Registration without ICP

Helmut Pottmann, Stefan Leopoldseder, Michael Hofer
Institute of Geometry, Vienna University of Technology
Wiedner Hauptstraße 8–10/113, A-1040 Wien
Email: pottmann@geometrie.tuwien.ac.at, stefan@geometrie.tuwien.ac.at,
hofer@geometrie.tuwien.ac.at

Version: 31-Jan-2002

We present a new approach to the geometric alignment of a point cloud to a surface and to related registration problems. Based on a careful geometric study of the ICP algorithm, we provide an alternative concept which relies on instantaneous kinematics and on the geometry of the squared distance function to a surface.

Key Words: registration, instantaneous kinematics, squared distance function

1. INTRODUCTION

We investigate the following registration problem. Suppose that we have a *CAD model* from which a workpiece has been produced. This workpiece has been scanned with some 3D measurement device (laser range scanning, light sectioning, ...) resulting in a *3D data point cloud* from the surface of this workpiece. Thereby, the CAD model shall describe the 'ideal' shape of the object and will be available in a coordinate system that is different to that of the 3D data point set. For the goal of shape inspection it is of interest to find the optimal Euclidean motion (translation and rotation) that aligns, or registers, the point cloud to the CAD model. This makes it possible to check the given workpiece for manufacturing errors and to visualize and classify the deviations.

A well-known standard algorithm to solve such a registration problem is the iterative closest point (ICP) algorithm of Besl and McKay [1], which we will briefly summarize in Sec. 1.1. For an overview of the recent literature on this topic we refer to [5, 6, 12]. ICP is an iterative algorithm which in each step applies a motion to the current position of the point cloud. The motion is such that the points move in a least squares sense as close as possible to their closest points on the model shape. In Sec. 2, we will study those cases where ICP works in one single step and see that these are very rare. In Sec. 3, we investigate the squared distance function d^2 to a surface and see that ICP actually works with local quadratic approximants to d^2 which are very good for points far away from the surface, but not good at all for points close to the surface. In Sec. 4, we review basic facts from instantaneous kinematics. Our alternative approach to the registration problem, which is based on instantaneous kinematics and local quadratic approximants to d^2, is presented in Sec. 5. The new method shows a faster convergence behavior and is also applicable for other types of registration and positioning problems. Finally, in Sec. 6 we outline the many directions for future research which are opened by the present concept.

1.1. The ICP Algorithm

The *iterative closest point (ICP) algorithm* has been introduced by Chen and Medioni [4] and Besl and McKay [1]. An excellent summary with new results on the acceleration of the ICP algorithm has been given by Rusinkiewicz and Levoy [12], who also suggest that *iterative corresponding point* is a better expansion for the abbreviation ICP than the original *iterative closest point*.

The point set ('data' shape) is rigidly moved (registered, positioned) to be in best alignment with the CAD model ('model' shape). This is done iteratively: In the first step of each iteration, for every data point the closest point on the surface ('normal footpoint') of the CAD model is computed. This is the most time consuming part of the algorithm and has to be implemented efficiently. As a result of this first step one obtains a point sequence $Y = (\mathbf{y}_1, \mathbf{y}_2, \ldots)$ of closest model shape points to the data point sequence $X = (\mathbf{x}_1, \mathbf{x}_2, \ldots)$. Each point $\mathbf{x}_i$ corresponds to the point $\mathbf{y}_i$ with the same index.

In the second step of each iteration the rigid motion m is computed such that the moved data points $m(\mathbf{x}_i)$ are closest to their corresponding points $\mathbf{y}_i$, where the objective function to be minimized is

$$F = \sum_{i=1}^{N} \|m(\mathbf{x}_i) - \mathbf{y}_i\|^2. \tag{1}$$

This least squares problem can be solved explicitly, see e.g. [1, 8]. The translational part of m brings the center of mass of X to the center of mass of Y. The rotational part of m can be obtained as the unit eigenvector that corresponds to the maximum eigenvalue of a symmetric 4×4 matrix. The solution eigenvector is nothing but the unit quaternion description of the rotational part of m.

After this second step the positions of the data points are updated via $X_{\text{new}} = m(X_{\text{old}})$. Now step 1 and step 2 are repeated, always using the updated data points, until the change in the mean-square error falls below a preset threshold. Since the value of the objective function decreases both in step 1 and 2, the ICP algorithm always converges monotonically to a local minimum.

2. WHERE CAN ICP WORK IN ONE ITERATION?

For a better understanding of the ICP approach it seems to be important to ask the question whether it is possible that the algorithm returns the correct solution in a single step. So we ask whether there exists a surface Φ and a dense point cloud $\{\mathbf{x}_i,\ i = 1, \ldots, N\}$, such that the transformation of this cloud to the cloud of closest points $\{\mathbf{y}_i,\ i = 1, \ldots, N\}$ on Φ is a rigid body motion. We assume a smooth surface and so the closest points are normal footpoints on Φ.

Let the surface be given in parametric form $\mathbf{x}(u, v)$. We apply a displacement to it, which can always be seen as a helical motion. In an adapted coordinate system a helical motion has the form

$$x' = x\cos\phi - y\sin\phi,\ \ y' = x\sin\phi + y\cos\phi,\ \ z' = z + h\phi. \tag{2}$$

The case of a pure translation and pure rotation are included with $\phi = 0$ and $h = 0$, respectively. The displacement (2) maps the surface $(x(u, v), y(u, v), z(u, v))$ to a surface $(x'(u, v), y'(u, v), z'(u, v))$, which we use instead of the dense point cloud. We now have to ask whether it is possible that all displacement vectors $\mathbf{x}' - \mathbf{x}$

are orthogonal to Φ. Thus, $\mathbf{x}' - \mathbf{x}$ has to be orthogonal to the tangent vector $\mathbf{x}_u = \partial\mathbf{x}/\partial u$,

$$(\mathbf{x}' - \mathbf{x}) \cdot \mathbf{x}_u = (\cos\phi - 1)(xx_u + yy_u) + \sin\phi(xy_u - x_u y) + h\phi z_u = 0, \qquad (3)$$

and it has to be orthogonal to the tangent vector $\mathbf{x}_v = \partial\mathbf{x}/\partial v$,

$$(\mathbf{x}' - \mathbf{x}) \cdot \mathbf{x}_v = (\cos\phi - 1)(xx_v + yy_v) + \sin\phi(xy_v - x_v y) + h\phi z_v = 0. \qquad (4)$$

These equations have to be satisfied for all u and v and thus we may differentiate them. Differentiating (3) with respect to v and (4) with respect to u, and subtracting those equations yields the necessary condition

$$(x_v y_u - x_u y_v)\sin\phi = 0. \qquad (5)$$

Depending on which factor in this product vanishes, we distinguish two cases.

Case A. If $\sin\phi = 0$, the rotational angle is 0 or π. In the first subcase, $\phi = 0$, all displacement vectors are parallel. At all its points the surface Φ must be orthogonal to this direction and hence it is a plane orthogonal to it. We have encountered the simplest special case where ICP works in a single step: *Surface Φ is a plane and the point cloud lies in a parallel position to it.* In the second subcase, $\phi = \pi$, the displacement consists of a reflection at the z-axis and, if $h \neq 0$, a translation parallel to it. Then, all displacement chords, connecting points $\mathbf{x}$ with $\mathbf{x}'$, intersect the z-axis. These lines have to be orthogonal to the surface Φ. It is well known that those surfaces whose normals meet a fixed line G are rotational surfaces with axis G, see [11]. Since the normals of a rotational surface are normals of the meridian curves in the planes through the axis G, the problem is now reduced to a planar one. We are looking for a planar curve p, such that a reflection of its points $\mathbf{p}(t)$ in G and successive translation parallel to G yields a point $\mathbf{p}'(t)$ which lies on the curve normal at $\mathbf{p}(t)$. This means that the so-called subnormal distances on G are constant for the curve p, namely equal to half of the translational distance h (Fig. 1).

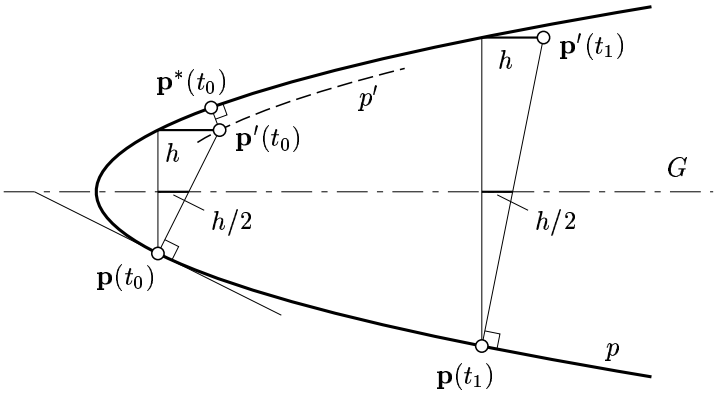


FIG. 1 Rotational surface with parabola as meridian curve.

Let us first assume $h \neq 0$. It is well-known that the curve then must be a *parabola* with axis G. Hence, the surface Φ is a *paraboloid of revolution.* However,

if we have a point cloud in this very special coaxial position with respect to Φ, ICP still would not yield the result in a single iteration. It would not match the points $\mathbf{p}'(t)$ to $\mathbf{p}(t)$, but to the closer normal footpoints $\mathbf{p}^*(t)$ on the same side of the axis G where $\mathbf{p}'(t)$ is lying (see Fig. 1). ICP in one iteration would not be sufficient for a small patch of Φ either, since in any neighborhood of $\mathbf{p}(t)$ there are surface points that are closer to $\mathbf{p}'(t)$ than $\mathbf{p}(t)$ is.

In the case $h = 0$, the normals of the profile curve p are orthogonal to G, and therefore p is a line parallel to A and thus Φ is a *right circular cylinder* with axis A. The displaced version Φ' is obtained by reflection at A and thus, as a whole, agrees with Φ. Here, the point cloud would already lie on the cylinder, and this is not a relevant case for our investigation.

Case B. Now we assume that $x_v y_u - x_u y_v = 0$. This equation expresses the fact that the third coordinate of the normal vector $\mathbf{x}_u \times \mathbf{x}_v$ is vanishing. Hence, all normals of the surface Φ are parallel to the xy-plane. Therefore Φ is a general cylinder surface with rulings parallel to the z-axis. The displacement vectors $\mathbf{x}' - \mathbf{x}$ are orthogonal to Φ, i.e., parallel to the xy-plane, which yields $h = 0$ in (2). The displacement is a pure rotation about the z-axis.

We have to figure out the cross sections of the cylinder Φ, say the one in the plane $z = 0$. The planar problem which still needs to be solved is the following: Is there a curve c such that for a rotation about the origin $\mathbf{o}$ the displacement vectors are orthogonal to c?

For a rotation, see Fig. 2(a), the displacement vectors $\mathbf{c}'(t) - \mathbf{c}(t)$ form a constant angle α to the radial lines through the center $\mathbf{o}$ of rotation. Since the displacement

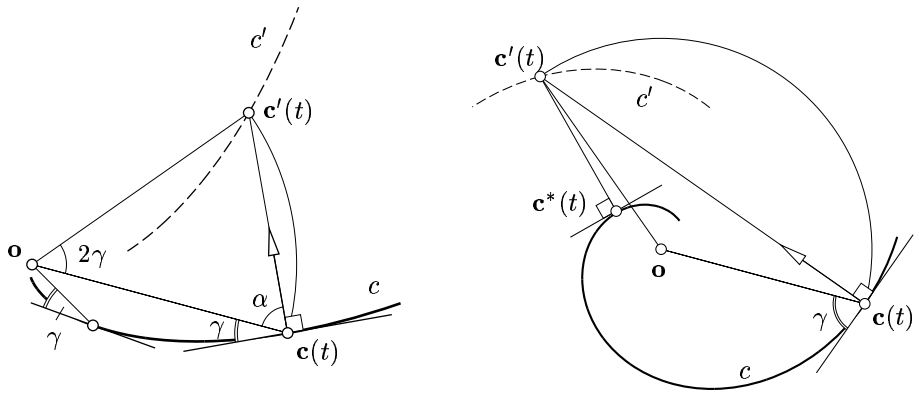


FIG. 2 Logarithmic spiral c for $\gamma = 25°$ (a), and $\gamma = 70°$ (b).

vectors also have to be orthogonal to c, the curve c has to intersect the radial pencil of lines under fixed angle $\gamma = \pi/2 - \alpha$. If this angle is $\pi/2$, we have a circle. It is also well known that otherwise we get a *logarithmic spiral* c with polar equation

$$r(u) = ae^{pu}, \quad p = \cot\gamma. \tag{6}$$

In case of a circle as base curve, the surface Φ is a right circular cylinder, which we have already realized as not relevant for our investigation. Thus, the only nontrivial situation in case B is that of a cylinder with a logarithmic spiral as basis and a

displaced point cloud in a very special position: If the cylinder Φ has a spiral basis with parameter p, the point cloud must be rotated about the 'spiral eye ruling' of Φ by an angle 2γ with $p = \cot\gamma$. Note, that for $\gamma > 29.98837\ldots^\circ$ the curve point closest to $\mathbf{c}'(t)$ is no longer $\mathbf{c}(t)$ but a second normal footpoint $\mathbf{c}^*(t)$, see Fig. 2(b). We summarize our results as follows.

THEOREM 1. *The only surfaces Φ for which a displaced copy Φ' exists such that the mapping of all points $\mathbf{x}' \in \Phi'$ to the closest points $\mathbf{x} \in \Phi$ is a rigid body motion, are planes and cylinders with a logarithmic spiral as basis.*

But even in these two cases the point cloud needs to be in a very special position such that ICP works in one iteration step. Thus, we will avoid matching to the closest points. In fact we will show that it is not necessary at all to work with corresponding points.

3. LOCAL QUADRATIC APPROXIMANTS OF THE SQUARED DISTANCE FUNCTION TO CURVES AND SURFACES

The algorithm we are proposing heavily relies on local quadratic approximants to the squared distance function of the surface Φ to which the point cloud should be registered. For a derivation and proofs of the following results we refer the reader to [10]. For a better understanding, we first present local quadratic approximants to planar curves and then generalize the obtained results to surfaces and space curves.

3.1. Local Quadratic Approximants of the Squared Distance Function to a Planar Curve

In Euclidean 3-space $\mathbb{R}^3$, we consider a planar C^2 curve $\mathbf{c}(t)$ with parameterization $(c_1(t), c_2(t), 0)$. The Frenet frame at a curve point $\mathbf{c}(t)$ consists of the unit tangent vector $\mathbf{e}_1 = \dot{\mathbf{c}}/\|\dot{\mathbf{c}}\|$ and the normal vector $\mathbf{e}_2(t)$. The two vectors form a right-handed Cartesian system in the plane. With $\mathbf{e}_3 = \mathbf{e}_1 \times \mathbf{e}_2 = (0, 0, 1)$ this system is extended to a Cartesian system Σ in $\mathbb{R}^3$. Coordinates with respect to Σ are denoted by (x_1, x_2, x_3). The system Σ depends on t and shall have the curve point $\mathbf{c}(t)$ as origin. At least locally, the shortest distance of a point $\mathbf{p} = (0, d, 0)$ on the x_2-axis (curve normal) is its x_2-coordinate d. For each t, locally the graph points $(0, d, d^2)$ of the squared distance function form a parabola p in the normal plane of $\mathbf{c}(t)$ (see Fig. 3). This parabola can be considered fixed in Σ. Varying t, the positions of the parabola in the original system generate a *moulding surface* Ψ (cf. [10]) with parameterization

$$\mathbf{x}(t, d) = \mathbf{c}(t) + d\mathbf{e}_2(t) + d^2\mathbf{e}_3. \tag{7}$$

The generated surface Ψ is in general not the graph surface Γ of the squared distance function to the given planar curve $\mathbf{c}(t)$, but Γ is contained in Ψ. If a vertical line through $(x, y, 0)$ intersects Ψ in several points, the one with the smallest z-coordinate lies on the graph surface Γ.

When we now study local quadratic (Taylor) approximants of Γ, we do not consider the global effects of Ψ. We give formulae for local approximants which work on the local distance function. This means that in determining d for neighboring points of $\mathbf{p}$ we are only locally varying the footpoint of the normal to the curve $\mathbf{c}(t)$. In other words, at points of the medial axis we work with just one sheet of the surface Ψ.

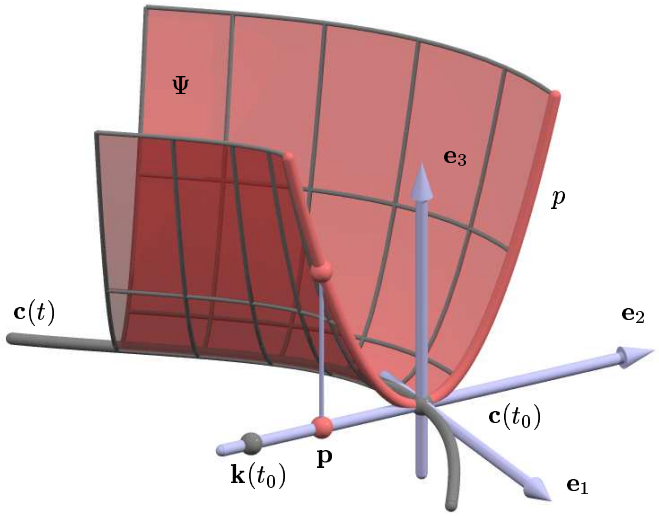


FIG. 3 Planar curve $\mathbf{c}(t)$ with Frenet frame $\mathbf{e}_1$, $\mathbf{e}_2$. The graph of the squared distance function d^2 to this curve is part of the moulding surface Ψ generated by the parabola p.

Consider a point $\mathbf{p}$ in π whose coordinates in the Frenet frame at the normal footpoint $\mathbf{c}(t_0)$ are $(0, d)$. The curvature center $\mathbf{k}(t_0)$ at $\mathbf{c}(t_0)$ has coordinates $(0, \rho)$. Here, ρ is the inverse curvature $1/\kappa$ and thus has the same sign as the curvature, which depends on the orientation of the curve.

PROPOSITION 1. *In the Frenet frame, the second order Taylor approximant F_d of the squared distance function d^2 at $(0, d)$ is given by*

$$F_d(x_1, x_2) = \frac{d}{d-\rho} x_1^2 + x_2^2. \tag{8}$$

For a derivation of this result and a discussion of the different types of the graph surface Γ_d of F_d we refer the reader to [10].

3.2. Local Quadratic Approximants of the Squared Distance Function to a Surface

Consider an oriented surface $\mathbf{s}(u, v)$ with a unit normal vector field $\mathbf{n}(u, v) = \mathbf{e}_3(u, v)$. At each point $\mathbf{s}(u, v)$, we have a local right-handed Cartesian system whose first two vectors $\mathbf{e}_1, \mathbf{e}_2$ are determined by the principal curvature directions. The latter are not uniquely determined at an umbilical point. There, we can take any two orthogonal tangent vectors $\mathbf{e}_1, \mathbf{e}_2$. We will refer to the thereby defined frame as *principal frame* $\Sigma(u, v)$. Let κ_i be the (signed) principal curvature to the principal curvature direction $\mathbf{e}_i$, $i = 1, 2$, and let $\rho_i = 1/\kappa_i$. Then, the two principal curvature centers at the considered surface point $\mathbf{s}(u, v)$ are expressed in Σ as $\mathbf{k}_i = (0, 0, \rho_i)$. The quadratic approximant F_d to the squared distance function d^2 at $\mathbf{p} = (0, 0, d)$ is the following.

PROPOSITION 2. *The second order Taylor approximant of the squared distance function to a surface at a point $\mathbf{p}$ is expressed in the principal frame at the normal*

footpoint via

$$F_d(x_1, x_2, x_3) = \frac{d}{d-\rho_1} x_1^2 + \frac{d}{d-\rho_2} x_2^2 + x_3^2. \tag{9}$$

Let us look at two important special cases.

- For $d = 0$ we obtain

$$F_0(x_1, x_2, x_3) = x_3^2. \tag{10}$$

  This means that the second order approximant to d^2 at a surface point $\mathbf{p}$ is the same for the surface Φ and for its *tangent plane* at $\mathbf{p}$. Thus, if we are close to the surface, the squared distance function to the tangent plane at the closest point to the surface is a very good approximant. At least at first sight it is surprising that the tangent plane, which is just a first order approximant, yields a second order approximant when we are considering the squared distance function d^2, to surface and tangent plane, respectively.

- For $d = \infty$ we obtain

$$F_\infty(x_1, x_2, x_3) = x_1^2 + x_2^2 + x_3^2. \tag{11}$$

  This is the squared distance to the footpoint on the surface.

We see that distances to normal footpoints, which are used in ICP, are just good if we are in a greater distance to the surface Φ. In the vicinity of the surface, it is much better to use other local quadratic approximants. The simplest one is the squared distance to the tangent plane at the normal footpoint. Registration typically starts with a rough guess of the correct position obtained for example via principal component analysis, matching special surface features or taking into account some preknowledge on surface and point cloud. Hence, for optimal alignment we typically need several iteration steps in the vicinity of the surface. This is the reason why we are not minimizing distances to the normal footpoints.

3.3. Local Quadratic Approximants of the Squared Distance Function to a Space Curve

In case that boundary curves of surfaces are involved, it is also useful to know about local quadratic approximants of the squared distance function d^2 to a space curve. Given a point $\mathbf{p}$ in $\mathbb{R}^3$, the shortest distance to a C^2 space curve $\mathbf{c}(t)$ occurs along a normal of the curve or at a boundary point of it. The latter case is trivial and thus we exclude it. At the normal footpoint $\mathbf{c}(t_0)$ we form a Cartesian system with $\mathbf{e}_1$ as tangent vector and $\mathbf{e}_3$ in direction of the vector $\mathbf{p} - \mathbf{c}(t_0)$. This *canonical frame* can be viewed as limit case of the principal frame for surfaces, when interpreting the curve as a pipe surface with vanishing radius. By this limit process, we can also show the following result.

PROPOSITION 3. *The second order Taylor approximant of the squared distance function to a space curve* $\mathbf{c}(t)$ *at a point* $\mathbf{p}$ *is expressed in the canonical frame* Σ *at the normal footpoint via*

$$F_d(x_1, x_2, x_3) = \frac{d}{d-\rho_1} x_1^2 + x_2^2 + x_3^2. \tag{12}$$

Here, $(0, 0, \rho_1)$ *are the coordinates (in* Σ*) of the intersection point of the curvature axis of* $\mathbf{c}(t)$ *at the footpoint* $\mathbf{c}(t_0)$ *with the perpendicular line* $\mathbf{pc}(t_0)$ *from* $\mathbf{p}$ *to* $\mathbf{c}(t)$.

4. INSTANTANEOUS KINEMATICS

For the algorithm we propose, some knowledge about kinematics is essential. Thus, in this section we briefly outline the basic facts we are using later on. Consider a differentiable one-parameter rigid body motion in Euclidean 3-space. Introducing Cartesian coordinate systems in the moving system Σ and in the fixed system Σ_0, the time dependent position $\mathbf{x}_0(t)$ of a point $\mathbf{x} \in \Sigma$ in the fixed system is given by

$$\mathbf{x}_0(t) = \mathbf{a}(t) + M(t)\mathbf{x}. \tag{13}$$

Here, the time dependent orthogonal matrix $M(t)$ represents the spherical component of the motion, and $\mathbf{a}(t)$ describes the trajectory of the origin of the moving system. All arising functions shall be C^1. By differentiation we get the velocity vectors. It is well-known that the velocity vector field is linear at any time instant. More precisely, at any time instant there exist vectors $\mathbf{c}, \bar{\mathbf{c}}$ such that the velocity vector $\mathbf{v}(\mathbf{x})$ of any point $\mathbf{x}$ of the moving body can be computed as

$$\mathbf{v}(\mathbf{x}) = \bar{\mathbf{c}} + \mathbf{c} \times \mathbf{x}. \tag{14}$$

Note that in this formula all arising vectors are represented in the same system; this may be the moving or the fixed system. The meaning of $\mathbf{c}, \bar{\mathbf{c}}$ is as follows: $\bar{\mathbf{c}}$ represents the velocity vector of the origin, and $\mathbf{c}$ is the so-called Darboux vector (vector of angular velocity).

It is well-known that only very special one-parameter motions have a constant, i.e., time-independent velocity vector field. These motions are

- A *translation* with constant velocity (if $\mathbf{c} = \mathbf{0}$)
- A uniform *rotation* about an axis (if $\mathbf{c} \cdot \bar{\mathbf{c}} = 0$)
- A uniform *helical motion* (if $\mathbf{c} \cdot \bar{\mathbf{c}} \neq 0$)

Thus, up to the first differentiation order, any motion agrees locally with one of these motions. The most general case is that of a uniform helical motion, which is the superposition of a rotation with constant angular velocity about an axis G and a translation with constant velocity parallel to G. If the moving body rotates about an angle α, the translation distance is $p \cdot \alpha$. The constant factor p is referred to as *pitch* of the helical motion. For more details on helical motions and the close relations to line geometry we refer to [11]. The Plücker coordinates $(\mathbf{g}, \bar{\mathbf{g}})$ of the axis G, the pitch p and the angular velocity ω are computed from $\mathbf{c}, \bar{\mathbf{c}}$ as

$$\mathbf{g} = \frac{\mathbf{c}}{\|\mathbf{c}\|}, \qquad \bar{\mathbf{g}} = \frac{\bar{\mathbf{c}} - p\mathbf{c}}{\|\mathbf{c}\|}, \qquad p = \frac{\mathbf{c} \cdot \bar{\mathbf{c}}}{\mathbf{c}^2}, \qquad \omega = \|\mathbf{c}\|. \tag{15}$$

Recall that the Plücker coordinates of a line G consist of a direction vector $\mathbf{g}$ and the moment vector $\bar{\mathbf{g}} = \mathbf{p} \times \mathbf{g}$, where $\mathbf{p}$ represents an arbitrary point on G.

5. REGISTRATION OF A POINT CLOUD TO A CAD MODEL USING INSTANTANEOUS KINEMATICS AND QUADRATIC APPROXIMANTS OF THE SQUARED DISTANCE FUNCTION

In the ICP algorithm the data points $\mathbf{x}_i$ are moved towards their closest points $\mathbf{y}_i$ on the model surface Φ. Instead of moving $\mathbf{x}_i$ towards $\mathbf{y}_i$ we aim at bringing the

points just closer to the surface Φ. For this, we employ local quadratic approximants of the squared distance function to Φ. As we have seen in Sec. 3.2, the squared distance functions to the tangent planes of Φ approximate the squared distance function to Φ very well in the vicinity of the surface. The aim is the same as for ICP. We would like to apply a motion to the point cloud such that the sum f of squared distances to the model surface becomes minimal. Let us first give an overview of the proposed algorithm and then study the individual steps in more detail. The new algorithm iteratively applies the following steps:

1. To each point of the current position of the point cloud, compute the squared distance function F_i of the tangent plane at the point $\mathbf{y}_i \in \Phi$, which is closest to $\mathbf{x}_i$. This step is used to quadratically approximate the function f to be minimized.

2. Compute a velocity vector field, which attaches to each point a velocity vector $\mathbf{v}(\mathbf{x}_i)$ such that the quadratic function $\sum F_i(\mathbf{x}_i + \mathbf{v}(\mathbf{x}_i))$ assumes a minimal value. This step estimates a motion towards the model surface, but does not yet represent a Euclidean motion. From the point of view of optimization, we use a quadratic approximation of f and a linearization of the constraint (i.e., displacement by a rigid body motion), but we do not yet fulfil the constraint.

3. From the velocity field we compute a displacement which displaces the points $\mathbf{x}_i$ in nearly the same way as the velocity vectors (used for the minimization in the previous step) would do. In terms of optimization, this is the step where we project onto the constraint manifold.

The details to the individual steps are as follows.

Step 1. For each data point $\mathbf{x}_i \in X$ determine the nearest point $\mathbf{y}_i$ of the surface of the CAD model and determine the tangent plane there. Let $\mathbf{n}_i$ denote a unit normal vector of this tangent plane in $\mathbf{y}_i$. If $\mathbf{y}_i$ is no boundary point of the surface, $\mathbf{x}_i$ lies on the surface normal in $\mathbf{y}_i$, i.e., $\mathbf{x}_i = \mathbf{y}_i + d_i\mathbf{n}_i$ with d_i denoting the oriented Euclidean distance of $\mathbf{x}_i$ to $\mathbf{y}_i$.

In case that $\mathbf{y}_i$ is a boundary point, one will define $\mathbf{n}_i = (\mathbf{x}_i - \mathbf{y}_i)/\|\mathbf{x}_i - \mathbf{y}_i\|$, i.e., $\mathbf{n}_i$ is orthogonal to the boundary curve in $\mathbf{y}_i$, pointing in the direction of $\mathbf{x}_i$. Again we have $\mathbf{x}_i = \mathbf{y}_i + d_i\mathbf{n}_i$.

Note that depending on the application one may reject a data point $\mathbf{x}_i$ in the minimization process, if its closest surface point $\mathbf{y}_i$ lies on the boundary. This is necessary, for instance, when partial scans of the same object are registered.

Step 2. A linearization of the motion is equivalent to the use of instantaneous kinematics. The use of instantaneous kinematics for registration appears in other papers as well (see e.g. [3, 5]), maybe for the first time in [2].

The velocity vector field of an instantaneous helical motion is given by $\mathbf{v}(\mathbf{x}) = \bar{\mathbf{c}} + \mathbf{c} \times \mathbf{x}$. To each point $\mathbf{x}_i$ we attach a velocity vector $\mathbf{v}(\mathbf{x}_i) = \bar{\mathbf{c}} + \mathbf{c} \times \mathbf{x}_i$. The distance of $\mathbf{x}_i + \mathbf{v}(\mathbf{x}_i)$ to the tangent plane of the parametric surface in the point $\mathbf{y}_i$ is given by

$$d_i + \mathbf{n}_i \cdot (\bar{\mathbf{c}} + \mathbf{c} \times \mathbf{x}_i). \tag{16}$$

Now, minimization of the objective function (which is quadratic in $\mathbf{c}, \bar{\mathbf{c}}$)

$$F(C) := F(\mathbf{c}, \bar{\mathbf{c}}) = \sum_i (d_i + \mathbf{n}_i \cdot (\bar{\mathbf{c}} + \mathbf{c} \times \mathbf{x}_i))^2, \tag{17}$$

yields the pair $(\mathbf{c}, \bar{\mathbf{c}})$ that determines the helical motion whose velocity vector field we are using. The minimization can be solved using a system of linear equations. For that we rewrite (16) as

$$d_i + \mathbf{n}_i \cdot \bar{\mathbf{c}} + (\mathbf{x}_i \times \mathbf{n}_i) \cdot \mathbf{c} \;=\; d_i + (\mathbf{x}_i \times \mathbf{n}_i, \mathbf{n}_i) \begin{pmatrix} \mathbf{c} \\ \bar{\mathbf{c}} \end{pmatrix} \;=\; d_i + A_i C, \qquad (18)$$

where A_i and $C := (\mathbf{c}, \bar{\mathbf{c}})^T$ are one–by–six and six–by–one matrices respectively. We use this notation to rewrite the objective function (17) as

$$\begin{aligned} F(C) &= \sum_i (d_i + A_i C)^2 \\ &= \sum_i d_i^2 + 2 \sum_i d_i A_i C + \sum_i C^T A_i^T A_i C \\ &= D + 2B \cdot C + C^T A C \end{aligned} \qquad (19)$$

where A is a symmetric, in general positive definite six–by–six matrix, B is a column vector with six entries, and D is just some scalar.

It is well-known that the unique minimum of the quadratic function $F(C)$ solves the linear system

$$AC + B = 0. \qquad (20)$$

Remark 1. Instantaneous kinematics as described in Sec. 4 has been used in the context of reverse engineering of 'kinematic surfaces', i.e., planes, general cylinders, surfaces of revolution, and helical surfaces (cf. [9, 11]). These surfaces are characterized by the fact that there exists a vector field $\mathbf{v}(\mathbf{x}) = \mathbf{c} + \bar{\mathbf{c}} \times \mathbf{x}$ such that for each surface point $\mathbf{p}$ the vector $\mathbf{v}(\mathbf{p})$ is tangential to the surface in $\mathbf{p}$.

In the context of registration, these kinematic surfaces play a special role as well. After the registration of a point cloud to such a surface, the point cloud can still be moved tangentially to the surface without increasing the objective function $F(C)$ in Equ. (19). Thus, in the special case of kinematic surfaces the linear system (20) gets ill-conditioned. Whereas the standard ICP algorithm heavily punishes tangential movement (which slows the convergence behavior), minimizing $F(C)$ in Equ. (19) does not restrict tangential movement at all.

It is straightforward to combine the functional $F(C)$ with a functional

$$F'(C) := \sum_i (\mathbf{x}_i - \mathbf{y}_i + \bar{\mathbf{c}} + \mathbf{c} \times \bar{\mathbf{x}}_i)^2,$$

which describes the sum of squared distances of the points $\mathbf{x}_i + \mathbf{v}(\mathbf{x}_i)$ to the normal footpoints $\mathbf{y}_i$. Minimizing the quadratic functional $\bar{F}(C) = F(C) + \omega F'(C)$, where ω is a small but positive weight, again leads to the solution of a linear system.

Step 3. Moving each point $\mathbf{x}_i$ by $\mathbf{v}(\mathbf{x}_i)$, i.e., $\mathbf{x}_i \mapsto \mathbf{x}_i + \mathbf{v}(\mathbf{x}_i)$, (as we have assumed for the minimization) would not yield a Euclidean rigid body motion, but an affine one. Therefore we use the underlying helical motion determined by $(\mathbf{c}, \bar{\mathbf{c}})$ from which we can calculate axis G and pitch p with equation (15).

We apply a rotation about this axis G through an angle of $\alpha = \arctan \|\mathbf{c}\|$ and a translation parallel to G by the distance $p \cdot \alpha$ (see Fig. 4). This motion brings each point $\mathbf{x}_i$ to a position $\mathbf{x}_i'$ close to $\mathbf{x}_i + \mathbf{v}(\mathbf{x}_i)$ which has been used for the minimization in (17).

Using the underlying helical motion is furthermore justified by the fact that for $\mathbf{x}_i$ we do not know the exact corresponding point on the surface anyway, we are moving the point closer to the tangent plane, and we iterate the whole procedure to find the optimal match.

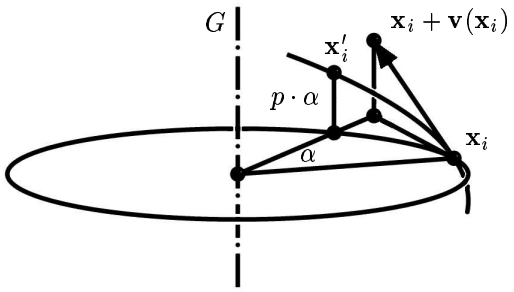
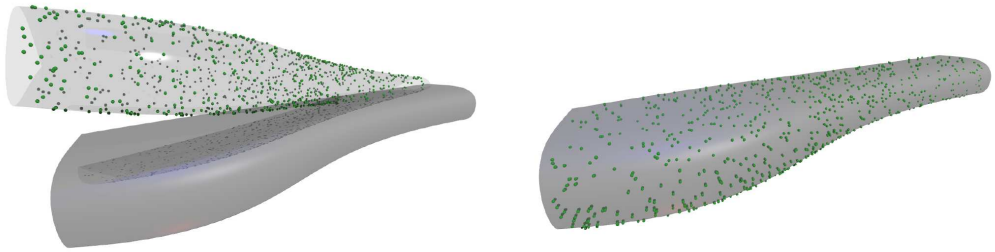


FIG. 4 New position $\mathbf{x}_i'$ of a point $\mathbf{x}_i$.

As a termination criterion for the iteration we use the change in the sum of squared distances of $\mathbf{x}_i$ to the surface. We terminate the algorithm if this value falls below a certain threshold.

EXAMPLE 1. The main application we have in mind is the quality inspection of industrial products. Here, the goal is to find the best alignment between the (exact) CAD model of a given workpiece, and a dense point cloud which has been obtained from the workpiece with a 3D scanning device.

In our example (see Fig. 5) we have generated the point data synthetically and added Gaussian noise. For reasons of visualization, a transparent surface is fitted to the point cloud in Fig. 5(a), but this surface is not used in the iterative alignment procedure. In each iteration step, a helical motion $\mathbf{x}_i \mapsto \mathbf{x}_i'$ (cf. Fig 4) is applied



(a) (b)

FIG. 5 Registration of a point cloud to a surface. Initial position (a) and final position after 4 iterations (b).

to the data points, and after only four iterations the point cloud reaches its final position (see Fig. 5(b)).

Compared to the ICP algorithm, our algorithm converges much faster. This is obvious, since we minimize the distances to the tangent planes of the surface. Therefore, tangential movement is not restricted, which is especially important in the final steps of the registration process. Several authors have pointed out this

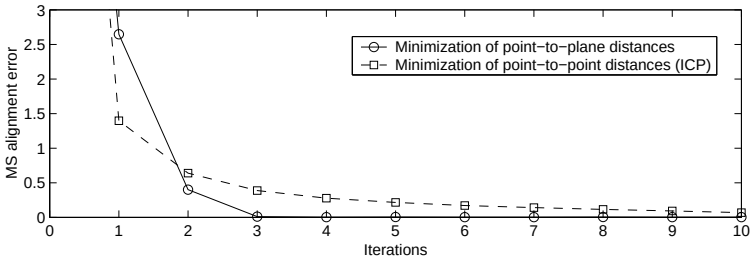


FIG. 6 Comparison of the convergence rate for minimizing point-to-plane distances vs. point-to-point distances (ICP).

fact already (see, e.g., [4, 12]). In Fig. 6 the mean squared error of the data points to the CAD surface after each iteration is given, both for our algorithm and for the standard ICP algorithm, applied to the example in Fig. 5.

Remark 2. We have described the algorithm in its simplest form. There are many ways to improve it, and actually many ideas for improvement of ICP and related registration algorithms (cf., e.g., [12, 13]) work as well. For example, we will not work in each step with all data points, but just with a random sample. Of course, more sophisticated sampling methods, e.g. by choosing data points with a good distribution of estimated normals (cf. [12]), can be applied as well. Furthermore one may reject a chosen data point, e.g., if its distance to the normal footpoint exceeds some threshold. Moreover, it is straightforward to extend the objective function (17) to a weighted scheme. There are 3D measurement devices that supply for each data point a tolerance for the occurring measurement errors. These can be included in the objective function to downweight outliers. This is especially important for a precise final alignment, but has less impact on the convergence speed of the iterative registration algorithm.

The algorithm outlined above is a special case of a more general strategy, where we use other quadratic approximants of the function f to be minimized. A natural extension is the use of the second order Taylor approximant, say F_i, to the squared distance function at the current data point position $\mathbf{x}_i$. In view of subsection 3.2, it is more complicated to compute these F_i's, but the remaining part of the algorithm is the same. In step 2, we still have to minimize a quadratic function F, and in step 3 we perform the same position correction.

Working with general Taylor approximants F_i is more subtle, however. To make sure that F is positive definite, we will use *nonnegative* quadratic approximants to d^2. One way to compute those has been presented in [10].

6. CONCLUSION AND FUTURE RESEARCH

A geometric analysis of the ICP algorithm reveals the following fact: Using closest points on the model surface as corresponding points will rarely give fast convergence. This is so since one step will almost never suffice, and the approximation of the squared distance function to the model shape with the help of squared distance functions to surface points does not work well in the vicinity of the surface. As an alternative, we provided a new framework for registration using better quadratic approximants of the squared distance function and instantaneous kine-

matics. Further work has to be done in order to satisfy the practical needs. Here is a list of some extensions.

- Instead of registration with a ridig body motion, we might allow a uniform scaling, i.e., a similarity. This is a minor change in the presented algorithm, since the velocity field is still linear and just has one more real parameter σ,

$$\mathbf{v}(\mathbf{x}) = \sigma\mathbf{x} + \bar{\mathbf{c}} + \mathbf{c} \times \mathbf{x}. \tag{21}$$

- In extension of [10], we have to investigate other quadratic approximants of the squared distance function to a surface. In particular, we need an efficient way of computing local quadratic approximants if the model shape is just given as a point cloud.

- Fast registration for industrial inspection requires the development of a spatial data structure, computed in a preprocessing step from the given model shape, such that the necessary quadratic approximants can be quickly computed or retrieved from that structure.

- It seems to be interesting to look at a hierarchical representation of the quadratic approximants of d^2, and use it efficiently in the various iteration steps of the registration procedure.

- The use of instantaneous kinematics allows us to extend the idea to the simultaneous registration of more than two geometric objects (partial scans).

ACKNOWLEDGMENTS

H. Pottmann is grateful for support by the Institute of Mathematics and Its Applications at the University of Minnesota; main ideas of the present work could be developed during a stay at IMA in spring 2001.

REFERENCES

[1] Besl, P. J., McKay, N. D. (1992), A method for registration of 3D shapes, *IEEE Trans. Pattern Anal. and Machine Intell.* **14**, 239–256.

[2] Bourdet, P., Clément, A. (1976), Controlling a complex surface with a 3 axis measuring machine, *Annals of the CIRP* **25**, 359–361.

[3] Bourdet, P., Clement, A. (1988), A study of optimal-criteria identification based on the small-displacement screw model, *Annals of the CIRP* **37**, 503–506.

[4] Chen, Y., Medioni, G. (1992), Object modelling by registration of multiple range images, *Image and Vision Computing* **10**, 145–155.

[5] Eggert, D. W. Fitzgibbon, A. W. Fisher, R. B. (1998), Simultaneous registration of multiple range views for use in reverse engineering of CAD models, *Computer Vision and Image Understanding* **69**, 253–272.

[6] Eggert, D. W., Larusso, A., Fisher, R. B. (1997), Estimating 3-D rigid body transformations: a comparison of four major algorithms, *Machine Vision and Applications* **9**, 272–290.

[7] Faugeras, O. D. Hebert, M. (1986), The representation, recognition, and locating of 3-D objects. *Int. J. Robotic Res.* **5**, 27–52.

[8] Horn, B. K. P. (1987), Closed form solution of absolute orientation using unit quaternions, *Journal of the Optical Society A* **4**, 629–642.

[9] Pottmann, H., Randrup, T. (1998), Rotational and helical surface reconstruction for reverse engineering. *Computing* **60**, 307–322.

[10] Pottmann, H., Hofer, M. (2002), Geometry of the squared distance function to curves and surfaces. Technical Report Nr. 90, Institute of Geometry, Vienna University of Technology.

[11] Pottmann, H., Wallner, J. (2001), Computational Line Geometry, Springer-Verlag Berlin Heidelberg New York.

[12] Rusinkiewicz, S., Levoy, M. (2001), Efficient variants of the ICP algorithm. in Proc. 3rd Int. Conf. on 3D Digital Imaging and Modeling, Quebec.

[13] Simon, D.A. (1996), Fast and Accurate Shape-Based Registration, Ph.D. Thesis, Carnegie Mellon University.