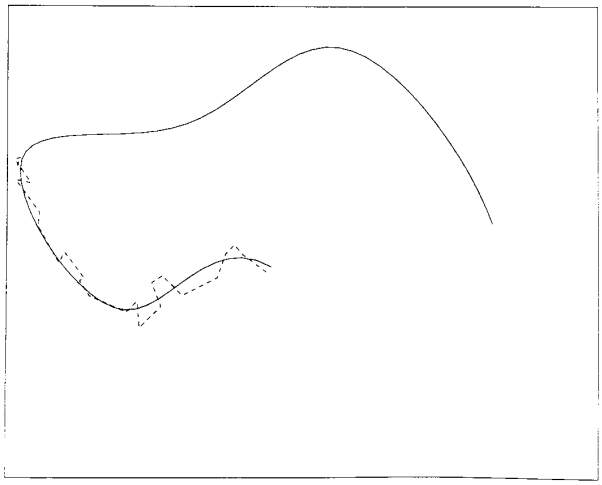


Fig. 7. Ideal and a partial noisy space curve after registration.

which is very close to the identity matrix:

```
Translation: 0.023540  0.006925 -0.015471
   Rotation: 0.999925 -0.012227 -0.001007
     Matrix: 0.012262  0.998647  0.050550
             0.000387 -0.050557  0.998721
```

C. Surface Matching

In this section, the ability of the ICP algorithm to register free-form surface shapes is demonstrated.

1) A Bezier Surface Patch: A simple parametric Bezier surface patch was constructed for quick testing of the free-form surface matching capability of the ICP algorithm. A set of 250 randomly positioned points was evaluated in the interior of the domain of the surface patch and translated and rotated in a random manner. The points of this point set are connected by lines indicating the point list sequence; they *do not* indicate line geometry to be matched. The surface patch is drawn

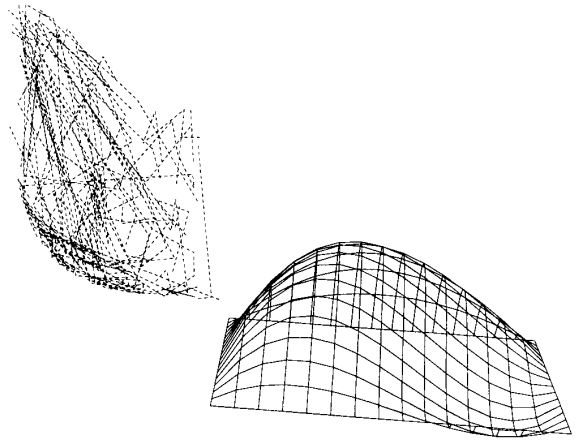


Fig. 8. Noisy point set and surface patch prior to registration: 250 points matched to 450 triangles.

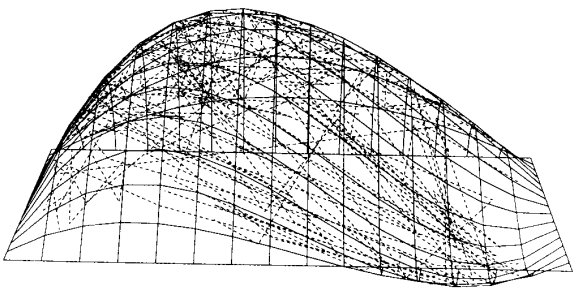


Fig. 9. Noisy point set and surface patch after registration.

via isoparametric lines indicating 450 triangles and fits in a $3 \times 3 \times 1$ units box. Following the space curve example, 3-D vector noise with a standard deviation of 0.1 units in each direction was added to the point set data to create a noisy point set. The surface patch and the noisy point set are shown prior to registration in Fig. 8. After running the ICP algorithm with same 24 initial rotation states for a total of about 3 min, we obtained the result shown in Fig. 9. The initial positioning transformation multiplied by the recovered transformation for the noisy point set yields the following approximate identity transformation:

```
Translation: -0.057329  0.013923  0.018430
   Rotation:  0.999357  0.034041  0.011264
     Matrix: -0.033883  0.999328 -0.013935
             -0.011731  0.013545  0.999840
```

This demonstrates that global matching under noisy conditions works quite well.

A subset of 138 noisy points was used to test the local matching ability. The surface patch and the noisy point subset are shown prior to registration in Fig. 10. After running the ICP algorithm with 24 initial rotation states and six initial translation states for a total of about 6 min, we obtained the result shown in Fig. 11. The initial positioning trans-

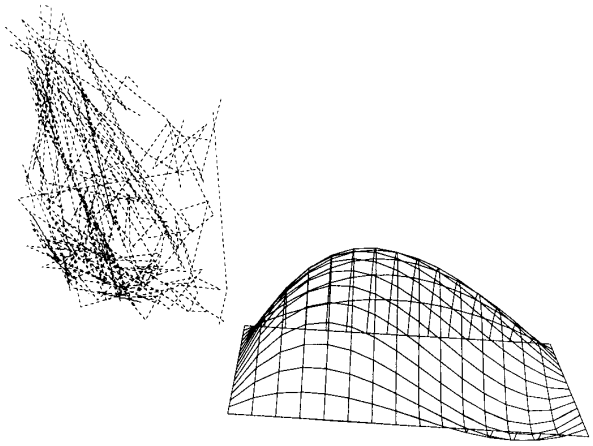


Fig. 10. Noisy point subset and surface patch prior to registration: 138 points and 450 triangles.

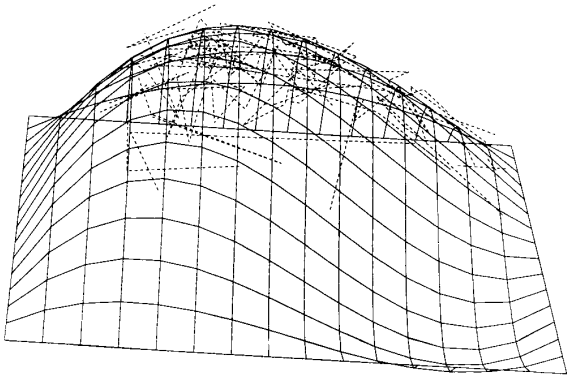


Fig. 11. Noisy point subset and surface patch after registration.

formation multiplied by the recovered transformation for the noisy point subset yields the following approximate identity transformation:

```
Translation: -0.166476  0.159480  0.128289
   Rotation:  0.990806  0.113548  0.073548
     Matrix: -0.113595  0.993521 -0.003545
             -0.073474 -0.004842  0.997285
```

This matrix approximation to the identity is much less precise than the global matching case, but the data is so noisy and the shape is so featureless that we found it surprising that the registration came out as well as it did.

2) The NRCC African Mask: In this experiment, range data from the National Research Council of Canada's African mask example was obtained using the commercially available Hyscan laser triangulation sensor from Hymarc, Ltd. A low-resolution 64 × 68 gridded image was computed from the original data set for use in our experiments and is shown in Fig. 12. This data will serve as the model surface description with 8442 triangles (4221 quadrilateral polygons). A thinned

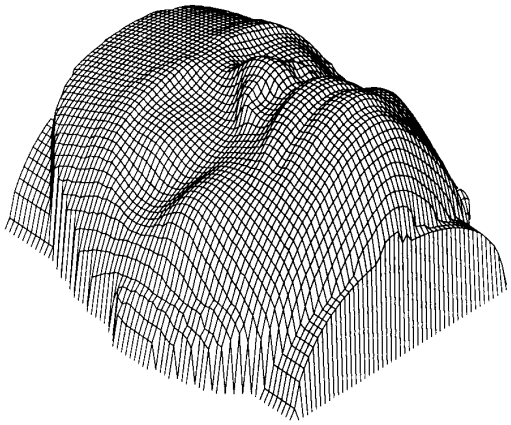


Fig. 12. Model surface: Range image of mask: 8442 triangles.

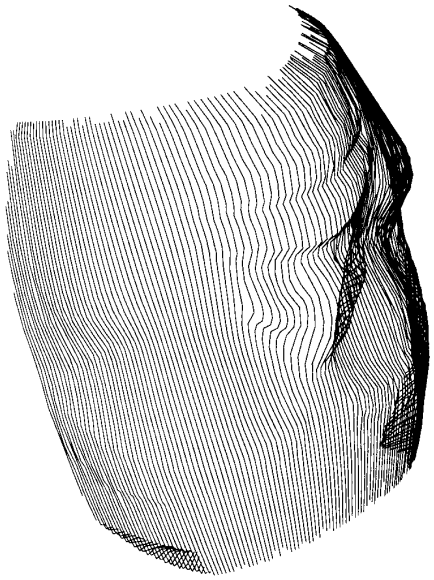


Fig. 13. Data: Rotated and translated point set of mask: 2546 points.

version of the measured data point set containing 2546 points is used as the data point set and is shown as scan lines in a test registration view in Fig. 13.

All trial positionings of the 90-mm object, including the one shown above, converged to the correct solution in about 10 min, and all cases had a 0.59-mm RMS error. This includes six iterations worth of testing on each of 24 initial state vectors and full iteration on the best state of the six initial iterations. A side view of both the digital surface model and the measured point set as registered is shown in Fig. 14. The registration is quite accurate.

A Bezier surface patch model of the mask was created to test the parametric surface matching capability on the given shape. This model is shown in in Fig. 15. The point set in various rotations and translations was then registered to the parametric surface model. We had expected about a 1.2-

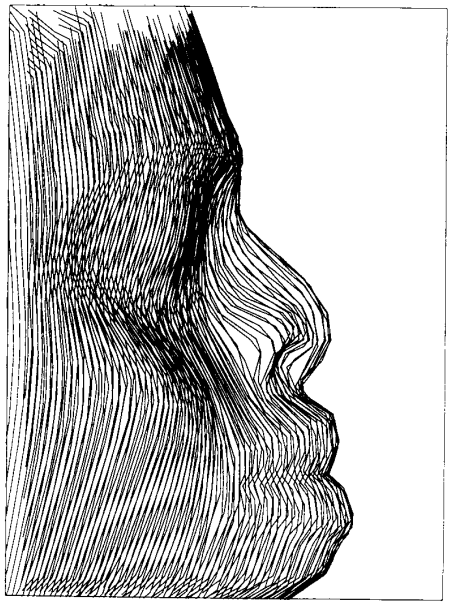


Fig. 14. Side view of range image model and registered point set.

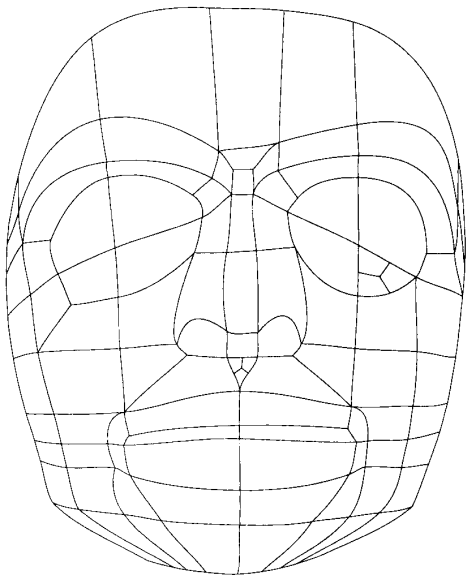


Fig. 15. Surface patch boundaries of a set of parametric surfaces: 97 cubic Bezier patches.

mm rms distance, but the final solution had a 3.4-mm rms distance. After examining the results closely, we discovered that surfaces had not been created in regions where there were measured points. Note the extra data points for which there was no possible surface correspondence and the evidence of misregistration in Fig. 16. Overall, this match was not bad considering the circumstances. After a post-processing step to ignore any measured points whose point-to-surface vectors were not within a few degrees of the surface normal at the corresponding points, the misregistration disappeared, and the data locked in on the surface with the expected rms distance. Although the ICP algorithm is not designed to handle data points that do not correspond to the model shape, one can conclude that a minor misgrouping of nearby points will usually have a minor effect. Another important point to keep in mind is that the matching algorithm does not care about the partitioning of the composite surface model into separate surface patches.

As a final test on the mask, about 30% of the points from the measured data point set were deleted as shown in Fig. 17. The registration algorithm locked in on the solution and gave a slightly improved rms distance in less time than the full data set.

3) Terrain Data: For the final set of experimental results, some terrain data for an area near Tucson was obtained from the University of Arizona. Fig. 18 shows a shaded image of the rugged terrain. The dimensions of the model surface are $6700 \times 6840 \times 1400$ units. A point set was extracted by performing 56 planar section cuts at regular intervals along one axis and then thinning out the data using a chord length deviation check. An interior section of this data set was extracted such that about 60% of the surface area of the original data set was covered. The resulting data set

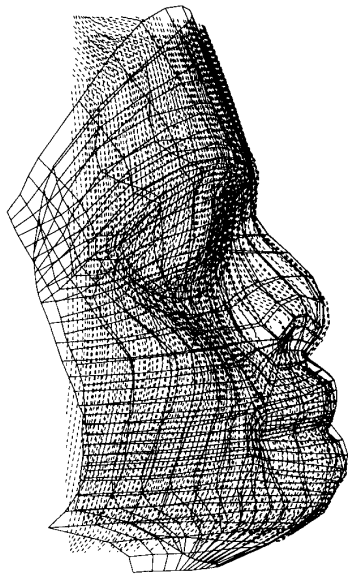


Fig. 16. Side view of parametric surface model and registered point set.

contained 13 655 points and is shown in Fig. 19. The data point set was then lifted above the model surface and rotated to be approximately orthogonal to the model set as shown in Fig. 20. The ICP algorithm performed local matching to the model surface using 24 initial rotations and one initial translation. The registration process for these larger data sets took about 1 hr. The results are shown in Fig. 21. The initial positioning transformation multiplied by the recovered transformation yields the following approximate identity transformation, which demonstrates that surface matching for very

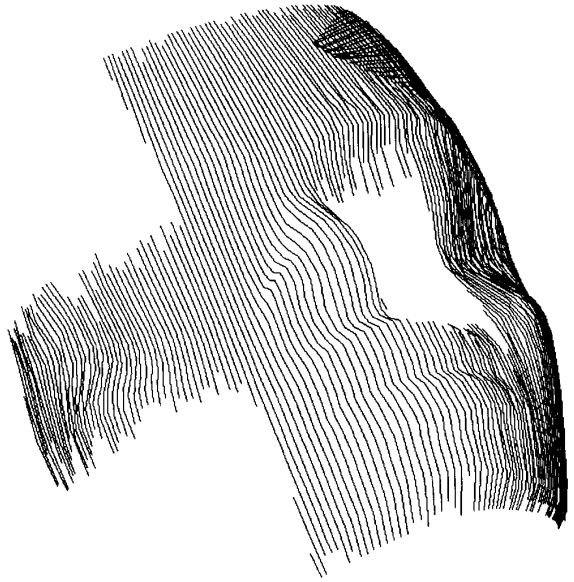


Fig. 17. Subset of the data point set: 1781 points.

Fig. 18. Model surface: Rugged terrain near Tucson, AZ: 45 900 triangles.

complex surface shapes works quite well:

```
Translation: -11.330336 1.404177  0.934043
   Rotation:   0.999994 0.003308 -0.000231
     Matrix:  -0.003309 0.999994 -0.000505
               0.000230 0.000505  1.000000
```

VII. Conclusions

The iterative closest point (ICP) algorithm is able to register a data shape with N_p points to a model shape with N_x primitives. The model shape can be a point set, a set of polylines, a set of parametric curves, a set of implicit curves, a set of triangles, a set of parametric surfaces, or a set of implicit surfaces. Any other type of shape representation can be incorporated if a procedure for computing the closest point on the model shape to a given point is available. If a data shape were to come in a form other than point set form, a dense set of points on the data shape can serve as the data point set.

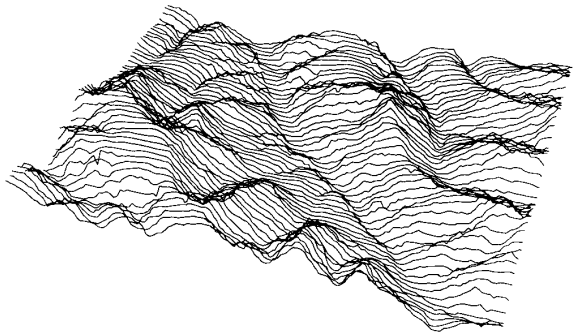


Fig. 19. Data: Rugged terrain near Tucson, AZ: 13 655 points.

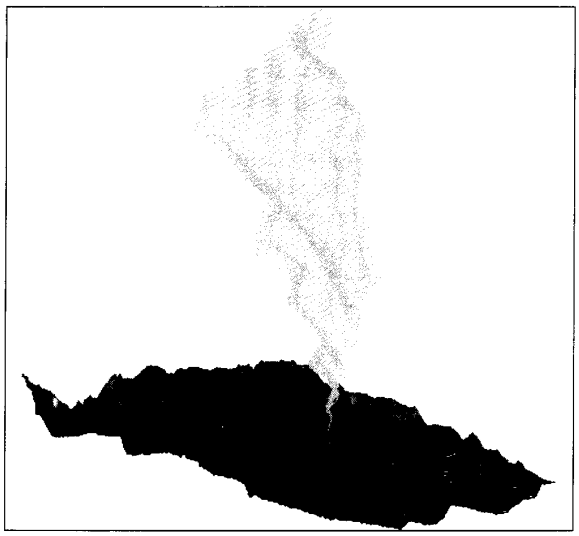


Fig. 20. Data and model of rugged terrain prior to registration.

Fig. 21. Data and model of rugged terrain after registration.

The accelerated ICP algorithm converges to a local minimum quickly in comparison with generic nonlinear optimization methods. It is fast enough that global shape matching can be achieved using a sufficiently dense sampling of unit quaternions used as initial rotation states, and local shape matching can be achieved by combining a sufficiently dense

sampling of relevant translations. The complexity of a single ICP iteration is $O(N_p N_x)$ in the worst case. For N_t initial translations and N_q initial rotations, the overall worst case complexity of local matching is $O(N_t N_q N_p N_x)$.

The advantages of the ICP shape matching algorithm are as follows:

- It handles the full six degrees of freedom.
- It is independent of shape representation.
- The surface patch or curve segment partitioning of parametric or implicit entities is essentially ignored by the matching procedure. This is important for using CAD data in its native form without elaborate user-guided preprocessing.
- It does not require preprocessing of 3-D point data, such as smoothing, as long as the number of statistical outliers is near zero. This is usually the case with accurate noncontact sensors used for inspection purposes.
- It does not require any derivative estimation or any local feature extraction.
- Almost all aspects of the algorithm are ideally suited for coarse-grain or fine-grain parallel architectures. For large problems, even remote execution procedures and distributed file systems on networks of workstations can provide worthwhile speedup without significant overhead.
- Global matching is achieved at predictable cost based on shape complexity.
- Local matching is achieved at predictable cost based on shape complexity and the percentage of allowable occlusion.
- It can handle a reasonable amount of normally distributed vector noise, with standard deviations of up to 10% of object size demonstrated above.
- The method generalizes to n dimensions by substituting the SVD algorithm [23] for the quaternion algorithm with the added feature that reflections are allowed.
- The method can be made statistically robust by substituting iterations of the SVD-based algorithm by Haralick *et al.* [28] for the quaternion algorithm to identify outliers. The increased computational requirements are significant.
- It can easily be used in conjunction with other algorithms, such as the covariance matrix alignment, which preorient the data so that fewer initial rotation states are necessary.
- Shapes with three sufficiently distinct principal moments (eigenvalues) can be globally matched at a cost of only four initial rotation states.
- It is relatively insensitive to minor data segmentation errors as indicated by the performance of the registration of points with the African mask parametric surface model.
- The results of the last iteration of closest point registration can be used directly as inspection results since the distance to the closest point on a surface is computed as a byproduct.
- The accelerated ICP algorithm can achieve Newton-type quadratic convergence steps at less cost than a numerical steepest descent step. No time is spent evaluating the objective function to find worse mean square errors off the path to the local minimum goal.

The disadvantages of the ICP shape matching algorithm are as follows:

- It is susceptible to gross statistical outliers unless a statistically robust method is substituted at some point (either in preprocessing or registration computation) for outlier detection.
- The fast least squares quaternion and SVD methods are not so easily adapted to weighted least squares extensions, meaning that it is difficult to extend the *fast* algorithm to allow the assignment of unequal uncertainties to points, as was done in [54]. This is not a major inconvenience for inspection applications but would yield noticably suboptimal results for navigation laser radars and long depth of field triangulation systems, where uncertainty increases significantly with range.
- The cost of local matching can get quite large for small allowable occlusion percentages, e.g., 10% or less. We do not advocate our proposed method if feature extraction techniques will successfully solve the problem.
- The generalization to matching deformable models with high order deformations [57] is not straightforward without, e.g., enumerating a dense set of possible deformations.
- As an extension of the outlier rejection issue, the stated algorithm does not solve the segmentation problem, of course. If data points from two shapes are intermixed and matched against the individual shapes, the registrations will be wrong, and the mean square distance metric will be large. This is a problem with almost all of the shape matching algorithms in the literature.
- For any given fixed initial set of rotations, the global shape matching capability can be defeated even without sensor noise by constructing "sea urchin" or "planetoid" shaped objects based on the set of rotations such that correct registration cannot be guaranteed. On the other hand, for a fixed set of objects and no sensor noise, one can determine an initial set of registrations in a finite amount of time such that one can guarantee registration in a finite amount of time with a sufficiently small probability of error.
- In the limit of very complicated sea urchins or perfectly spherical planets with a single 1 μm bump or in the limit of very localized matching (1% of object shape or less) on any object, the ICP algorithm degenerates to brute-force 3-D template matching. Feature extraction techniques, if possible, are preferable in such circumstances.

VIII. Future Directions

Although we have tried to present compelling results, no method with such promise is likely to be widely accepted until more testing has been done. We believe the algorithm for point sets, polyline sets, and triangle sets are simple enough to be widely implemented and tested.

The accelerated ICP algorithm is quite efficient, but there is plenty of room for further computational speedup. For example, we have not discussed the use of $k-d$ trees to ensure expected cost of $O(\log N_x)$ rather than the worst case $O(N_x)$

performance in the closest point operation. Actual testing on parallel architectures also needs to be done.

Given a large but finite amount of off-line processing, it seems reasonable to make the following statements: For a fixed set of objects, a given level of allowable occlusion, a given maximum possible (Gaussian) noise level, and a set of initial registration states, it is possible to estimate the probability of registration failure by carrying out exhaustive tests. For a 1% (of the smallest shape size) noise level and a 40% maximum allowable occlusion, our tests indicate that very low probabilities of failure should be achievable with minimal extra work. The method needs to be characterized in such detail.

A single complicated object in a set of simpler objects may require a large set of initial registrations to handle the entire set of objects with certainty. Unfortunately, much time is then spent testing initial registrations with simpler objects such that one is continually homing in on the same local minimum over and over again. With some extra bookkeeping, it may be possible to quickly recognize a familiar local minimum well that you have already fallen into a few times and abort that iteration or to use penalty function methods to penalize walking down a path that has already been explored. The appropriate shape of the penalty functions would depend on the shape of the region of attraction in 6-D, which is difficult to quantify and analyze.

It may be possible to extend the basic least squares registration solution to allow deformations (independent axis scaling and bending) of the model shapes when matching to the data shapes. Shears and separate axis scaling can be easily accomodated by allowing a general affine transformation; allowing even quadratic bending about the center of mass significantly complicates matters.

Finally, these free-form shape matching methods are likely to be useful as part of a 3-D object recognition system.

ACKNOWLEDGMENT

The authors wish to thank A. Morgan, R. Khetan, R. Tilove, W. Wiitanen, R. Bartels, D. Field, W. Meyer, C. Wampler, D. Baker, R. Smith, N. Sapidis, and the reviewers for their comments.

REFERENCES

[1] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least square fitting of two 3-D point sets," *IEEE Trans. Patt. Anal. Machine Intell.* vol. PAMI-9, no. 5, pp. 698–700, 1987.

[2] R. Bajcsy and F. Solina, "Three-dimensional object representation revisited," *Proc. 1st Int. Conf. Comput. Vision* (London), June 8–11, 1989, pp. 231–240.

[3] P. J. Besl, "Geometric modeling and computer vision," *Proc. IEEE*, vol. 76, no. 8, pp. 936–958, Aug. 1988.

[4] P. J. Besl, "Active optical range imaging sensors," in *Advances in Machine Vision* (J. Sanz, Ed.). New York: Springer-Verlag, 1989; see also *Machine Vision and Applications*, vol. 1, pp. 127–152, 1989.

[5] ——, "The free-form surface matching problem," *Machine Vision for Three-Dimensional Scenes* (H. Freeman, Ed.). New York: Academic, 1990.

[6] P. J. Besl and R. C. Jain, "Three dimensional object recognition," *ACM Comput. Surveys*, vol. 17, no. 1, 75–145, March 1985,

[7] J. Blumenthal, "Polygonizing implicit surfaces," Xerox Parc Tech. Rep. EDL-88–4, 1988.

[8] B. Bhanu and C. C. Ho, "CAD-based 3-D object representation for robot vision," *IEEE Comput.*, vol. 20, no. 8, pp. 19–36, Aug. 1987.

[9] W. Boehm, G. Farin, and J. Kahmann, "A survey of curve and surface methods in CAGD," *Comput. Aided Geometric Des.*, vol. 1, no. 1, pp. 1–60, July 1984.

[10] R. C. Bolles and P. Horaud, "3DPO: A three-dimensional part orientation system," *Int. J. Robotic Res.*, vol. 5, no. 3, pp. 3–26, Fall 1986.

[11] P. Brou, "Using the Gaussian image to find the orientation of an object," *Int. J. Robotics Res.*, vol. 3, no. 4, pp. 89–125, 1983.

[12] J. Callahan and R. Weiss, "A model for describing surface shape," in *Proc. Conf. Comput. Vision. Patt. Recogn.* (San Francisco, CA), June 1985, pp. 240–247.

[13] C. H. Chen and A. C. Kak, "3DPOLY: A robot vision system for recognizing 3-D objects in low-order polynomial time," Tech. Rep. 88–48, Elect. Eng. Dept., Purdue Univ., West Lafayette, IN, 1988.

[14] R. T. Chin and C. R. Dyer, "Model-based recognition in robot vision," *ACM Comput. Surveys*, vol. 18, no. 1, pp. 67–108, Mar. 1986.

[15] C. DeBoor, *A Practical Guide to Splines*. New York: Springer-Verlag, 1978.

[16] C. DeBoor and K. Hollig, "B-splines without divided differences," *Geometric Modeling: Algorithms and New Trends* (G. Farin, Ed.), *SIAM*, pp. 21–28, 1987.

[17] G. Farin, *Curves and Surfaces in Computer Aided Geometric Design: A Practical Guide*. New York: Academic, 1990.

[18] O. D. Faugeras and M. Hebert, "The representation, recognition, and locating of 3-D objects," *Int. J. Robotic Res.* vol. 5, no. 3, pp. 27–52, Fall 1986.

[19] T. J. Fan, "Describing and recognizing 3-D objects using surface properties," Ph.D. dissertation, IRIS-237, Inst. Robotics Intell. Syst., Univ. of Southern California, Los Angeles, 1988.

[20] P. J. Flynn and A. K. Jain, "CAD-based computer vision: From CAD models to relational graphs," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 13, no. 2, pp. 114–132, 1991; see also Ph.D. Thesis, Comput. Sci. Dept., Michigan State Univ., E. Lansing, MI.

[21] E. G. Gilbert and C. P. Foo, "Computing the distance between smooth objects in 3D space," RSD-TR-13–88, Univ. of Michigan, Ann Arbor, 1988.

[22] E. G. Gilbert, D. W. Johnson, S. S. Keerthi, "A fast procedure for computing the distance between complex objects in 3D space," *IEEE J. Robotics Automat.*, vol. 4, pp. 193–203, 1988.

[23] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Baltimore, MD: Johns Hopkins Univ. Press, 1983.

[24] W. E. L. Grimson, "The combinatorics of local constraints in model-based recognition and localization from sparse data," *J. ACM*, vol. 33, no. 4, pp. 658–686, 1986.

[25] W. E. L. Grimson and T. Lozano-Pérez, "Model-based recognition and localization from sparse range or tactile data," *Int. J. Robotics Res.*, vol. 3, no. 3, pp. 3–35, Fall 1984.

[26] K. T. Gunnarsson and F. B. Prinz, "CAD model-based localization of parts in manufacturing," *IEEE Comput.*, vol. 20, no. 8, pp. 66–74, Aug. 1987.

[27] E. Hall, J. Tio, C. McPherson, and F. Sadjadi, "Measuring curved surfaces for robot vision," *Comput.* vol. 15, no. 12, pp. 42–54, Dec. 1982.

[28] R. M. Haralick *et al.*, "Pose estimation from corresponding point data," in*Machine Vision for Inspection and Measurement* (H. Freeman, Ed.). New York: Academic, 1989.

[29] H. Hilton, *Mathematical Crystallography and the Theory of Groups of Movements*. Oxford: Clarendon, 1963; London: Dover, 1963.

[30] B. K. P. Horn, "Extended Gaussian images," *Proc. IEEE*, vol. 72, no. 12, pp. 1656–1678, Dec. 1984

[31] ——, "Closed-form solution of absolute orientation using unit quaternions," *J. Opt. Soc. Amer. A* vol. 4, no. 4, pp. 629–642, Apr. 1987

[32] ——, "Relative orientation," A.I. Memo 994, AI Lab, Mass. Inst. Technol., Cambridge, Sept. 1987.

[33] B. K. P. Horn and J. G. Harris, "Rigid body motion from range image sequences," *Comput. Vision Graphics Image Processing*, 1989.

[34] K. Ikeuchi, "Generating an interpretation tree from a CAD model for 3-D object recognition in bin-picking tasks," *Int. J. Comput. Vision*, vol. 1, no. 2, pp. 145–165, 1987.

[35] A. K. Jain and R. Hoffman, "Evidence-based recognition of 3-D objects," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 10, no. 6, pp. 793–802, 1988.

[36] B. Kamgar-Parsi, J. L. Jones, and A. Rosenfeld, "Registration of multiple overlapping range images: Scenes without distinctive features," *Proc. IEEE Comput. Vision Patt. Recogn. Conf.* (San Diego, CA), June 1989.

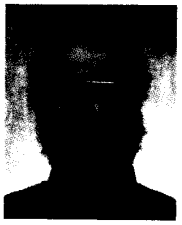
[37] Y. Lamdan and H. J. Wolfson, "Geometric hashing: A general and efficient model-based recognition scheme," *Proc. 2nd Int. Conf. Comput. Vision* (Tarpon Springs, FL), Dec. 5–8, 1988, pp. 238–251.

[38] S. Z. Li, "Inexact matching of 3D surfaces," VSSP-TR-3–90, Univ. of Surrey, England, 1990.
[39] P. Liang, "Measurement, orientation determination, and recognition of surface shapes in range images," Cent. Robotics Syst., Univ. of California, Santa Barbara, 1987.
[40] D. Luenberger, *Linear and Nonlinear Programming*. Reading, MA: Addison-Wesley, 1984.
[41] A. P. Morgan, *Solving Polynomial Systems Using Continuation for Engineering and Scientific Problems*. Englewood Cliffs, NJ: Prentice-Hall, 1987.
[42] M. E. Mortenson, *Geometric Modeling*. New York: Wiley, 1985.
[43] J. Mundy *et al.*, "The PACE system," in *Proc. CVPR '88 Workshop*; see also *DARPA IUW*.
[44] D. W. Murray, "Model-based recognition using 3D shape alone.," *Comput. Vision Graphics Image Processing*, vol. 40, pp. 250–266, 1987.
[45] Press, Flannery, Teukolsky, and Vetterling, *Numerical Recipes in C*. Cambridge, UK: Cambridge University Press, 1988.
[46] J. Rieger, "On the classification of views of piecewise smooth objects," *Image and Vision Computing*, vol. 5, no. 2, pp. 91–97, 1987.
[47] *Proc. IEEE Robust Methods Workshop*. Univ. of Washington, Seattle.
[48] P. Sander, "On reliably inferring differential structure from 3D images," Ph.D. dissertation, Dept. of Elect. Eng., McGill Univ., Montreal, Canada, 1988.
[49] P. H. Schonemann, "A generalized solution to the orthogonal procrustes problem," *Psychometrika*, vol. 31, no. 1, 1966
[50] J. T. Schwartz and M. Sharir, "Identification of partially obscured objects in two and three dimensions by matching noisy characteristic curves," *Int. J. Robotics Res.*, vol. 6, no. 2, pp. 29–44, 1987.
[51] T. W. Sederberg, "Piecewise algebraic surface patches," *Comput. Aided Geometric Des.*, vol. 2, no. 1, pp. 53–59, 1985.
[52] B. M. Smith, "IGES: A key to CAD/CAM systems integration," *IEEE Comput. Graphics Applications*, vol. 3, no. 8, pp. 78–83, 1983.
[53] G. Stockman, "Object recognition and localization via pose clustering," *Comput. Vision Graphics Image Processing*, vol. 40, pp. 361–387, 1987.
[54] R. Szeliski, "Estimating motion from sparse range data without correspondence," *2nd Int. Conf. Comput. Vision* (Tarpon Springs, FL), Dec. 5–8, 1988, pp. 207–216.
[55] G. Taubin, "Algebraic nonplanar curve and surface estimation in 3-space with applications to position estimation," Tech. Rep. LEMS-43, Div. Eng., Brown Univ., Providence, RI, 1988.
[56] ——, "About shape descriptors and shape matching," Tech. Rep. LEMS-57, Div. Eng., Brown Univ., Providence, RI, 1989.
[57] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer, "Elastically deformable models." *Comput. Graphics*, vol. 21, no. 4, pp. 205–214, July 1987.
[58] B. C. Vemuri, A. Mitiche, and J. K. Aggarwal, "Curvature-based representation of objects from range data," *Image Vision Comput.*, vol. 4, no. 2, pp. 107–114, May 1986.
[59] B. C. Vemuri and J. K. Aggarwal, "Representation and recognition of objects from dense range maps," *IEEE Trans. Circuits Syst.*, vol. CAS-34, no. 11, pp. 1351–1363, Nov. 1987.
[60] A. K. C. Wong, S. W. Lu, and M. Rioux, "Recognition and shape synthesis of 3-D objects based on attributed hypergraphs," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 11, no. 3, pp. 279–290, Mar. 1989.

Paul J. Besl (M'86) received the B.S. degree *summa cum laude* in physics from Princeton University, Princeton, NJ, in 1978 and the M. S. and Ph. D. degrees in electrical engineering and computer science from the University of Michigan, Ann Arbor, in 1981 and 1986, respectively.

Prior to 1984, he worked for Bendix Aerospace Systems in Ann Arbor and Structural Dynamics Research Corporation, Cincinnati, OH. Since 1986, he has been a research scientist at General Motors Research Laboratories, Warren, MI, where his primary research interest is computer vision, especially the practical applications of range imaging sensors.

Dr. Besl is a member of the Association for Computing Machinery. He received a Rackham Distinguished Dissertation Award for his Ph. D. work on range image understanding, which was later published in book form by Springer Verlag.



Neil D. McKay received the B. S. degree in electrical engineering from the University of Rochester, Rochester, NY, in 1980, the M.S. degree in computer and systems engineering from Rensselaer Polytechnic Institute in 1982, and the Ph. D. degree in electrical engineering from the University of Michigan in 1985.

He has been with the Computer Science Department of General Motors Research Laboratories since 1985. His interests include robot path planning and optimal control.