GLIDING SPLINE MOTIONS AND APPLICATIONS

JOHANNES WALLNER

Abstract. We consider the ICP (iterative closest point) algorithm, which may in general be used for moving 'active' elements such as curves and surfaces towards geometric objects whose distance field is computable. We show how it may be accelerated, and how it can be applied to the design of near-Euclidean near-contact spline motions. One particular application of this concept is the modeling of milling tool paths in five-axis milling. The method involves computing the distance from and footpoints in both the Euclidean motion group and the configuration space of surface-surface contact.

1. Introduction

Motion design is a topic which constantly attracts interest in the CAGD community. Instead of detailed references to the literature, we refer the interested reader to the survey article (Röschel, 1998). The main source of problems which occur in motion design is that the geometry of the set of Euclidean motions is not as simple as that of points, say, of Euclidean space. For purposes of computation, coordinates have to be introduced in that set, and experience has shown that each of the methods which have been used so far has its own deficiencies, and that the decision for a certain system of coordinates depends on the application one has in mind.

Here we embed the set of Euclidean motions in the set of affine transformations, where coordinates are found in a straight forward way. The set of affine transformations is an affine space. What makes the situation complicated is they way the Euclidean motions are embedded in that space: They occur as a surface whose dimension is one half of the dimension of the ambient space. Nevertheless, the distance from this surface may be computed. This fact makes it possible to apply the concept of *active* curves and surfaces to motion design: We produce piecewise polynomial near-Euclidean and near-gliding one-parameter and k-parameter motions.

Technical Report No. 96, Institut für Geometrie, TU Wien. To appear in *Computer-Aided Geometric Design*.

How to use near-Euclidean motions which are not Euclidean in practice is another question. Clearly we cannot expect that a rigid body undergoes an affine transformation which is not Euclidean. But that problem is easily solved: The transfer from a numerical representation to the 'real' motion, has to be modified accordingly, e.g. by using the footpoint map described by Th. 2.

2. Active elements in a distance field

2.1. **Active curves and surfaces.** In this paper motion design is based on the principle of 'active' curves and surfaces and how to move them closer to a target. More specifically, it is based on a variant of the so-called *iterative closest point* (i.e., ICP) algorithm. The word 'active' has been given to geometric entities whose shapes change during an iterative process, and especially it applies to shapes determined by control points evolving with the ICP algorithm. The general concept of 'ICP', as described in (Kass et al., 1988), (Pottmann Leopoldseder, 2002), and (Pottmann et al, 2002), is the following: Assume that r feature points $x_1, \ldots, x_r$ in the Euclidean space $\mathbb{R}^d$ are determined by control points $b_1, \ldots, b_k$, and that this dependence is *affine* in each argument. A prominent example is a spline curve

$$b(t) = \sum_{i=1}^{k} N_i(t) b_i \tag{1}$$

defined by the B-spline basis functions $N_i(t)$ and the control points

$$b_1, \ldots, b_k. \tag{2}$$

We choose $u_1, \ldots, u_r \in \mathbb{R}$ and let

$$x_i(b_1, \ldots, b_k) := b(u_i). \tag{3}$$

Further, we assume that a subset $T \subseteq \mathbb{R}^d$ (the *target*) allows computation of the distance from T and the footpoint $F_T(x) \in T$ of a point $x \in \mathbb{R}^d$.

$$F_T : \mathbb{R}^d \to T, \quad \operatorname{dist}(x, T) = \|x - F_T(x)\|. \tag{4}$$

Then the ICP algorithm is given by the recursion algorithm No. 1. The purpose of the algorithm is to bring the curve near the target.

Numerical evidence has shown that the ICP algorithm can be accelerated by replacing the distance to the footpoints by better approximants of the target's distance function. (Pottmann and Leopoldseder, 2002) proposed to use certain nonnegative quadratic approximants $\widetilde{\operatorname{dist}}_p^2$ of the function $\operatorname{dist}(\cdot, T)^2$, which are derived from the second order Taylor

The ICP algorithm:
input: $b_1, \ldots, b_k$
repeat
  for $i = 1, \ldots, r$
    evaluate feature points $x_i^0 = x_i(b_1, \ldots, b_k)$
    compute footpoints $y_i = F_T(x_i^0)$
  choose control points $b_1, \ldots, b_k$
    such that $w := \sum_{i=1}^r \|x_i(b_1, \ldots, b_k) - y_i\|^2 \to \min$
until w small enough.
result: current values of $b_1, \ldots, b_k$

ALGORITHM 1

A refined ICP algorithm:
input: $b_1, \ldots, b_k$
repeat
  for $i = 1, \ldots, r$
    evaluate feature points $x_i^0 = x_i(b_1, \ldots, b_k)$
    compute footpoints $y_i = F_T(x_i^0)$
    determine functions $\operatorname{dist}_{x_i^0}$
  choose control points $b_1, \ldots, b_k$
    such that $w := \sum_{i=1}^r \widetilde{\operatorname{dist}}_{x_i^0}(x_i(b_1, \ldots, b_k))^2 \to \min$
until w small enough.
result: current values of $b_1, \ldots, b_k$

ALGORITHM 2

polynomial of that function, and which are the topic of Sec. 2.3. This leads to Algorithm 2. The approximants $\widetilde{\operatorname{dist}}_p^2$ have the property that

$$\widetilde{\operatorname{dist}}_p(x)^2 = \operatorname{dist}(x, T)^2 \quad \text{if} \quad x \in [p, F_T(p)], \tag{5}$$

i.e., they agree with $\operatorname{dist}(\cdot, T)^2$ along the entire line segment spanned by p and its footpoint $F_T(p)$. So does the distance to the footpoint, but it turned out that the behaviour of the approximant outside the line segment $[p, F_T(p)]$ has an influence on the convergence of the algorithm.

Remark: It is not necessary that the coefficients which control the feature points are arranged in the form of coefficients of control points. The reason why the algorithms have been presented with control *points* instead of control *coefficients* is that then they perhaps look more familiar.

Remark: The ICP algorithm tries to model curves as string with limited elasticity (by the finite dimensionality of the underlying spline space) which is attracted by the target and finally rests as close as possible to it. The behaviour of the algorithm in the presence of disconnected or complicately shaped targets is similar to the behaviour of its physical analogue (e.g., the resulting curve will not follow the target's boundary if it has holes). Another familiar phenomenon which has an analogue in the real world is 'folding' of the result. The latter can be avoided to some extent by adding a bending energy term to the functional begin minimized.

2.2. **Taylor expansion of the squared distance from a surface.**

2.2.1. *Principal curvatures with respect to a normal vector.* If M is a smooth m-surface in $\mathbb{R}^d$, parametrized by a smooth $\mathbb{R}^d$-valued function $g(u_1, \ldots, u_m)$, we consider the basis vector fields $\partial_j g$ and their scalar products $g_{ij} = \langle \partial_i g, \partial_j g \rangle$. The tangent vector space of M at $p = g(u_1, \ldots, u_m)$ is spanned by $\partial_1 g, \ldots, \partial_m g$ and is denoted by $T_p M$. Its orthogonal complement is the normal space $\perp_p M$. If n is a unit normal vector attached to the point $p = g(u_1, \ldots, u_m)$, we consider

$$h^n_{ij} = \langle n, \partial_i \partial_j g \rangle \quad (i, j = 1, \ldots, m). \tag{6}$$

Any eigenvector $(\lambda_1, \ldots, \lambda_m)$ of the matrix $(g_{ij})^{-1} \cdot (h^n_{ij})$ defines a principal curvature vector

$$v = \sum_{j=1}^{m} \lambda_j \partial_j g. \tag{7}$$

It is well known that for all $p \in M$ and $n \in \perp_p M$ there is an orthonormal basis $e_1, \ldots, e_d$ such that

$$\begin{array}{ll} e_1, \ldots, e_m & \text{are curvature vectors w.r.t. } n \text{ and span } T_p M \\ e_{m+1}, \ldots, e_d & \text{span } \perp_p M, \text{ and } e_d = n. \end{array} \tag{8}$$

The eigenvalues corresponding to $e_1, \ldots, e_m$ are denoted by $\kappa^n_1, \ldots, \kappa^n_m$. They are the princpal curvatures at p with respect to n.

2.2.2. *Taylor expansion of the squared distance.* Here we use Cartesian coordinates defined by the coordinate system $(p; e_1, \ldots, e_d)$, whose origin is p (cf. Equ. (8)). The quadratic Taylor expansion of $\mathrm{dist}^2(x, M)$ at the point

$$(0, \ldots, 0, \delta) \tag{9}$$

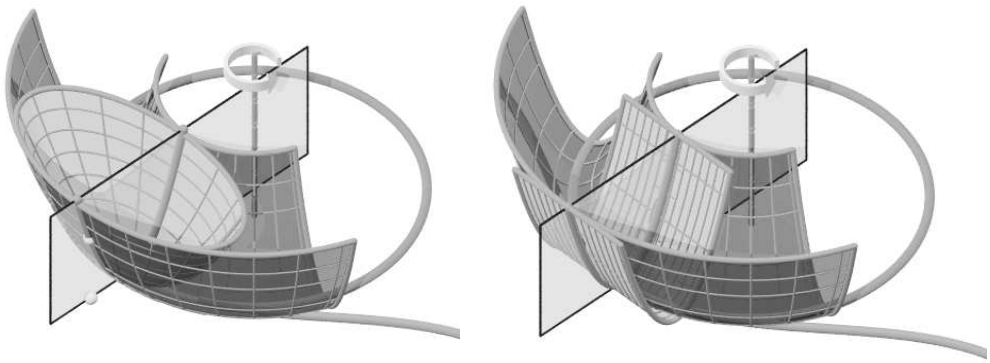

FIGURE 1. Graph of the squared distance from the osculating circle of a planar curve, and Taylor approximants. Left: Taylor approximant in a point outside the osculating circle. Right: Taylor approximant in a point of the curve (courtesy M. Hofer)

is given by the quadratic function

$$\sum_{i\le m} \frac{\delta}{\delta - 1/\kappa_i^n} x_i^2 + \sum_{i>m} x_i^2, \tag{10}$$

if the line segment $[(0,\dots,0),(0,\dots,0,\delta)]$ does not contain any of the points $(0,\dots,0,1/\kappa_i^n)$.

A proof of Equ. (10) and related results can be found in (Ambrosio and Soner, 1996) and (Ambrosio and Mantegazza, 1998). A more elementary introduction into that topic is found in (Pottmann and Hofer, 2002). Fig. 1 shows graphs of two such Taylor expansions: M is a planar curve, the surface of revolution shown is the graph of the squared distance from one of its osculating circles, and the other surfaces are graphs of Taylor approximations in various points.

Remark: When computing approximantes which take into account first and second derivatives, we can replace the distance from the curve by the distance from its osculating circle. As the distance function of the circle is much simpler, it is shown by Fig. 1 instead of the distance function of the original curve.

2.2.3. *Low-dimensional special cases.* In the case that M is a 2-dimensional surface in $\mathbb{R}^3$, every surface point has two unit normal vectors $\pm n$. The principal curvatures have values $\kappa_i^n = -\kappa_i^{-n}$, and coincide with the usual principal curvatures.

If M is a curve in $\mathbb{R}^3$, whose Frenet frame is given by $\widetilde{e}_1, \widetilde{e}_2, \widetilde{e}_3$, and whose curvature is κ, then any unit normal vector can be written in the form $n = \cos\phi \cdot \widetilde{e}_2 + \sin\phi \cdot \widetilde{e}_3$. It is easy to verify that $\kappa_1^n = \kappa \cos\phi$.

A curve $c(t)$ in Euclidean $\mathbb{R}^2$ parametrizes a one-dimensional surface (we have $m = 1$ and $d = 2$). If the unit normal vector $n(t)$ points to the same side of the curve as $c''(t)$, then $\kappa_1^n = |\kappa|$ is positive, with κ being the curvature of the curve. For other normal vectors we use the relation $\kappa_1^{-n} = -\kappa_1^n$.

So the cases which are most relevant for curve and surface design ($d = 2, 3$, $m = 1, 2$) are rather elementary.

2.3. **Acceleration of the ICP algorithm and the choice of the functions $\widetilde{\operatorname{dist}}_x$.** The functions $\widetilde{\operatorname{dist}}_{x_i^0}$ mentioned in Algorithm 2 of Sec. 2.1 must be approximants of the squared distance from the target, they should be quadratic (otherwise minimization is difficult), and positive semidefinite (otherwise minimizing does not make sense). Using a quadratic approximant would suggest to use the second order Taylor polynomial, but this won't work in all cases: For small δ, the quadratic Taylor approximants as given by Equ. (10) are never positive semidefinite in both cases $\delta > 0$ and $\delta < 0$, unless all principal curvatures happen to vanish. However, for $\delta = 0$ and also in the limit case $\delta \to \infty$ we always have positive semidefiniteness.

Algorithm 2 becomes Alg. 1, if the quadratic approximant $\widetilde{\operatorname{dist}}_p^2$ of $\operatorname{dist}(\cdot, T)^2$ is chosen as

$$\widetilde{\operatorname{dist}}^2_{(0,\ldots,0,\lambda)} = \sum x_i^2 \tag{11}$$

(in the coordinate system given by Equ. (8)), for all λ. This is the limit case $\delta \to \infty$ of Equ. (10) — $\widetilde{\operatorname{dist}}$ is the Taylor expansion of $\operatorname{dist}(\cdot, T)^2$ 'at infinity'. The second possibility wich is always positive semidefinite is given by the Taylor expansion in the surface point $(0, \ldots, 0)$ itself:

$$\widetilde{\operatorname{dist}}^2_{(0,\ldots,0,\lambda)} = \sum_{i>m} x_i^2, \tag{12}$$

for all λ. Note that Equ. (12) computes the squared distance from the tangent space of the surface at the footpoint.

The positive semidefinite quadratic function which in some way is 'closest' to the Taylor approximant in the point $(0, \ldots, 0, \delta)$ itself is found if we cancel negative terms in Equ. (10):

$$\widetilde{\operatorname{dist}}^2_{(0,\ldots,0,\delta)} = \sum_{\substack{i\text{: coeff.of } x_i^2 \\ \text{is nonneg.}}} \frac{\delta}{\delta - 1/\kappa_i^n} x_i^2 + \sum_{i>m} x_i^2, \tag{13}$$

A refined ICP algorithm, second version:
input: $b_1, \ldots, b_k$
choose λ such that $0 \leq \lambda \leq 1$, preferably small.
repeat
  for $i = 1, \ldots, r$
    evaluate feature points $x_i^0 = x_i(b_1, \ldots, b_k)$
    compute footpoints $y_i = F_T(x_i^0)$
    compute T's tangent planes T_i at y_i.
  choose control points $b_1, \ldots, b_k$
    such that $w := \sum_{i=1}^{r}(\lambda\|x_i - y_i\|^2 + (1-\lambda)\mathrm{dist}(x_i, T_i)^2)$
    is minimized, where $x_i = x_i(b_1, \ldots, b_k)$.
until w small enough.
result: current value of $b_1, \ldots, b_k$.

Algorithm 3

Depending on the circumstances, the computation of the principal curvatures may be computationally expensive.

Numerical evidence shows that using Equ. (12) in Algorithm 2 leads to much faster convergence, but introduces instability. It turns out that a *convex combination* of Equ. (11) and Equ. (12) results in an algorithm which is both fast and stable (Algorithm 3). An additional feature which is computationally attractive is that it does not require the computation of principal curvatures.

2.4. **Computation of the distance field.** Numerical computation of the distance field of the target T means collecting data which are sufficient for evaluating the distance from T for any point of space (with varying accuracy, depending on the application), or even computing quadratic approximants of the squared distace, as described above.

For the purposes of the ICP algorithm fast methods for solving the eikonal equation $\|\mathrm{grad} f(x)\| = 1$ offer an approach to this problem. (Pottmann and Leopoldseder, 2003) present a data collecting strategy based on a linear-complexity sweeping method of (Zhao, 2002) which allows computing the functions $\widetilde{\mathrm{dist}}_{x_i^0}$ if the target is a polyhedral surface or a point cloud.

3. Near-Euclidean spline motions

3.1. **Motions as curves.** An affine motion is a curve $(A(t), a(t))$ in the affine space of affine transformations: The affine mapping characterized by the pair (A, a) is defined by

$$(A, a) \in \mathit{Aff}_d : x \mapsto Ax + a \quad (A \in \mathbb{R}^{d\times d},\ a \in \mathbb{R}^d). \tag{14}$$

It is easy to construct and control *affine* motions by control positions, but not so easy to do the same for Euclidean motions: $\mathit{Aff}_d = \mathbb{R}^{d\times d+d}$ contains the group G of Euclidean congruence transformations as a $d(d+1)/2$-dimensional submanifold. The elements of G are defined by the condition that A is an orthogonal matrix, i.e.,

$$(A, a) \in G \iff A^T A = E_d, \tag{15}$$

with E_d being the $d \times d$ identity matrix. G consists of two components, namely the subgroup G^0 of orientation-preserving Euclidean congruence transformations (the Euclidean *motions*), and a second component whose elements reverse orientation:

$$(A, a) \in G^0 \iff A^T A = E_d,\ \det A > 0. \tag{16}$$

By actively moving a spline curve $(A(t), a(t))$ in $\mathbb{R}^{d\times d+d}$ towards G or G^0 we get near-Euclidean spline motions. An example of such a motion is given by Fig. 2, right.

3.2. **The distance field of the Euclidean motion group.** In order to be able to use the ICP algorithm and its variants for motion design, we have to introduce a Euclidean metric in $\mathbb{R}^{d\times d+d}$ and to compute footpoints on G and G^0. It makes sense to choose a left-invariant metric, as the approximant should be independent of the choice of coordinate system. The distance field of the groups G and G^0 with respect to appropriate invariant metrics has been considered in (Horn, 1987), (Higham 1989), (Shoemake and Duff, 1992), (Belta and Kumar, 2002), and (Wallner, 2002). One particular definition of a distance $d(f, g)$ between mappings f and g is to choose points $w_1, \ldots, w_r$ in the domain of f, g and let

$$d(f, g)^2 = \sum_{i=1}^{r} \|f(w_i) - g(w_i)\|^2. \tag{17}$$

This definition gives us a metric of the desired type. More generally, we could choose a mass distribution μ and instad of a sum use the integral

$$d(f, g)^2 = \int \|f(x) - g(x)\|^2 \, d\mu(x). \tag{18}$$

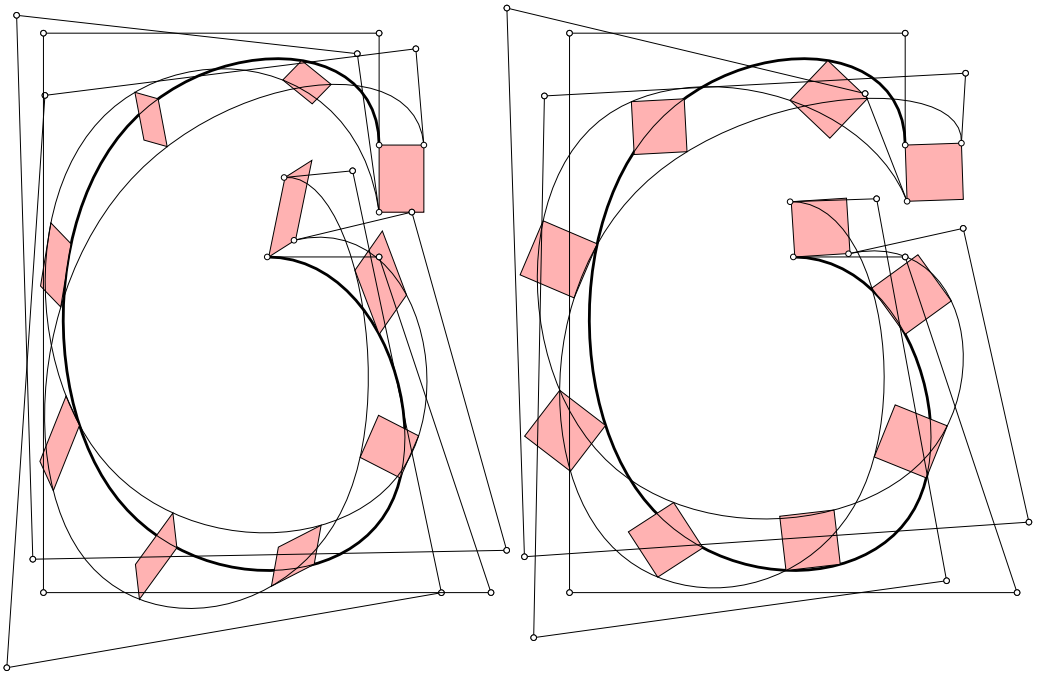

FIGURE 2. Left: Affine planar spline motion with control points of three point paths (initial value for Ex. 3.4). Right: Near-Euclidean spline motion.

The previous definition is the special case of unit point masses at locations $w_1, \ldots, w_r$. It turns out that all left invariant metrics on O_n can be written in this way. We summarize the results as presented in (Wallner, 2002): For a given mass distribution (i.e., a positive Borel measure) $\mu \in \mathbb{R}^d$ we consider the L^2 space of mappings f of $\mathbb{R}^d$ into $\mathbb{R}^d$:

$$f \in L^2_\mu(\mathbb{R}^d, \mathbb{R}^d) \iff \int \|f(x)\|^2 \, d\mu < \infty, \tag{19}$$

with the usual identification of functions which are equal μ-almost everywhere. We assume that μ is such that

$$G^0 \subset G \subset \mathit{Aff}_d \subset L^2_\mu(\mathbb{R}^d, \mathbb{R}^d). \tag{20}$$

Total mass $|\mu|$ and the inertia tensor J are defined by

$$|\mu| = \int 1 \, d\mu, \quad \langle a, Jb \rangle = \int \langle a, x \rangle \langle b, x \rangle \, d\mu(x), \;\; J_{kl} = \int x_k x_l \, d\mu(x). \tag{21}$$

Without loss of generality we assume a coordinate system such that the barycenter of μ is located in the origin:

$$\int x \, d\mu(x) = 0. \tag{22}$$

Then the restriction of the L^2 scalar product to the linear subspace $Aff_d = \mathbb{R}^{d\times d+d}$ of L^2 is given by

$$\langle (A,a),(B,b)\rangle = \operatorname{tr}(A^T B J) + |\mu|\langle a,b\rangle, \tag{23}$$

where $\langle a,b\rangle$ denotes the canonical scalar product in $\mathbb{R}^d$. The distance $d(f,g)$ of L^2 functions is given by $d(f,g)^2 = \langle f-g, f-g\rangle$. Obviously the inertia tensor J and the total mass μ determine the distance function. The computation of footpoints and related results are summarized in the following theorems. We use the symbols O_d and SO_d for the orthogonal group and the special orthogonal group in $\mathbb{R}^d$, respectively.

Theorem 1. *The vector $(X,x) \in \mathbb{R}^{d\times d+d}$ is tangent (orthogonal, resp.) to G or G^0 in the point (P,p), if $P^T X$ is skew-symmetric (if $P^T X J$ is symmetric and $x = 0$, resp.).*

Theorem 2. *Assume that (A,a) is an affine transformation, and that*

$$AJ = Q_1 D Q_2 \tag{24}$$

is a singular value decomposition with $Q_1, Q_2 \in \mathrm{O}_d$ and a nonnegative diagonal matrix D. Then (Q_1Q_2, a) is a footpoint of (A,a) in G, and vice versa. The footpoint is unique if and only if $\det A \neq 0$.

Theorem 3. *Assume that (A,a) is an affine transformation, and that*

$$AJ = Q_1' D' Q_2' \tag{25}$$

is an SVD-type decomposition as follows: If $\det A > 0$ it is the ordinary SVD. If $\det A = 0$, it is an ordinary SVD such that $\det Q_1' \det Q_2' > 0$. If $\det A < 0$, then it is such that $Q_1', Q_2' \in \mathrm{O}_d$, $\det Q_1' \det Q_2' > 0$, and

$$D' = \operatorname{diag}(w_1, \ldots, w_{d-1}, -w_d) \text{ with } w_1 \geq \cdots \geq w_d \geq 0 \tag{26}$$

($w_1, \ldots, w_d$ are the singular values of AJ). Then in all three cases, $(Q_1'Q_2', a)$ is a footpoint of (A,a) on G^0 and vice versa. The footpoint is unique if $\det A > 0$. In the case $\det A \leq 0$ it is unique if and only if the smallest eigenvalue of JA^TAJ has multiplicity one.

Theorem 4. *The footpoint in G of $(A,a) \in Aff_d$ depends smoothly (indeed, analytically) on (A,a) if $\det A \neq 0$. The same holds true for G^0 if $\det A \neq 0$ and the footpoint is unique.*

Proofs can be found in (Wallner, 2002).

Remark: Theorems 2 and 3 show how to compute, for a given affine transformation, the nearest Euclidean motion. This is useful if we are given an affine transformation (A,a), which is already near-Euclidean, and which is to be applied to an actual rigid body. Within tolerance, we may apply the footpoint of (A,a) in the motion group. By Th. 4,

the dependence of the footpoint on (A, a) is smooth in a certain (big) neighbourhood of the motion group.

3.3. **Differential geometry of the motion group.** It is well known that in a (by no means 'small') neighbourhood of E_d, the group O_d can be regularly parametrized by the exponential of skew-symmetric matrices. Thus we parametrize the group G in the neighbourhood of a point (P, p) by

$$\mathbb{R}^{d\times d}_{\mathrm{skew}} \times \mathbb{R}^d \to \mathit{Aff}_d, \quad (X, x) \mapsto (P \cdot \exp(X), x) \tag{27}$$

Surface parameters u_{ij} $(1 \le i < j \le d)$ and u_k $(1 \le k \le d)$ are defined by

(28)

$$X = \sum_{i<j} u_{ij}(E_{ij} - E_{ji}) = \begin{bmatrix} 0 & u_{12} & \dots & u_{1d} \\ -u_{12} & 0 & \dots & u_{2d} \\ \vdots & & \ddots & \end{bmatrix}, \quad x = (u_1, \dots, u_d).$$

where E_{ij} is a matrix whose only nonzero entry is in the i-th row and the j-th column. Thus we get the parametrization

$$g : \mathbb{R}^{d(d+1)/2} \to \mathbb{R}^{d\times d+d}, \quad (u_{12}, \dots, u_{d-1,d}, u_1, \dots, u_d) \mapsto$$
$$(P \cdot \exp \begin{bmatrix} 0 & u_{12} & \dots & u_{1d} \\ -u_{12} & 0 & \dots & u_{2d} \\ \vdots & & \ddots & \end{bmatrix}, u_1, \dots, u_d).$$

First partial derivatives at $u = 0$ are

$$\frac{\partial g}{\partial u_{ij}} = (P(E_{ij} - E_{ji}), 0), \quad \frac{\partial g}{\partial u_k} = (0, \delta_{1k}, \dots, \delta_{dk}).$$

Most of the second order partial derivatives $\dfrac{\partial^2 g}{\partial u_{ij}\partial u_{kl}}$ at $u = 0$ are zero. The nonzero ones are given by

$$\begin{aligned} &(-P(E_{ik} + E_{ki}), 0) \quad \text{if } j = l\,; \quad (-P(E_{jl} + E_{lj}), 0) \quad \text{if } i = k\,; \\ &(P(E_{il} + E_{li}), 0) \quad \text{if } j = k\,; \quad (P(E_{jk} + E_{kj}), 0) \quad \text{if } i = l\,; \\ &(-2P(E_{ii} + E_{jj}), 0) \quad \text{if } (i, j) = (k, l). \end{aligned} \tag{29}$$

From here the computation of principal curvatures with respect to a unit normal vector (N, n) runs as described in Sec. 2.2.

Remark: A geometric interpretation of these principal curvatures or the principal curvature vectors which does not involve the L^2 distance is not apparent to the author.

3.4. **Numerical example.** Fig. 2 shows the result of actively moving an affine cubic B-spline motion

$$(B(t), b(t)) = \sum_{i=0}^{k} N_i^3(t)(B_i, b_i) \tag{30}$$

towards the Euclidean motion group. Here $N_i^3(t)$ are the cubic B-spline basis functions defined by the knot list $(0, 0, 0, 0, 1/2, 2/3, 1, 1, 1, 1)$. The linear parts B_i of the spline coefficients $(B_i, b_i) \in \mathbb{R}^{2\times 2+2}$ $(i = 1, \ldots, 7)$ are listed below; the vectors b_i determine the translational part of the resulting motion, but have no influence on its euclidicity.

$$\begin{aligned}
B_1 &= \begin{bmatrix} 1.005 & 0.033 \\ -0.034 & 1.005 \end{bmatrix}, & B_2 &= \begin{bmatrix} 1.080 & -0.716 \\ 0.716 & 1.080 \end{bmatrix}, \\
B_3 &= \begin{bmatrix} -0.449 & -1.12 \\ 1.122 & -0.449 \end{bmatrix}, & B_4 &= \begin{bmatrix} -0.641 & -0.806 \\ 0.806 & -0.641 \end{bmatrix}, \\
B_5 &= \begin{bmatrix} -1.260 & 0.212 \\ -0.212 & -1.260 \end{bmatrix}, & B_6 &= \begin{bmatrix} -0.511 & 1.037 \\ -1.037 & -0.511 \end{bmatrix}, \\
B_7 &= \begin{bmatrix} -0.057 & 0.987 \\ -0.987 & -0.057 \end{bmatrix}.
\end{aligned} \tag{31}$$

We used Algorithm 3 with $k = 7$, $r = 50$, $x_i = \sum_{i=0}^{k} N_i^3(\frac{i}{k})(B_i, b_i)$, $\lambda = 0.01$, and the footpoint map given by Th. 2, with $d = 2$, $|\mu| = 1$, $J = \operatorname{diag}(1.3, 1.0)$. The average squared distance of the points x_i from the Euclidean motion group during the iteration was 0.3983, 0.00538, 0.00538, ... (i.e., constant after the second iteration step).

This good behaviour of the iteration procedure is apparently due to the 'good shape' of the Euclidean motion group as a surface in $\mathbb{R}^{d\times d+d}$.

4. Gliding motions

4.1. **The configuration space.** A gliding motion defined by a surface pair M, M' is a path $(A(t), a(t))$ in the Euclidean motion group G^0 which has the property that for all parameter values t the surface $A(t)\cdot M + a(t)$ is in contact with the surface M'. This means that there is $p(t) \in M$ and $p'(t) \in M'$, a normal vector $n(t)$ of M at $p(t)$ and a normal vector $n'(t)$ of M' at $p'(t)$ such that

$$A(t)p(t) + a(t) = p'(t), \quad A(t)n(t) = n'(t). \tag{32}$$

Actually we modify this definition by an additional requirement: We assume that n and n' are unit normal vector fields of M and M', and that both M, M' are (part of) boundaries of solids. We write $n(p)$ and $n'(p')$ for the normal vectors attached to points. We imagine that n

is pointing outward, and n' is pointing inwards. Then the contact of $A(t) \cdot M + a(t)$ with M' is required to happen in a way such that

$$A(t) \cdot n(p(t)) = n'(p'(t)) \tag{33}$$

We say that M and M' are in oriented contact (with respect to previously defined unit normal vector fields).

The set of motions (A, a) such that $AM+a$ is in oriented contact with M' is called the *configuration space* or *configuration manifold* of surface-surface-contact, and will be abbreviated by the letter C. Properties of C relevant for motion design have been investigated in (Pottmann and Ravani, 2000), (Wallner, 1999), and (Wallner, 2000).

4.2. **Differential geometry of the configuration space.** We will describe how the configuration space can be parametrized. This parametrization can be used to compute tangent spaces and principal curvatures.

4.2.1. *Preparations.* We write u short for $(u_1, \ldots, u_{d-1})$. Assume that $g(u)$ and $g'(u)$ parametrize M and M', resp., and that $n(u)$, $n'(u)$ are normal vector fields. We apply Gram-Schmidt orthonormalization to the first $d-1$ vectors of the basis

$$\overline{B}(u) = (n(u), \partial_1 g(u), \ldots, \partial_{d-1} g(u)), \tag{34}$$

and get $(b_0(u), \ldots, b_{d-2}(u))$. The vector $b_{d-1}(u)$ is uniquely determined by the requirement that

$$B(u) = (b_0(u), \ldots, b_{d-1}(u)) \tag{35}$$

is an orthonormal basis with positive determinant. The same we do for M' and get B'. By the nature of the Gram-Schmidt process, the tangent space of M at $p = g(u)$ is given by

$$T_p M = g(u) + [b_1(u), \ldots, b_{d-1}(u)], \tag{36}$$

and analogously for M'. Further,

$$b_0 = n/\|n\|, b_1 = \partial_1 g/\|\partial_1 g\| \quad b'_0 = n'/\|n'\|, b'_1 = \partial_1 g'/\|\partial_1 g'\|. \tag{37}$$

Now assume that (A, a) is a Euclidean motion such that $A \cdot M + a$ touches M' in $A \cdot g(u) + a = g'(u')$ in such a way that the normal vectors of M and M' are mapped onto each other by A:

$$Ab_0 = b'_0. \tag{38}$$

The linear mapping $L = L(A, u, u') \in \mathrm{SO}_d$ is defined by

$$AB(u) = B'(u')L. \tag{39}$$

Conversely, if u, u', L are given, Equ. (39) defines A.

In the *Euclidean plane* ($d = 2$), the tangent space of both M and M' is one-dimensional. We required that $Ab_0 = b'_0$, so it follows that either $Ab_1 = b'_1$ or $Ab_1 = -b'_1$. As A was supposed to be a motion, we have $Ab_1 = b'_1$, and L is the identity. The fact that there is no freedom left for L is in accordance with the result that in the Euclidean plane the position $AM + a$ of a 1-surface (i.e., curve) M is uniquely determined if we know which point of $AM + a$ is in contact with with point of M'.

In *Euclidean three-space* ($d = 3$), the tangent spaces of both M and M' are two-dimensional. As $Ab_0 = b'_0$, L necessarily has the form

$$L = \begin{bmatrix} 1 & & \\ & \cos\phi & -\sin\phi \\ & \sin\phi & \cos\phi \end{bmatrix}, \quad \phi = \sphericalangle(Ab_1, b'_1) = \sphericalangle(A \cdot \partial_1 g, \partial_1 g'). \tag{40}$$

In general L has the block matrix structure

$$L = \begin{bmatrix} 1 & 0 \\ 0 & L_1 \end{bmatrix} \quad \text{with } L_1 \in \mathrm{SO}_{d-1}. \tag{41}$$

4.2.2. *Parametrization of C.* Having set up parametrizations g, g', normal vector fields n, n', frame fields B, B', and the correspondence between A, B, B', and L, we may parametrize the configuration space defined by M, M' and the normal vector fields n, n' as follows: We parametrize M with $d-1$ parameters $u_1, \ldots, u_{d-1}$, and M' with $u'_1, \ldots, u'_{d-1}$. Further, we parametrize the set of possible matrices L. According to the discussion in Sec. 4.2.1, there is nothing to do if $d = 2$; for $d = 3$ a parametrization is given by Equ. (40). In higher dimensions, the set of possible L's is given by Equ. (41), and we may parametrize SO_{d-1} in a way analogous to Equ. (27).

In any case, we write '$L(\phi)$' symbolically for these parameters (their number equals $(d-1)(d-2)/2$). Then the points (A, a) of the configuration space are parametrized by

$$(A, a)(u, u', \phi) = (B'(u')L(\phi)B^T(u), g'(u') - Ag(u)). \tag{42}$$

The total number of parameters equals $2(d-1) + (d-1)(d-2)/2 = (d+2)(d-1)/2 = \dim C = \dim G - 1$.

4.2.3. *Tangent space and orthogonal space of C.* The tangent space of the configuration manifold might be computed via a parametrization. It turns out that the singularities of the parametrization (42) are actual singularities of C if both g and g' are regular. The singularities can be characterized by the result of (Wallner, 2000), which is given below as Th. 5, and which uses the notion of second order line contact: It may happen that M, M' touch each other in the points of a curve, not only in one point. This is called line contact of M, M'. *Second order*

line contact means that there is a surface M'' which is in second order contact with M and in line contact with M'. This is equivalent to the difference of second fundamental forms being singular. For this concept, see also (Pottmann and Wallner, 2001), p. 458.

Theorem 5. *Assume that $AM + a$ touches M' in $Ap + a = p'$. The configuration space defined by M, M' is a regular $(d + 2)(d - 1)/2$-dimensional surface in $\mathbb{A}\mathrm{ff}_d$ in a neighbourhood of (A, a) if $AM + a$ and M' are not in second order line contact.*

In the regular case, however, the tangent space is easily described without reference to curvature:

Theorem 6. *Assume that $AM + a$ touches M' at $Ap + a = p'$, and that the unit surface normals of M and M' at p and p' are given by vectors n and n', respectively. Assume further that the conditions of Th. 5 on the regularity of the configuration manifold are fulfilled. Then*

$$(X, x) \in T_{(A,a)}C \iff A^T X \text{ skew-symmetric}, \; x + Xp \in T_{p'}M' \tag{43}$$

The orthogonal space $\perp_{(A,a)}C$ is spanned by $\perp_{(A,a)}G$, which according to Th. 1 consists of the pairs $(X, 0)$ with $A^T X J$ symmetric, and by either

$$(-|\mu| \cdot A\, p\, n^T J^{-1}, n') \quad \text{or} \quad (|\mu| \cdot n' p^T J^{-1}, n'). \tag{44}$$

Proof. Equ. (43) is well known, a proof can be found in (Wallner, 2000) and (Pottmann and Wallner, 2001), pp. 454ff. It means that $T_{(A,a)}C$ is spanned by infinitesimal rotations $(X, -Xp)$ which assign the velocity $Xp + (-Xp) = 0$ to the point of contact; and by infinitesimal translations $(0, x)$ with x tangent to M' at the contact point.

As $T_{(A,a)}C \subset T_{(A,a)}G$, for their orthogonal complements the reverse inclusion holds true. The difference in dimension between both spaces equals one. It remains to show that the two vectors given by Equ. (44) are contained in $\perp_{(A,a)}C$, but not in $\perp_{(A,a)}G$. The latter is clear because of Th. 1 and $n \neq 0$. In order to establish the former, we compute scalar products with $(X, x) \in T_{(A,a)}C$.

$$\langle (X, x), (-|\mu| A\, p\, n^T J^{-1}, n') \rangle = -|\mu| \cdot \mathrm{tr}(X^T A\, p\, n^T J^{-1} J) + |\mu| \langle x, n' \rangle. \tag{45}$$

We let $\widetilde{X} = A^T X$, which implies $\widetilde{X}^T A^T = X^T$, and use $\langle a, b \rangle = \mathrm{tr}(ab^T)$ to modify the expression in Equ. (45) involving trace:

$$-\mathrm{tr}(\widetilde{X}^T A^T A\, p\, n^T) = \mathrm{tr}(\widetilde{X} p\, n^T) = \langle \widetilde{X} p, n \rangle = \langle A \widetilde{X} p, A\, n \rangle = \langle Xp, n' \rangle. \tag{46}$$

Thus the scalar product of Equ. (45) reduces to $|\mu| \langle Xp + x, n' \rangle$. As $Xp + x$ is tangent to M' in p', it equals zero. As to the second vector

mentioned in Equ. (44), we compute

$$\langle (X,x),(|\mu|\cdot n' p^T J^{-1}, n')\rangle = |\mu|\cdot \mathrm{tr}(X^T n' p^T J^{-1} J) + |\mu|\langle x, n'\rangle. \quad (47)$$

In a way analogous to above, we express the trace in terms of a scalar product:

$$\begin{aligned}\mathrm{tr}(\widetilde{X}^T A^T A\, n\, p^T) &= \mathrm{tr}(\widetilde{X}^T n\, p^T) = \mathrm{tr}(np^T \widetilde{X}^T) = \mathrm{tr}(n(\widetilde{X}p)^T)\\ &= \langle n, \widetilde{X}p\rangle = \langle n', Xp\rangle.\end{aligned}$$

It follows that also in the second case the scalar product is zero. The theorem is proved. □

4.2.4. *Curvatures.* By using the procedure described in Sec. 2.2.1, it is possible to compute principal curvatures from the parametrizion of C given by Sec. 4.2.2

4.3. **Footpoints on the configuration manifold.** Computing the footpoint of an element $(A,a) \in \mathit{Aff}_d$ on the configuration space is not as easy as computing footpoints on the Euclidean motion group itself: In contrast to Th. 1 (which enables to derive Th. 2 and 3), Th. 6 does not provide an explicit formula for computing footpoints.

The method of (Pottmann and Leopoldseder, 2003) mentioned in Sec. 2.4 cannot be used directly, with the configuration manifold as a target, because the amount of data handled by it grows exponentially with the dimension of the space it works in. It is however useful in another way, see Sec. 4.3.2.

4.3.1. *An iterative algorithm: Overview.* A rough approximation of such a footpoint is found in the following way: For given $(A,a) \in \mathbb{R}^{n\times n+n}$, we first compute the footpoint (B,b) in G or G^0. Next we assume that M' is endowed with an *oriented distance* $\overrightarrow{\mathrm{dist}}(\cdot, M')$, which is zero on M', positive outside M' and negative inside M'. Then we look for points p, p' such that

$$\begin{aligned}&\overrightarrow{\mathrm{dist}}(Bp+b, M') \to \min \quad (p \in M)\\ &\mathrm{dist}(p', Bp+b) \to \min \quad (p' \in M').\end{aligned} \quad (48)$$

It follows that the line segment p, p' is orthogonal to both surfaces $BM + b$ and M'. The Euclidean congruence transformation

$$(Q,q) = (B, b + (Bp - p')) \quad (49)$$

will be contained in G or G^0, and is also contained in the configuration space. There is of course no reason why (Q,q) should be (A,a)'s footpoint in C, but if (A,a) is in C, it certainly is. As both the mapping $(A,a) \to (Q,q)$ and the footpoint mapping are smooth if we stay away

Footpoints on the configuration space:
input: (A, a)
compute footpoint (B, b) of (A, a) on G
repeat
  compute (Q, q) from (B, b) according to Equ. (49)
  compute tangent space $T_{(Q,q)}C$
  compute orthogonal projection $(Q, q) + (V, v)$ of (A, a) onto $T_{(Q,q)}C$
  choose path $(Q(t), q(t))$ in group:
    such that $(Q(0), q(0) = (Q, q)$ and $(\dot{Q}(0), \dot{q}(0)) = (V, v)$
  choose parameter value $\tau \leq 1$
  $(B, b) := (Q(\tau), q(\tau))$
until (V, v) is small enough.
result: footpoint is (B, b).

Algorithm 4

from medial axes, it is clear that (Q, q) converges to the footpoint if (A, a) converges towards C. This is the basis of an iterative algorithm (Alg. 4) for computing footpoints on the configuration space, the details of which are explained in §§ 4.3.3 and 4.3.4. It works by iterately computing an approximate footpoint according to Equ. (49), and by performing an additional Newton-type shooting step.

4.3.2. *Computation of shortest distance.* Algorithm 4 requires computing the shortest distance between two surfaces. This is a very general problem and efficient solutions often depend on more specific information on the surfaces involved. From the many contributions to this subject we mention only (Sun et al., 2002). In the case of Algorithm 4 one of the two surfaces remains the same for all instances of this problem, so it is useful to employ the approach proposed by (Pottmann and Leopoldseder, 2003), which allows evaluation of the distance from a surface after a certain data structure representing the distance field has been initialized. We do not go into details here.

4.3.3. *Projection onto the tangent space.* Algorithm 4 requires computing the orthogonal projection of a point onto the tangent space $T_{(Q,q)C}$ of the configuration manifold C: Assume that $QM + q$ touches M' in the point $Qp + q = p'$. By Th. 6 and especially Equ. (43), the linear space parallel to $T_{(Q,q)}C$ is spanned by the following $(d + 2)(d - 1)/2$

elements of $\mathbb{R}^{d\times d+d}$:

$$\begin{array}{ll} (Q(E_{ij}-E_{ji}), -Q(E_{ij}-E_{ji})p) & (1\le i<j\le d), \\ (0, v_i) & (1\le i<d,\ v_i\in T_{p'}M') \end{array} \tag{50}$$

We number them in the form

$$(W_1, w_1), \ldots, (W_s, w_s). \tag{51}$$

Then the orthogonal projection $(Q,q)+(V,v)$ of $(A,a)=(Q,q)+(X,x)$ onto the tangent space is uniquely determined by coefficients $\lambda_1,\ldots,\lambda_s$ such that

$$(V,v)=\sum_{i=1}^{s}\lambda_i(W_i,w_i). \tag{52}$$

It is well known that $\lambda_1,\ldots,\lambda_s$ are solutions of the following linear system of equations

$$\sum_{j=1}^{s}\langle(W_i,w_i),(W_j,w_j)\rangle\lambda_j=\langle(X,x),(W_i,w_i)\quad(i=1,\ldots,s). \tag{53}$$

4.3.4. *Paths in the motion group.* Algorithm 4 further requires a path $(Q(t),q(t))$ in the group G which emanates from a given point $(Q,q)=(Q(0),q(0))$ and which has (V,v) has an initial tangent vector. There are many curves which satisfy this condition. One particular choice which is not subject to non-invariant arbitrariness is the stationary motion starting in (Q,q) and having $x\mapsto Vx+v$ as stationary velocity field: It is well known that it is parametrized by

$$(Q(t),q(t))=(Qe^{tY}, Q\frac{e^{tY}-1}{tY}(tY)+q),\ \text{with } Y=Q^{-1}V \tag{54}$$

Here e^Y and $(e^Y-1)/Y$ are matrix functions defined by the power series $\sum_{k=0}^{\infty}Y^k/k!$ and $\sum_{k=0}^{\infty}Y^k/(k+1)!$, respectively.

Other paths with tangent vector (V,v) are given by $(Q(t),q(t)=(Q\exp(Q^T(tV)),\ tv+q)$ or $(Q(t),q(t)=F_G((Q,q)+t(V,v))$, where F_G means the footpoint map onto the group G.

4.3.5. *The medial axis of the configuration manifold.* The projection of an affine position $(A,a)\in \mathit{Aff}_d$ onto the configuration manifold C is not well defined if (A,a) is contained in C's medial axis. One particular instance of this case is that (A,a) is Euclidean and $AM'+a$ touches M in two points. (A,a) has distance zero from two different branches of C, and C's medial axis passes through (A,a). In an actual application, if M and M' are thought to be the boundaries of solids, this situation means that an umwanted collision of M' and M is imminent.

The general case of (A, a) being on C's medial axis means that M' does not know in which direction to move in order to come closer to M. This is a problem of the input data rather than a problem of the algorithm. Part of this problem is adressed in Th. 6 of (Wallner, 2000), which gives conditions where C has no self-intersections.

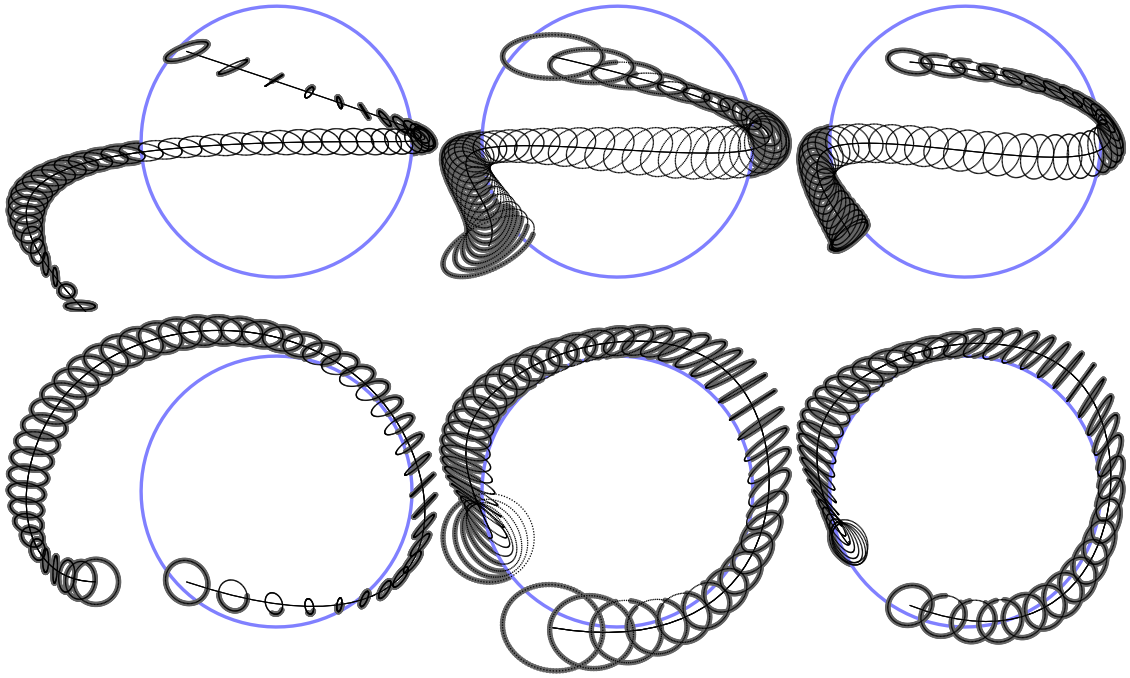

FIGURE 3. Gliding motion: Left: Initial affine motion. Center: after 1 step of Alg. 3. Right: after 5 iterations.

4.4. **Numerical examples.** Fig. 3 shows the result of applying Algorithm 3 to an affine spline motion. From left to right the initial affine spline motion, then the motion after one iteration step, and finally the motion after five iteration steps are shown.

We used a cubic B-spline motion $(Q(t), q(t))$ defined by control elements (B_1, b_1), ..., $(B_6, b_6) \in \mathbb{R}^{3\times 3+3}$, so $r = 6$. We let $k = 50$, $x_i = (Q(\frac{i}{k}), Q(\frac{i}{k}))$, and $\lambda = 0.01$. The footpoint map F_T is the one described in Sec. 4.3. The surface M is a thin torus, and M' is a sphere.

The following table shows average squared distance δ_G from the Euclidean motion group, the average squared distance δ_C from the configuration manifold, and the average weighted distance w which is minimized by Algorithm 3, during the first steps of the iteration.

(55)

No. of iterations	δ_G	δ_C	w
0	$4.39 \cdot 10^{-1}$	$9.11 \cdot 10^{-0}$	$8.48 \cdot 10^{-0}$
1	$2.57 \cdot 10^{-1}$	$2.90 \cdot 10^{-1}$	$2.86 \cdot 10^{-1}$
2	$5.25 \cdot 10^{-3}$	$7.71 \cdot 10^{-3}$	$7.44 \cdot 10^{-3}$
3	$4.14 \cdot 10^{-4}$	$1.31 \cdot 10^{-3}$	$1.18 \cdot 10^{-3}$
4	$2.84 \cdot 10^{-4}$	$9.40 \cdot 10^{-4}$	$8.44 \cdot 10^{-4}$

Gliding motions on polyhedral surfaces are to be modeled over an appropriate knot vector — smooth motions will not be able to glide on non-smooth objects.

5. Extensions and Applications

5.1. **More-parameter motions.** A Euclidean l-parameter motion of a rigid body is an l-dimensional surface contained in the Euclidean motion group G^0. An affine l-parameter motion is an l-dimensional surface contained in the space $\mathit{Aff}_d = \mathbb{R}^{d\times d+d}$ of affine transformations. Actively moving such a surface towards G or G^0 is possible with the algorithms described earlier in this paper. The only modification concerns the way feature points are computed from control points. For example, we may choose control points $(B_{ij}, b_{ij}) \in \mathit{Aff}_d$, knot lists $t_0 \le t_1 \le \ldots$, $\widetilde{t}_0 \le \widetilde{t}_1 \le \ldots$, and define a bicubic B-spline surface by letting

$$(B(u,v), b(u,v)) = \sum_{i=0}^{k_1} \sum_{j=0}^{k_2} N_i^3(u) \widetilde{N}_j^3(v) (B_{ij}, b_{ij}), \tag{56}$$

with N_i^3 and $\widetilde{N}_j^3$ being the B-spline basis functions defined by the knot lists t_i and $\widetilde{t}_i$.

5.2. **Hermite-like interpolation of contact positions.** Assume that $(A_0, a_0), \ldots, (A_k, a_k)$ are positions of a rigid body B such that $A_i(B)+a_i$ touches a given surface O. A one-parameter near-Euclidean motion interpolating (A_i, a_i) is an interpolating curve in $\mathbb{R}^{d\times d+d}$, which lies as close as possible to the contact manifold C defined by B and O.

If we use a spline curve defined by the derivative vectors (V_i, v_i) at (A_i, a_i) (such as the cubic spline in Hermite form), the parameters determined by the minimization process are the coefficients of linear

combination of the (V_i, v_i) in bases of $T_{(A_i,a_i)}C$ $(i = 1, \ldots, l)$. More explicitly assume that

$$(A_1, a_1), \ldots, (A_l, a_l) \tag{57}$$

are contact positions such that A_iM+a_i touches M' in points $A_ip_i+a_i = p'_i$. Then a basis of the tangent space $T_{(A_k,a_k)}C$ $(k = 1, \ldots, l)$ is given by Equ. (50) with (A_k, a_k) instead of (Q, q). A curve $(B(t), b(t))$ in the configuration manifold with $(B(t_k), b(t_k)) = (A_k, a_k)$ has a tangent vector (V_k, v_k) which is a linear combination of this basis:

$$(V_k, v_k) = \sum_{1\le i<j<d} \lambda_{ij}^{(k)} A_k(E_{ij} - E_{ji}) + \sum_{1\le i<n} \lambda_i^{(k)} v_i. \tag{58}$$

The coefficients $\lambda_{ij}^{(k)}$ (together with possible additional control points, depending on the spline scheme) then are the control coefficients which the curve and its feature points depend on. The control coefficients may be determined using the ICP algorithm or one of its variants (cf. the remark at the end of Sec. 2.1).

5.3. **Applications in NC milling.** An important example of a two-parameter gliding motion is the motion of a milling tool along the surface to be manufactured. We take the active part of the tool as surface M and the final shape of the workpiece as surface M'. Possibly not all positions of M' which are contained in the configuration space are admissible as positions of an actual milling tool — it may happen that the tool intersects the interior of M somewhere. Such problems of collision avoidance have been studied in (Wallner and Pottmann, 2000).

A series of one-parameter tool paths can be seen as the u parameter lines in a surface $b(u, v)$ contained in the configuration space. How to construct such a surface has been described in Sec. 5.1.

5.3.1. *Footpoint on the configuration space.* The computation of points p and p' according to Equ. (48) can take advantage of the fact that the milling tool is, geometrically, a surface of revolution, and it surface normals intersect the axis of the tool. The computation of Equ. (48) can be simplified, if the milling tool happens to be a spherical, cylindrical, or toroidal one: In that case the tool has a degenerate inner offset which is a curve or even a point. We may replace the tool by its inner offset and the workpiece by its outer offset at the same distance.

5.3.2. *Collision avoidance.* Other constraints can be incorporated into the framework of active milling paths. One particular example which is important for collision avoidance is that the axis positions of the milling

tool during the motion have to be contained in a certain congruence of lines (i.e., a 2-parameter manifold of lines). After choosing an appropriate distance function in line space (cf. (Pottmann and Wallner, 2001), pp. 195ff), the ICP algorithm which moves affine spline motions towards the configuration space is easy augmented by an additional function measuring the sum of squared distances of a certain number of tool axes to the given line congruence.

Acknowledgements

This research was carried out as part of the project P13938-MAT, which is supported by the Austrian Science Fund (FWF). The authors wants to thank the anonymous referees for their valuable suggestions.

References

[1] Ambrosio, L., Soner, H., 1996. Level set approach to mean curvature flow in arbitrary codimension. J. Differ. Geom. 43, 693–737.

[2] Ambrosio, L., Mantegazza, C., 1998. Curvature and distance function from a manifold. J. Geom. Anal. 8, 723–748.

[3] Belta, C., Kumar, C., 2002. An SVD-based projection method for Interpolation on $SE(3)$. IEEE Trans. Robotics Automation 18/3, 334–345.

[4] Higham, N. J., 1989. Matrix nearness problems and applications. in: Gover M. J. C., and Barnett S. (eds): Applications of Matrix Theory, Oxford University Press. pp. 1–27.

[5] Hofer, M., Pottmann, H., Ravani, B., 2002. Subdivision algorithms for motion design based on homologous points. In: Lenarćić, J., Thomas, F. (eds.): Advances in Robot Kinematics — Theory and Applications, Kluwer Academic Publishers, 2002. pp. 235-244.

[6] Hofer, M., Pottmann, H., Ravani, B., 2002. From curve design algorithms to motion design. Technical Report No. 95, Institut für Geometrie, TU Wien,

[7] Horn, B., 1987. Closed-form solution of absolute orientation using unit quaternions. J. Opt. Soc. Am. A, 4/4, 629–642.

[8] Hyun, D.-E., Jüttler, B., Kim, M.-S., 2001. Minimizing the distortion of affine spline motions. in: in: 9th Pacific Conference on Computer Graphics and Applications (Pacific Graphics 2001), IEEE Computer Society, pp. 50–59.

[9] Kass, M., Witkin, A., Terzopoulos, D., 1988. Snakes: active contour models. Int. J. Computer Vision 1, 321–332.

[10] Röschel, O., 1998. Rational motion design — a survey. Comput. Aided Geom. Design 30, 169–178.

[11] Pottmann, H., Ravani, B., 2000. Singularities of motions constrained by contacting surfaces, Mechanism and Machine Theory 35, 963–984.

[12] Pottmann, H., Wallner, J., Glaeser, G., Ravani, B., 1999. Geometric criteria for gouge-free 3-axis milling of sculptured surfaces. J. Mech. Design 121, 241–248.

[13] Pottmann, H., Wallner, J., 2001. Computational Line Geometry. Springer, Berlin.

[14] Pottmann, H., Hofer, M., 2002. Geometry of the squared distance function to curves. Proceedings of the International Workshop on Visualization and Mathematics in Berlin-Dahlem, May 22–25, 2002, Springer, Berlin.

[15] Pottmann, H., Leopoldseder, S., 2002. A concept for parametric surface fitting which avoids the parametrization problem. Technical Report No. 94, Institut für Geometrie, TU Wien.

[16] Pottmann, H., Leopoldseder, S., Hofer, M., 2002. Approximation with active B-spline curves and surfaces. in: 10th Pacific Conference on Computer Graphics and Applications (Pacific Graphics 2002), IEEE Computer Society, pp. 8–25.

[17] Pottmann H., Leopoldseder, S., 2003. The d^2 tree: A hierarchical representation of the squared distance function. Technical Report No. 101, Institut für Geometrie, TU Wien, March 2003.

[18] Shoemake, K., Duff, T., 1992. Matrix animation and polar decomposition. in: Proceedings of the Graphics Interface '92, Vancouver. Canadian Info Processing Society (pp. 258–264).

[19] Sun, K.-A., Jüttler, B., Kim, M.-S., Wang, W., 2002. Computing the distance between two surfaces via line geometry, in: Proceedings of the Tenth Pacific Conference on Computer Graphics and Applications (S. Coquillart et al., eds.), IEEE Press, Los Alamitos 2002, pp. 236-245.

[20] Wallner, J., Pottmann, H., 2000. On the Geometry of Sculptured Surface Machining. in: Laurent, P.-J., Sablonnière, P., Schumaker, L.L. (eds.), Curve and Surface Design: Saint-Malo 1999, Vanderbilt University Press, Nashville.

[21] Wallner, J., 2000. Configuration space of surface-surface contact. Geom. Dedicata 80, 173–185.

[22] Wallner, J., 2002. L^2 approximation by Euclidean motions, Technical Report No. 93, Institut für Geometrie, TU Wien.

[23] Zhao, H., 2002. Fast Sweeping Method for Eikonal Equations I: Distance Function, preprint available from http://www.math.uci.edu/∼zhao/publication/mypapers/pdf/fastsweep_distance.pdf.

Institut für Geometrie, Technische Universität Wien, Wiedner Hauptstr. 8–10/113, A 1040 Wien