Summary of "A Method for Registration of 3-D Shapes"

Original Article by Paul J. Besl

Tianhe Yang

Optics Department
University of Michigan

What is Iterative Closest Point?

Definition

An ICP algorithm attempts to match two sets of points. One of these sets might be a reference image, while the other is a set of data points describing the ranges to certain points on an object.

Basic ICP Steps

In general ICP has to select some set of points in one or both meshes, match these points to samples in the other set, assigning a definition for error, minimizing that definition of error by iteration. The model shape can be a point set, a set of polylines, a set of parametric curves, a set of implicit curves, or a set of implicit surfaces.

Goals

Given

- 3D data in a sensor coordinate system.
- Model shape in model coordinate system.

Goals

Produce

- Optimal rotation or translation that aligns model shape and data shape.
- This "alignment" is called registration.
- Goal is to minimize the distance between model shape and data shape.
- The error is measured by a mean-square distance metric.

Application

- Register (align) digitized data from rigid objects with an idealized model (CAD, etc.)
- Inspect to see if shape of object is within specs.

Depth Measurement

- Using high-accuracy measurement tools over a shallow depth of field, uncertainty in different points does not change very much.
- Measurement device doesn't generate bad data; no outliers.

The Norm

Distance Between Two Points

The distance $d(\vec{r}_1, \vec{r}_2)$ where $\vec{r}_1 = \{x_1, y_1, z_1\}$ and $\vec{r}_2 = \{x2, y2, z2\}$ is:

$$d(\vec{r}_1, \vec{r}_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2)} \tag{1}$$

This is just the equation for the norm of a vector.

Distance to a Set of Points

The distance to a set of N_a points, A, from a point $\vec{p}$ is the distance to the closest point in A. $A = \{a_i\}$

$$d(\vec{p}, A) = \min_{i \in \{1,\ldots,N_a\}} d(\vec{p}, \vec{a}_i) \tag{2}$$

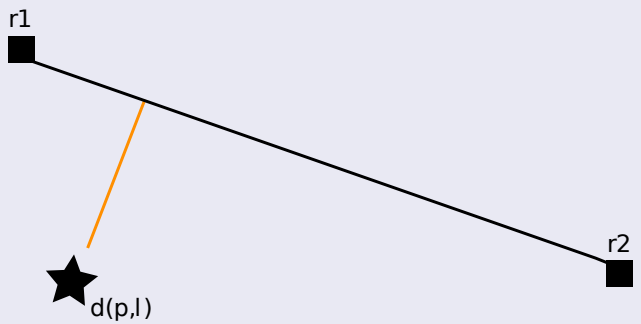
For a Line

Distance to a Line

For a line segment l connecting two points $\vec{r}_1$ and $\vec{r}_2$, the distance between the point $\vec{p}$ (typo in the paper) and the line segment l is:

$$d(\vec{p}, l) = \min_{u+v=1} \|u\vec{r}_1 + v\vec{r}_2 - \vec{p}\| \tag{3}$$

where $u \in [0, 1]$, $v \in [0, 1]$.



For N_l Lines

For a set of N_l lines, L

N_l line segments are denoted by l_i, and $L = \{l_i\}$ for $i = 1, \ldots, N_l$. The distance between point $\vec{p}$ and the line segments L is thus:

$$d(\vec{p}, L) = \min_{i \in \{1,\ldots,N_l\}} d(\vec{p}, l_i) \tag{4}$$

For a Triangle

Distance Between a Point and a Triangle

We can similarly state that the distance between a point $\vec{p}$ and a triangle t is:

$$d(\vec{p}, t) = \min_{u+v+w=1} \|u\vec{r}_1 + v\vec{r}_2 + w\vec{r}_3 - \vec{p}\| \tag{5}$$

where $u \in [0, 1]$, $v \in [0, 1]$, and $w \in [0, 1]$.

For N_t Triangles

Distance Between a Point and a Set of Triangles

Let T be a set of N_t triangles denoted t_i, and let $T = \{t_i\}$ for $i = 1, \ldots, N_t$. The distance between a point $\vec{p}$ and the set of triangles T can be written as:

$$d(\vec{p}, T) = \min_{i \in \{1,\ldots,N_t\}} d(\vec{p}, t_i) \tag{6}$$

Review of Parametric Curves

Why Use Parametric Curves?

For simple functions such as $y = mx + b$, there is no need for parametric functions to describe this object. However, as curves get more complex, we may not be able to describe them in this form. For example, consider a sphere $(x^2 + y^2 = r^2)$. To describe the whole sphere, we need at least two equations $(x = \sqrt{r^2 - y^2}$ for the right side, and $x = -\sqrt{r^2 - y^2})$ for the left side.

Parametric Curves

Instead of defining a variable as a function of another variable, we can define the other variable as a function. For the circle example, x and y can be parameterized to $x = r\cos t$ and $y = r\sin t$.

Parametric Curves and Surfaces

Parametric Entities

This paper treats parametric curves and surfaces as "single entities". This is just saying what was in the previous slide. For example, a parametric curve is written $\vec{r}(\vec{u})$ where $\vec{u} \in \Re^1$. Parametric surfaces have $\vec{u} = (u, v)$ where $\vec{u} \in \Re^2$. $\Re$ is all real numbers.

Distance From a Point $\vec{p}$ to a Parametric Entity E

$$d(\vec{p}, E) = \min_{\vec{r}(\vec{u}) \in E} d(\vec{p}, \vec{r}(\vec{u})) \tag{7}$$

Point-to-Curve/Surface Distance

Parametric Entities

We can state the distance from a point to a curve/surface as we have done before. Let F be the set of N_e parametric entities denoted Ei, and $F = \{E_i\}$ for $i = 1, \ldots, N_e$ (note error in paper). The distance between a point $\vec{p}$ and the set of parametric entities F is then:

$$d(\vec{p}, F) = \min_{i \in \{1,\ldots,N_e\}} d(\vec{p}, \vec{E}_i) \tag{8}$$

Approximation of a Curve

Polyline Approximation

Polyline approximation is using multiple lines to approximate a curve, such that the maximum deviation from the curve is never more than δ. For a parametric curve $C = \{\vec{r}(u)\}$, one can compute a polyline $L(C, \delta)$ such that each vertex of the polyline gives an estimate, u_a, of the corresponding u argument values of the parametric curve.

Approximation of a Surface

Triangular Approximation

Similarly, triangles $T(S, \delta)$ can be used to approximate a parametric surface $S = \{\vec{r}(u, v)\}$. The vertices of the triangles must not deviate from the surface by more than a distance δ. Vertices of the triangle are tagged with corresponding points (u, v), and give an estimate (u_a, v_a). One can assume that the initial value of $\vec{u}_a$ is very close to the corresponding point on the surface.

Minimizing Distance to a Surface

Newtom's Minimization Approach

If we have an accurate starting value $\vec{u_a}$, we can use a Newton's minimization approach. The function to be minimized is:

$$f(\vec{u}) = \|\vec{r}(\vec{u}) - \vec{p}\|^2 \tag{9}$$

If $\nabla = [\frac{\partial}{\partial \vec{u}}]^t$ is the vector differential gradient operator, the minimum of f occurs when $\nabla f = 0$.

Surface Gradient Vector

2D Hessian Matrix

When the parametric entity is a surface, the gradient vector becomes $\nabla f = [f_u, f_v]^2$, where $f_u(\vec{u}) = 2r_u^{-t}(\vec{u})(\vec{r}(\vec{u}) - \vec{p})$ and $f_v(\vec{u}) = 2r_v^{-t}(\vec{u})(\vec{r}(\vec{u}) - \vec{p})$. The second-order partial derivative of f can be written in the form of a matrix:

$$\nabla\nabla^t f = \begin{pmatrix} f_{uu} & f_{uv} \\ f_{uv} & f_{vv} \end{pmatrix}$$

Where $f_{uu}(\vec{u}) = 2r_{uu}^{-t}(\vec{u})(\vec{r}(\vec{u}) - \vec{p}) + 2r_u^{-t}(\vec{u})\vec{r}_u(\vec{u})$, $f_{vv}(\vec{u}) = 2r_{vv}^{-t}(\vec{u})(\vec{r}(\vec{u}) - \vec{p}) + 2r_v^{-t}(\vec{u})\vec{r}_v(\vec{u})$, and $f_{uv}(\vec{u}) = 2r_{uv}^{-t}(\vec{u})(\vec{r}(\vec{u}) - \vec{p}) + 2r_u^{-t}(\vec{u})\vec{r}_v(\vec{u})$

Converging Update Formula

The "I" in ICP

Newton's update formula is given by:

$$\vec{u}_{k+1} = \vec{u}_k - [\nabla\nabla^t(f)(\vec{u}_k)]^{-1}\nabla f(\vec{u}_k) \tag{10}$$

Where $\vec{u}_0 = \vec{u}_a$. Using a small δ, Newton's method for computing the closest point usually converges in one to five iterations, and typically three. The computational cost of this iterative method is very low compared to the cost of finding good starting points.

What are Implicit Functions?

Explicit Functions

Explicit functions have dependent variables that are given "explicitly." The following is an example of an explicit function:

$$y = f(x) \tag{11}$$

What are Implicit Functions?

Implicit Functions

A function is implicit if one must solve an equation to obtain y from x[1]

$$R(x, y) = 0 \tag{12}$$

One variable or the other may determine each other, but an explicit formula is not given for one in terms of another. R may be a multiple-valued "function" (an element in one set may be associated with multiple elements in the other set).

Implicit Entities

Distance to an Implicit Entity

The geometric entity is defined as the zero set of a function $\vec{g}(\vec{r})$. The zero set of this function is a subset of $\vec{r}$ such that $\vec{g}(\vec{r}) = 0$. The distance to an implicit entity I can be expressed as:

$$d(\vec{p}, I) = \min_{\vec{g}(\vec{r})=0} d(\vec{p}, \vec{r}) = \min_{\vec{g}(\vec{r})=0} \|\vec{r} - \vec{p}\| \tag{13}$$

Distance Between a Point and a Set of Parametric Entities

Let J be a set of N_I parametric entities denoted I_k and $J = \{I_k\}$ for $k = 1, \ldots, N_I$. The distance between a point $\vec{p}$ and the set J is:

$$d(\vec{p}, J) = \min_{k \in \{1,\ldots,N_I\}} d(\vec{p}, \vec{I}_k) \tag{14}$$

Implicit Entities

Computing the Distance from a Point

The first step to finding the closest point requires an approximation similar to the one we made before for parametric entities, and involves creating lines or triangles to approximate the surface. Using this method, we again get an approximate closest argument, $\vec{r}_a$.

Implicit Entity Problem vs. Parametric Entity Problem

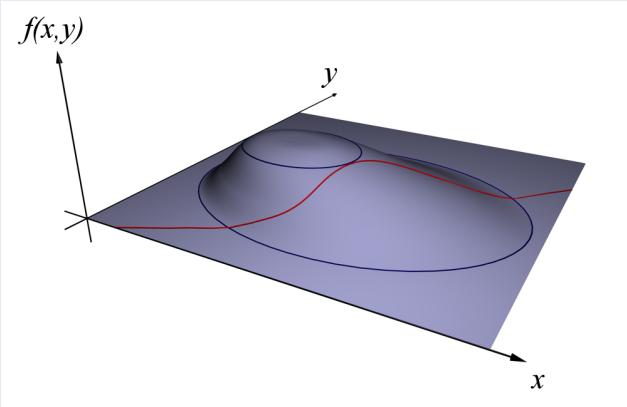
To find the closest point on an implicit entity $\vec{g}(\vec{r}) = 0$ to a given point $\vec{p}$, we need to minimize:

$$f(\vec{r}) = \|\vec{r} - \vec{p}\|^2 \text{ given the condition that } \vec{g}(\vec{r}) = 0 \tag{15}$$

Implicit Entities

Lagrange Multipliers

The previous requirements fit the Langrage Multipliers strategy of solving this problem; the Lagrange Multipliers method provides a strategy for finding the maximum/minimum of a function subject to constraints[2]. A full derivation can be found in the cited text.



In this figure, we want to find the maximum point in $f(x, y)$ (gray) given that $g(x, y)$ (red) is subject to the constraint $g(x, y) = c$.

Implicit Entities

Lagrange Multipliers

A Lagrange function is defined by:

$$\Lambda(x, y, \lambda) = f(x, y) - \lambda(g(x, y) - c) \tag{16}$$

To maximize $f(x, y)$, we solve the following:

$$\nabla\Lambda(x, y, \lambda) = 0 \tag{17}$$

Implicit Entities

Lagrange Multipliers

Back to our problem, we minimize $f(\vec{r}) = \|\vec{r} - \vec{p}\|^2$ where $\vec{g}(\vec{r}) = 0$ by using the Lagrange method:

$$\nabla f(\vec{r}) + \vec{\lambda}^t \nabla \vec{g}(\vec{r}) = 0 \tag{18}$$

$$\vec{g}(\vec{r}) = 0 \tag{19}$$

Where $\nabla = [\frac{\partial}{\partial \vec{u}}]^t$. The number of equations and unknowns for the system is three for curves, four for surfaces, and five for implicitly defined space curves.

Implicit Entities

- A good starting value will allow the use of faster methods.
- No real CAD systems store curves or surfaces in implicit form.
- Parametric methods are mathematically less complicated.

Quaternions

What are Quaternions?[3]

Quaternions are used for calculations involving 3D rotations. Quaternions have four dimensions, one of which is a real dimension and the other three imaginary. Imaginary dimensions have a value of $\sqrt{-1}$, and each $\sqrt{-1}$ is perpendicular to the other two. A unit quaternion can be represented as a vector: $\vec{q}_R = [q_0, q_1, q_2, q_3]^t$. $q_0 > 1$, and $q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$.

Quaternion Rotation

A quaternion rotation matrix is given by:

$$R = \begin{pmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 - q_0 q_3) & 2(q_1 q_3 + q_0 q_2) \\ 2(q_1 q_2 + q_0 q_3) & q_0^2 + q_2^2 - q_1^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_2 q_3 + q_0 q_1) & q_0^2 + q_3^2 - q_1^2 - q_2^2 \end{pmatrix}$$

Finding the Optimal Rotation

- Our goal is to find the optimal rotation that would match a set of measured data points $P = \{\vec{p}_i\}$ to a set of model points $X = \{\vec{x}_i\}$.
- $N_x = N_p$, and each point $\vec{p}_i$ corresponds to the point $\vec{x}_i$ with the same index.
- The translation vector is $\vec{q}_T = [q_4, q_5, q_6]^t$.
- The translation vector can be appended to the unit vector to obtain the complete registration state vector $\vec{q} = [q_1, q_2, q_3, q_4, q_5, q_6]^t$.

Finding the Optimal Rotation

- The optimum rotation will give the smallest mean squared error between the model points and the translated measured data points: $f(\vec{q}) = \frac{1}{N_p} \sum_{i=1}^{N_p} \|\vec{x}_i - R(\vec{q}_R)\vec{p}_i - \vec{q}_T\|^2$
- To measure the similarity of sets P and X, we can find their cross-covariance: $\frac{1}{N_p} \sum_{i=1}^{N_p} [\vec{p}_i, \vec{x}_i^t] - \vec{\mu}_p \vec{\mu}_x^t$, where $\vec{\mu}_p = \frac{1}{N_p} \sum_{i=1}^{N_p} \vec{p}_i$ and $\vec{\mu}_x = \frac{1}{N_p} \sum_{i=1}^{N_x} \vec{x}_i$ are the centers of mass for $\vec{\mu}_p$ and $\vec{\mu}_x$
- The cross-covariance can be put into the form of a matrix, and the maximum eigenvalue is selected as the optimal rotation. The optimal translation vector is given by: $\vec{q}_T = \vec{mu}_z - R(\vec{q}_R)\vec{\mu}_p$
- The least-squares quaternion operation is: $(\vec{q}, d_{ms}) = Q(P, X)$, where d_{ms} is the mean square point matching error.

Finding the Optimal Rotation

The Q matrix in the previous slide is found using quaternion math and the derivation was not proided in the paper.

$$Q(\sum_{px}) = \begin{pmatrix} tr(\sum_{px}) & \Delta^T \\ \Delta & \sum_{px} + \sum_{px}^T - tr(\sum_{px} I_3) \end{pmatrix}$$

Where I_3 is the $3x3$ identity matrix, $\Delta = [A_{23}, A_{31}, A_{12}]$ and $A_{ij} = (\sum_{px} - \sum_{px}^T)_{ij}$.

Introduction

- An ICP algorithm moves (registers, positions) a data shape P to align with a model shape X.
- The data shape must be decomposed into a set of points (if P is a line or triangle set, just use the endpoints or vertices; if P is a curve or surface, approximate it using lines or triangles, and use the endpoints and vertices of the approximations).
- The number of points in the data shape is denoted N_p.
- The number of points in the model shape is denoted N_x.

Distance Between Data Point and Model Shape

Distance d

The distance d between an individual data point $\vec{p}$ and model shape X is denoted:

$$d(\vec{p}, X) = \min_{\vec{x} \in X} \|\vec{x} - \vec{p}\| \tag{20}$$

Closest Point $\vec{y}$

The closest point in X that yeilds the minimum distance is denoted $\vec{y}$ such that $d(\vec{p}, y) = d(\vec{p}, X)$, where $\vec{y} \in X$. Computing the closest point for one point in P has a worst case cost of $O(N_x)$ and an expected cost of $O(\log(N_x))$. For all the points in P, the worst case cost is $O(N_x N_p)$, and the expected cost is $O(N_p \log(N_x))$

Distance Between Data Point and Model Shape

Closest Point Operator

Let Y be the resulting set of closest points, and C be the closest point operator.

$$Y = C(P, X) \tag{21}$$

Given the resulting set Y, which is the correspondence of P and X, the least squares registration is computed by:

$$(\vec{q}, d) = Q(P, Y) \tag{22}$$

Applying the registration, $P = \vec{q}(P)$.

ICP Algorithm Statement

- A point set P with N_p points $\{\vec{p}_i\}$ from the data shape and the model shape X with Nx points, lines or triangles are given.
- Iteration is initialized with $P_0 = P$, $\vec{q}_0 = [1, 0, 0, 0, 0, 0, 0]^t$, and $k = 0$
- 1. Compute the closest points: $Y_k = C(P_k, X)$, cost: $O(N_p \log(N_x))$ average, $O(N_p N_x)$ worse case.
- 2. Compute the registration: $(\vec{q}_k, d_k) = Q(P_0, Y_k)$, cost: $O(N_p)$
- 3. Apply the registration: $P_{k+1} = \vec{q}_k(P_0)$, cost: $O(N_p)$
- 4. Terminate the iteration when the change in mean-square error falls below a threshold $\tau > 0$; $d_k - d_{k+1} < \tau$
- Steps 1-4 are repeated until convergence is within the toloerance.

Objects That Won't Work

- Proposed registration algorithm will have problems registering "sea urchin" type shapes due to limited number of initial rotation states.
- Small features relative to object size.
- ICP is not useful if the set of data points contain many points that don't correspond with any model points.

Point Set Matching

Matching a set of 8 points against a set of 11 points, using an ICP algorithm with six iterations. The ICP algorithm took less than 1s. One initial rotation and translation state.

Pre-Registration

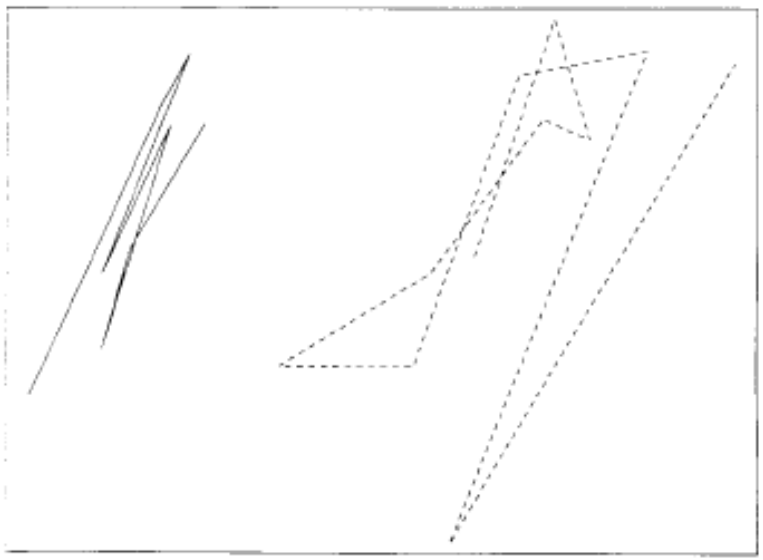


Fig. 4. Set 1 and set 2 prior to registration.

Post-Registration

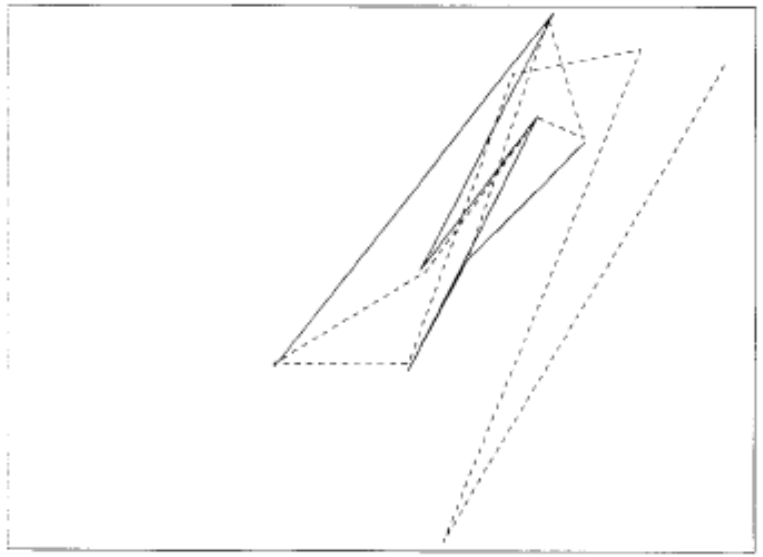


Fig. 5. Set 1 and set 2 after ICP registration.

Curve Matching

A copy of a 3D parametric spline was rotated to be difficult to match the reference. The rotated and translated curve used a polyline description with 64 points. Gaussian noise is then applied to each point of the polyline. 12 initial rotation states and 6 initial translation states = 72 initial registration states were used. The correct solution was found in 6 min.

Curve Matching

Pre-Registration

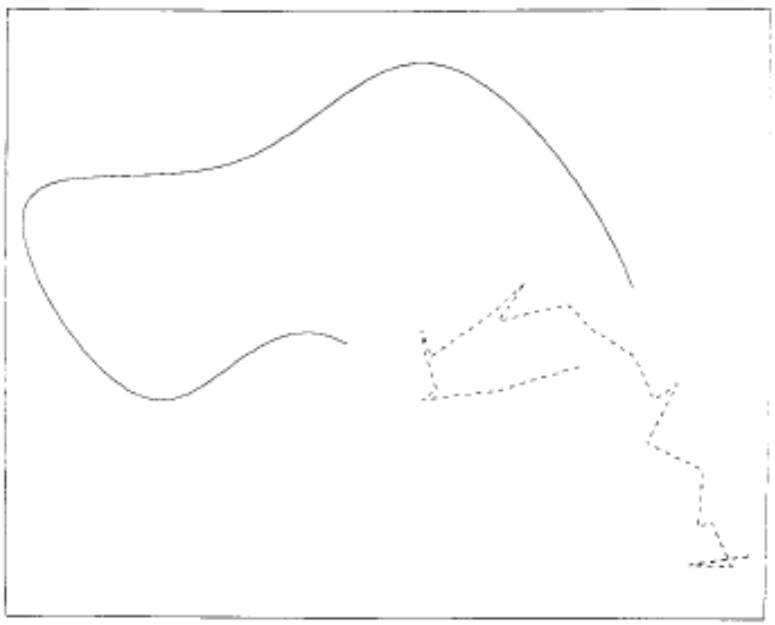


Fig. 6. Ideal and a partial noisy space curve before registration.

Post-Registration

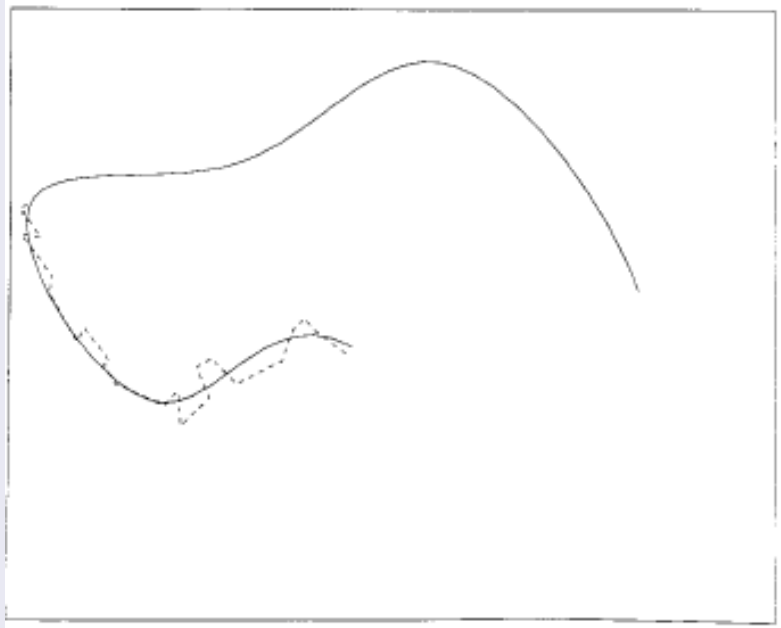


Fig. 7. Ideal and a partial noisy space curve after registration.

Surface Matching - Bezier Surface Patch - Global

250 randomly selected points on the surface patch were translated and rotated randomly.The surface is approximated with 450 triangles. Random noise was then added.

Surface Matching - Bezier Surface Patch - Global

Pre-Registration

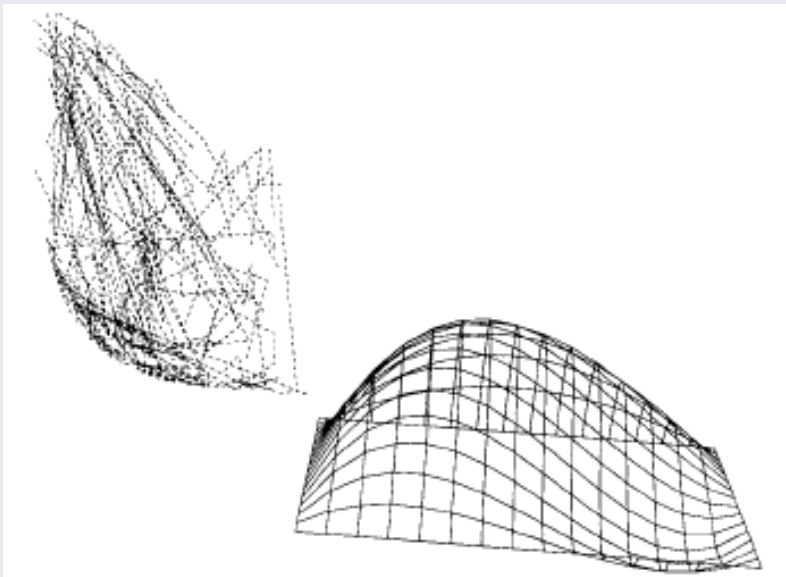


Fig. 8. Noisy point set and surface patch prior to registration: 250 points matched to 450 triangles.

Post-Registration

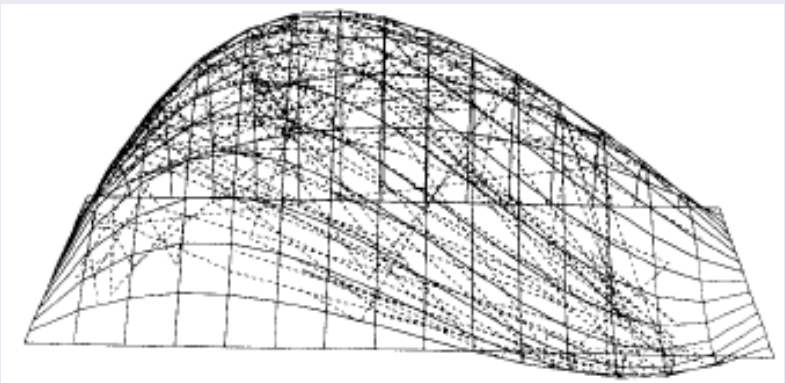


Fig. 9. Noisy point set and surface patch after registration.

Surface Matching - Bezier Surface Patch - Local

A subset of 138 noisy points was selected for local matching. 24 initial rotation states and six initial translation states were used.

Pre-Registration

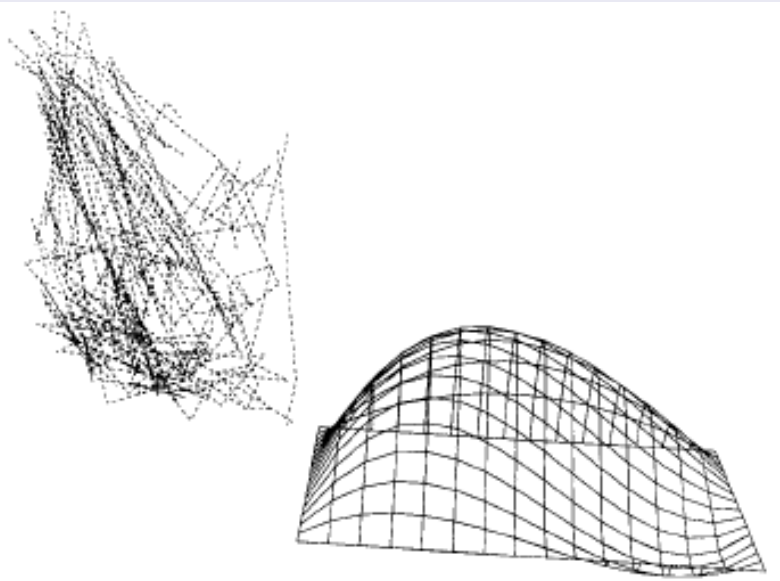


Fig. 10. Noisy point subset and surface patch prior to registration: 138 points and 450 triangles.

Post-Registration

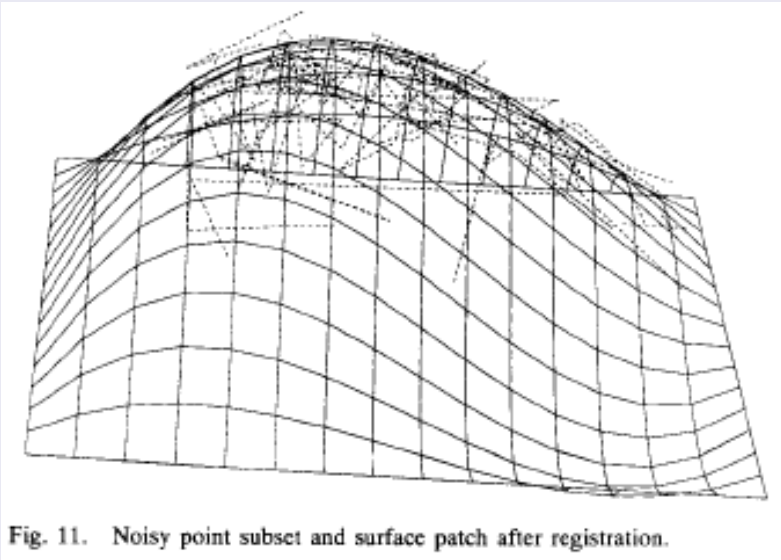


Fig. 11. Noisy point subset and surface patch after registration.

Surface Matching - Mask

The model surface was approximated with 2546 points.The correct solution was found in 10 min. with 0.59RMS error. Six iterations were used on each of the 24 intitial state vectors.

Pre-Registration

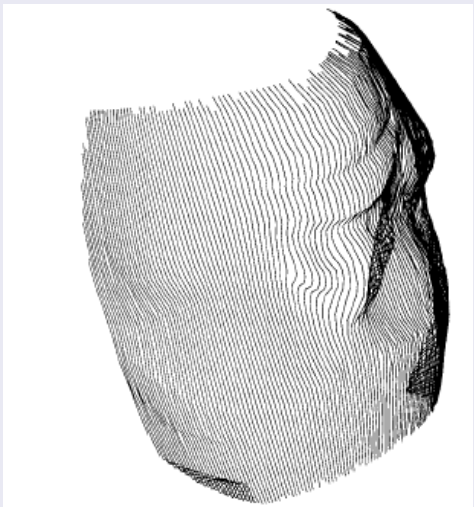


Fig. 13. Data: Rotated and translated point set of mask: 2546 points.

Post-Registration

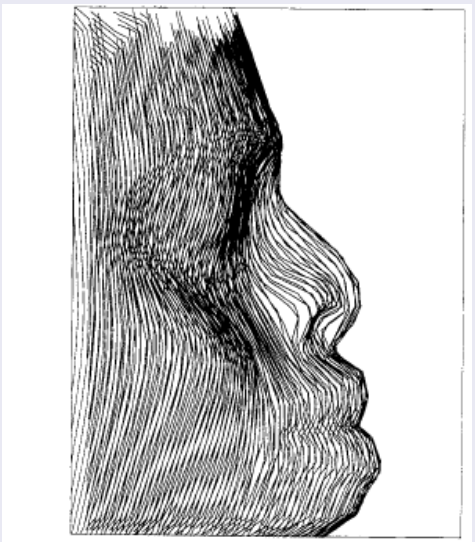


Fig. 14. Side view of range image model and registered point set.

Surface Matching - Terrain

The model surface was approximated with 13655 points. The correct solution was found in 1 hr. using 24 initial rotations and one initial translation.

Pre-Registration

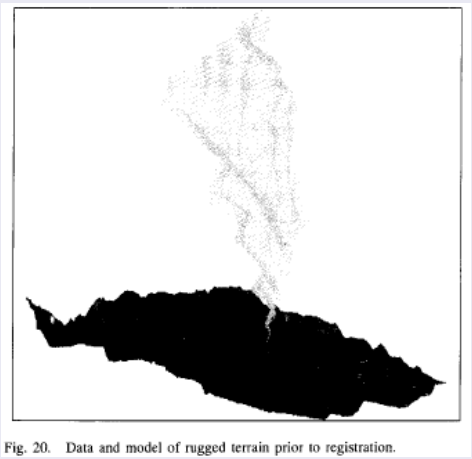


Fig. 20. Data and model of rugged terrain prior to registration.

Post-Registration

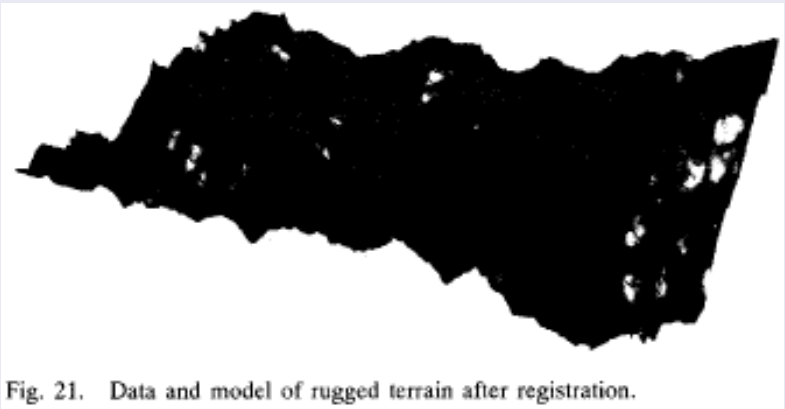


Fig. 21. Data and model of rugged terrain after registration.

Question 1: What is Depth Measurement?

Definition

A depth measurement is a method of determining how far a point is from the camera.

Depth Measurements

Depth measurements are also useful for other applications such as "assembly, gaging, robot navigation, medical diagnosis, and cartography."[4] Some techniques to obtain range images are: "radar, triangulation, moire, holographic interferemetry, focusing, and diffraction". In this paper, we ignore unequal uncertainties in different sensed points.

Bibliography


Wikipedia
Implicit Function


Wikipedia
Lagrange Multipliers


euclideanspace.com
Maths - Quaternions


Paul J. Besl
Active, Optical Range Imaging Sensors