Hausdorff and Minimal Distances between Parametric Freeforms in $I\!R^2$ and $I\!R^3$

Gershon Elber[1] and Tom Grandine[2]

[1] Dept. of Computer Science, Technion – IIT, Haifa 32000, Israel
[2] The Boeing Company, Seattle, USA

Abstract

We present algorithms to derive the precise Hausdorff distance and/or the minimal distance between two freeform shapes, either curves or surfaces, in $I\!R^2$ or $I\!R^3$. The events at which the Hausdorff/minimal distance can occur are identified and means to efficiently compute these events are presented. Examples are also shown and the extension to arbitrary dimensions is briefly discussed.
Keywords: Bisectors, Antipodal points, Algebraic constraints, Spline geometry, Collision detection.

1 Introduction and Previous Work

The need to compute the maximal or minimal distance between two entities in $I\!R^2$ or $I\!R^3$ emerges in a whole variety of applications. Both collision detection calculations and Haptic interaction can greatly benefit from such black boxes [7]. Force feedback for Haptic devices is typically applied once the interaction tool gets closer to the surface of the approached object and is typically applied in the direction from the closest point on that surface. Similarly, the Hausdorff distance computation plays a major role in any approximation method of a curve or a surface by lower degree curves or surfaces or even piecewise linear approximations, as it provides L_∞ bounds over the approximation.

Given two objects, $\mathcal{O}_1, \mathcal{O}_2 \in I\!R^n$, the Hausdorff distance between them is defined as:

$$D_H(\mathcal{O}_1, \mathcal{O}_2) = \max\left(\max_{P \in \mathcal{O}_1} \min_{Q \in \mathcal{O}_2} ||P - Q||, \max_{Q \in \mathcal{O}_2} \min_{P \in \mathcal{O}_1} ||P - Q|| \right).$$

Figure 1 illustrates this definition using the geometric insight that the Hausdorff distance can sometimes (but not always!) be captured as the last contact point of the offset of one shape with the other shape, and vice versa.

That said, not much can be found on the problem and its solution for freeform polynomial geometry or even for piecewise linear polygonal geometry. In [2], point sampling over two polygonal meshes is proposed as an approximation

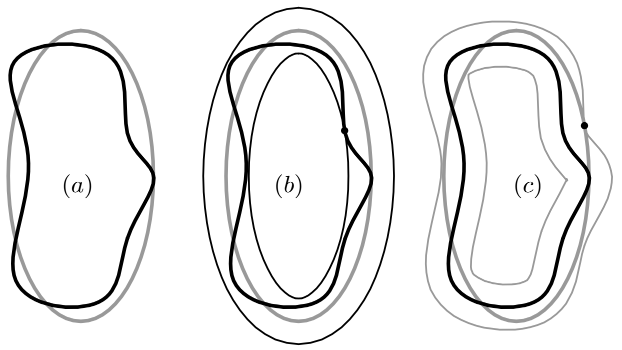

Fig. 1. The Hausdorff distance between two planar curves in (a) could be computed as the maximum offset amount between the last contact point of the offset front of one shape with the second shape (b) and vice versa (c).

for the Hausdorff distance between the two polygonal meshes. [2] offered the 'Metro' point sampling tool that is the acceptable tool in the computer graphics community to estimate Hausdorff distances over polygonal meshes.

In [8], bounds on the Hausdorff distance of two given curves are presented. The cases of two implicits, an implicit and a parametric form, and two parametric form are considered but only to give an upper bound. The bound between two parametric forms is the worst offered of the three and is derived by examining the difference vectors of the corresponding control points, bringing both curves into a comparable representation. This approach further assumes the two curves are fairly close to each other, an assumption that might yield poor answers if not.

It is not common that, in principle, a problem is simpler in the piecewise polynomial domain than in the piecewise linear case. The Hausdorff distance computation between two objects is one such problem. The fact that the polygonal mesh is not tangent plane continuous, makes it very difficult to track the exact position when an event of an extreme distance can occur. In [1], the exact Hausdorff distance between points and freeform planar parametric curves is investigated, taking advantage of the fact that the input is C^1 continuous. The events where the Hausdorff distance can occur are then identified and reduced to a set of differential algebraic constraints. The finite solution set of these constraints is then examined for the actual Hausdorff distance. Our work here builds upon [1] and while we follow a similar approach, we will lay out the differences and also go beyond to consider freeform geometry in $\mathbb{R}^3$ and $\mathbb{R}^n$ as well.

This paper is organized as follows. In Section 2, we consider the problem of the Hausdorff distances in the plane. Extensions of the result to $\mathbb{R}^3$ and $\mathbb{R}^n$ are discussed in Section 3. Minimal distances between freeforms are discussed in Section 4 and examples for distance computations in $\mathbb{R}^2$ and $\mathbb{R}^3$ are presented in Section 5. Possible extensions, and computational considerations are discussed in Section 6 and finally, we conclude in Section 7.

2 Hausdorff Distance in the Plane

We following [1], who presents the necessary algebraic constraints for Hausdorff distances in the plane, and express all the events at which the Hausdorff distance could occur at, between two planar C^1 parametric curves.

Let $C_1(r),\ r \in [0,1]$ and $C_2(t),\ t \in [0,1]$ be two regular [3] C^1 continuous planar parametric curves. Then,

Definition 1. *A* **normal-line** *to $C_1(r)$ at the parameter $r = r_0$ is a line through $C_1(r_0)$ that is parallel to the curve's normal, $N_{C_1}(r_0)$.*

The Hausdorff distance could clearly occur at the end (or C^1 discontinuity) points of one of the curves, if the curves are open (or only piecewise C^1). See Figure 2 (a) and (b). This amounts to examining the distance between the end points of the two curves but also to looking for the normal-lines of $C_2(t)$ that go through $C_1(r_0), r_0 = 0, 1$, if any, or vice versa. These normal-lines' locations could be identified by resolving the following algebraic constraint:

$$\langle C_1(r_0) - C_2(t), C_2'(t) \rangle = 0, \tag{1}$$

having one non-linear equation in one unknown, t, to solve for. The Hausdorff distance between a point and and curve in the plane is now a simple problem that could be reduced to examining end-point vs. end-point events as well as the events satisfied by Equation (1). To consider more events at which the Hausdorff distance could occur between two planar curves, we also need the following:

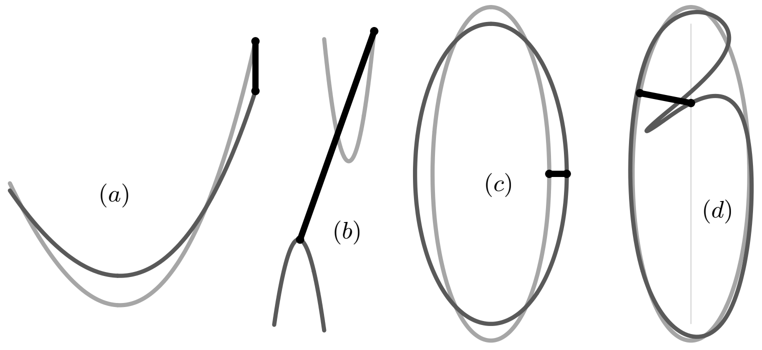

Fig. 2. The Hausdorff distance events (in black) between two curves (in two different gray colors) can occur at either the end points (a), end point of one curve along a normal-line of the other curve (b), antipodal locations (c), or when one (dark-gray) curve intersects with the (think line) self-bisector of the other (light-gray) curve (d).

[3] A parametric form is considered regular if its derivatives span the form's tangent space, at every point in the domain. For a curve C, this amounts to the constraints $||C'|| > 0$.

Definition 2. *The line through $C_1(r)$ and $C_2(t)$ is denoted as a* **curves' bi-normal-line** *at parameters $r = r_0$ and $t = t_0$, if it is a curve normal-line at both $C_1(r_0)$ and $C_2(t_0)$. Points $C_1(r_0)$ and $C_2(t_0)$ are then denoted* **antipodal points**.

The Hausdorff distance between $C_1(r)$ and $C_2(t)$ could also occur at antipodal points of the two curves. See Figure 2 (c) for an example. These antipodal location could be identified using the following set of constraints,

$$\begin{aligned} \langle C_1(r) - C_2(t), C_1'(r) \rangle &= 0, \\ \langle C_1(r) - C_2(t), C_2'(t) \rangle &= 0, \end{aligned} \tag{2}$$

having two equations and two unknowns, t and r, to solve for.

Interestingly enough, these are not the only events at which the Hausdorff distance (event) can occur between two planar curves, and [1] identifies a third case. Let B_i be the self bisector of planar curve C_i. I.e. the locus of points that are equidistant from two different locations on C_i. Then, the Hausdorff distance between $C_1(r)$ and $C_2(t)$ could occur at the locations where $C_1(r)$ intersects B_2 or when $C_2(t)$ intersects B_1 (See Figure 2 (d)). Algebraically speaking, the constraint for $C_1(r)$ to intersect B_2 means that $C_1(r)$ is on the intersection of two independent normal-lines of C_2, at $C_2(t)$ and $C_2(s)$, and further, this intersection is at equal distance from the two *foot points* of these normal-lines, as $C_1(r)$ is on B_2:

$$\begin{aligned} \langle C_1(r) - C_2(t), C_1(r) - C_2(t) \rangle - \langle C_1(r) - C_2(s), C_1(r) - C_2(s) \rangle &= 0, \\ \langle C_1(r) - C_2(s), C_2'(s) \rangle &= 0, \\ \langle C_1(r) - C_2(t), C_2'(t) \rangle &= 0, \end{aligned} \tag{3}$$

where the first constraint makes sure the distances to the two bisector's foot points are the same and the last two constraints ensure the foot directions are orthogonal to the tangents of the curve. In all, Equation (3) presents three constraints in three unknowns, r, s, and t. The first constraint in Equation (3) could be rewritten as

$$\langle C_2(t) - C_2(s), C_2(t) + C_2(s) - 2C_1(r) \rangle = 0, \tag{4}$$

hinting to the fact that the term $(t-s)$ exists in this constraint. Hence, for $t = s$, the first constraint is always satisfied. Further, the last two constraints coalesce so the solver is likely to return the entire domain as a valid solution to Equations (3). In [1], a partial remedy that alleviates the problem is offered by adding a fourth constraint (and a fourth variable u) in the form of $1 - u(t - s) = 0$ to ensure that $t \neq s$, having u within some finite parametric domain. The solution of [1] is not only expensive due to the expansion of the formulation into four equations and four unknowns but will also miss any valid answer where $(t - s)$ is below the $1/u$ selected resolution.

A simpler yet more efficient and more robust alternative approach that one can employ in this specific case, is to divide all input curves at all locations where the curvature, κ, achieves an extremum. I.e. solve first for the locations where C_i satisfy $\kappa_i' = 0$, $i = 1, 2$, and split the two curves at those extrema. One should note that while κ is not rational in general, κ^2 is. Then, and since s and t must be on the opposite sides of some curvature extrema parameter value, one only needs to deal with three different curves, two of which are segments of C_2.

An even better and more general solution would aim at eliminating the $(t-s)$ term from Equations (3) before attempting to solve Equations (3), an approach we are taking in this work. In [9], we present an algorithm to algebraically decompose and remove a $(t-s)$ term from a function known to hold such a term, when the function is in either a Bézier or a B-spline form.

In [1], the subdivision solver of [10] is employed to solve these algebraic constraint. While [10] supports only Bernstein polynomials, as part of this work we use a similar solver that is capable of handling piecewise polynomials B-spline constraints as well [3, 5]. All examples presented in this work employ the solver [3, 5] over the B-spline domain, that is implemented using the IRIT [6] solid modeling environment.

3 Hausdorff Distances in $\mathbb{R}^3/\mathbb{R}^n$

Interesting enough, Equations (1), (2) and (3) holds for $\mathbb{R}^n$, and specifically, for $\mathbb{R}^3$. The direct extensions of Definitions 1 and 2 to $\mathbb{R}^n$ paves the way to the rest of the necessary extensions:

Definition 3. *A* **normal-line** *in* $\mathbb{R}^n$ *to a parametric form* $F(\mathbf{u})$, $\mathbf{u} = (u_1 \cdots u_m)$ *at the parametric location* $\mathbf{u} = \mathbf{u}_0$ *is a line through* $F(\mathbf{u_0})$ *that is also in the normal space of* F *at* $\mathbf{u_0}$.

Definition 4. *The line in* $\mathbb{R}^n$ *through* $F(\mathbf{u})$, $\mathbf{u} = (u_1 \cdots u_m)$ *and* $G(\mathbf{v})$, $\mathbf{v} = (v_1 \cdots v_n)$, *is denoted as* F *and* G*'s* **bi-normal-line** *at parameters* $\mathbf{u} = \mathbf{u}_0$ *and* $\mathbf{v} = \mathbf{v}_0$, *if it is a normal-line at both* $F(\mathbf{u_0})$ *and* $G(\mathbf{v_0})$. *Points* $F(\mathbf{u_0})$ *and* $G(\mathbf{v_0})$ *are then denoted* **antipodal points**.

We now consider the more involved cases of a curve and a surface (in Section 3.1) and two surfaces in space (in Section 3.2), in $\mathbb{R}^3$, while we also portray the necessary steps for these constraints in $\mathbb{R}^n$.

3.1 Hausdorff Distance Between a Curve and a Surface

In order to further extend the ability to compute the Hausdorff distance and support it between a curve C and a surface S, similar events to those presented in Section 2 should first be extended to $\mathbb{R}^3$. If S is open, all its boundary corner points and boundary curves should be examined against C as space point-point, point-curve, and curve-curve Hausdorff distances cases. However, we also need to consider a new type of a Hausdorff event between a space curve, $C(t)$, and a freeform surface, $S(u, v)$, in $\mathbb{R}^3$.

An equivalent condition to the antipodal curve-curve event, following Definition 4, can be expressed by requiring that the line through $C(t)$ and $S(u, v)$

be indeed a bi-normal-line and reside in the normal space of C and the normal space of S. Algebraically, we have,

$$\begin{aligned} \langle S(u,v) - C(t), C'(t) \rangle &= 0, \\ \left\langle S(u,v) - C(t), \frac{\partial S(u,v)}{\partial u} \right\rangle &= 0, \\ \left\langle S(u,v) - C(t), \frac{\partial S(u,v)}{\partial v} \right\rangle &= 0, \end{aligned} \tag{5}$$

having three constraints in three unknowns.

Extending Constraint (5) to $I\!R^n$ between parametric manifolds $F(\mathbf{u})$, $\mathbf{u} = (u_1 \cdots u_m)$ and $G(\mathbf{v})$, $\mathbf{v} = (v_1 \cdots v_n)$ is fairly straight forward having m orthogonality constraints of the form $\frac{\partial F}{\partial u_i} = 0$ and n orthogonality constraints of the form $\frac{\partial G}{\partial v_j} = 0$, in $m + n$ degrees of freedom, to solve for.

Similarly, we are required to extend the events resulting from intersecting one shape with the self-bisector of the other. Considering the self-bisector of C (parametrized twice by independent parameters t and s) yields,

$$\begin{aligned} &\langle S(u,v) - C(t), S(u,v) - C(t) \rangle \\ &\quad - \langle S(u,v) - C(s), S(u,v) - C(s) \rangle = 0, \\ &\qquad\qquad \langle S(u,v) - C(s), C'(s) \rangle = 0, \\ &\qquad\qquad \langle S(u,v) - C(t), C'(t) \rangle = 0, \end{aligned} \tag{6}$$

having three equations and four unknowns. Indeed, this should not come as a surprise as the self-bisector sheet of C in $I\!R^3$ is a bivariate surface and its intersection with S yields the univariate solution space that Equation (6) seeks. Let $N_S(u,v)$ be a normal field of $S(u,v)$. Interested in the extreme distances only along this univariate, Equation (6) could, for example, be augmented with the extreme distance condition, that occurs when the three vectors of $C(t) - C(s)$, $\frac{C(t)+C(s)}{2} - S(u,v) = \frac{1}{2}(C(t) + C(s) - 2S(u,v))$, and $N_S(u,v)$, are all coplanar, or,

$$\langle (C(t) - C(s)) \times (C(t) + C(s) - 2S(u,v)), N_S(u,v) \rangle = 0. \tag{7}$$

Figure 3 shows this special case, with this augmented constraint.

The first constraint in Equation (6) could be rewritten as

$$\langle C(t) - C(s), C(t) + C(s) - 2S(u,v) \rangle = 0, \tag{8}$$

clearly hinting once more to the fact that the term $(t-s)$ exists in this constraint as well. By subdividing C at the locations of maximum curvature, or better yet, algebraically eliminating the $(t-s)$ term from Equations (6) altogether, we avoid the need to introduce an additional parameter, as in [1].

Now consider the intersection of the self-bisector of S (parametrized independently twice as $S(u,v)$ and $S(r,s)$) with $C(t)$, to yield,

$$\langle S(u,v) - C(t), S(u,v) - C(t) \rangle$$

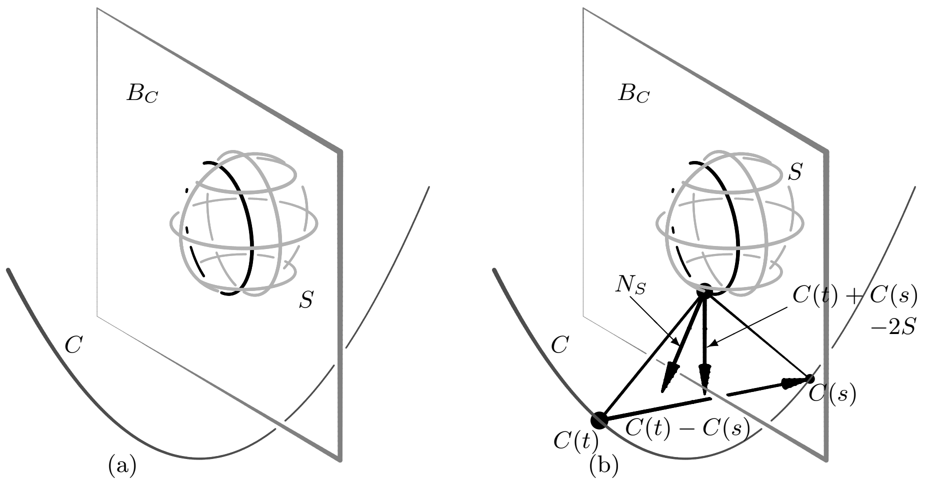

Fig. 3. The Hausdorff distance event between a surface S and a curve C can occur at the intersection of S with the self-bisector of C, B_C. While this intersection is a (black) curve in (a), we only seek the extreme distances along this curve, using, for example, a condition that occurs when the three vectors of $C(t) - C(s)$, $C(t) + C(s) - 2S$, and N_S, the normal of S, are all coplanar (b).

$$\begin{aligned}
\langle S(r,s) - C(t), S(r,s) - C(t) \rangle &= 0, \\
\left\langle S(u,v) - C(t), \frac{\partial S(u,v)}{\partial u} \right\rangle &= 0, \\
\left\langle S(u,v) - C(t), \frac{\partial S(u,v)}{\partial v} \right\rangle &= 0, \\
\left\langle S(r,s) - C(t), \frac{\partial S(r,s)}{\partial r} \right\rangle &= 0, \\
\left\langle S(r,s) - C(t), \frac{\partial S(r,s)}{\partial s} \right\rangle &= 0, \qquad (9)
\end{aligned}$$

having five equations and five unknowns.

Unfortunately, Equations (9) form, again, a singular set of constraints as every location for which $u = r$ and $v = s$ is identically satisfying the first constraint in Equations (9). The solution is again to eliminate the terms $(u = r)$ and $(v = s)$, a process that is beyond the scope of this paper. See [4] for more on this algebraic decomposition.

3.2 Hausdorff Distance Between Two Surfaces

Continuing to the most general case of the Hausdorff distance in $I\!R^3$ between two different surfaces, $S(u,v)$ and $R(r,s)$, we now need to consider the computation of bi-normal-lines and detect all antipodal locations between these surfaces,

$$\begin{aligned}
\left\langle S(u,v) - R(r,s), \frac{\partial S(u,v)}{\partial u} \right\rangle &= 0, \\
\left\langle S(u,v) - R(r,s), \frac{\partial S(u,v)}{\partial v} \right\rangle &= 0, \\
\left\langle S(r,s) - R(r,s), \frac{\partial R(r,s)}{\partial r} \right\rangle &= 0, \\
\left\langle S(r,s) - R(r,s), \frac{\partial R(r,s)}{\partial s} \right\rangle &= 0, \qquad (10)
\end{aligned}$$

having four equations and four unknowns.

Considering the self bisector of one surface, say R (parametrized as $R(r,s)$ and $R(a,b)$), against the other surface S would again yield a univariate solution as the intersection of one (self-bisector of R) surface with another (S). Interested in the extreme distance only, we once more augment this set of constraints with an extreme distance constraint, having in all,

$$\begin{aligned}
\langle S(u,v) - R(r,s), S(u,v) - R(r,s)) \rangle \\
\langle S(u,v) - R(a,b), S(u,v) - R(a,b) \rangle &= 0, \\
\left\langle S(u,v) - R(a,b), \frac{\partial R(a,b)}{\partial a} \right\rangle &= 0, \\
\left\langle S(u,v) - R(a,b), \frac{\partial R(a,b)}{\partial b} \right\rangle &= 0, \\
\left\langle S(u,v) - R(r,s), \frac{\partial R(r,s)}{\partial r} \right\rangle &= 0, \\
\left\langle S(u,v) - R(r,s), \frac{\partial R(r,s)}{\partial s} \right\rangle &= 0, \qquad (11)
\end{aligned}$$

and one possible co-planarity extreme distance constraint to fully constraint the system of equations, following Equation (7), of

$$\langle (R(a,b) - R(r,s)) \times (R(a,b) + R(r,s) - 2S(u,v)), N_S(u,v) \rangle = 0,$$

having six equations and six unknowns, in all.

4 Minimal Distance Between Curves and Surfaces

Having all this machinery we developed so far, it can also be used to determine the minimal distance between two curves or surfaces in $I\!R^2$ or $I\!R^3$. The minimal distance events could occur at either the boundaries (end points for curves, boundary curves and corner points for surfaces) or at the interior of the domain at antipodal locations. Since we have already seeing how to compute these events, we can deduce the minimal distances as well. Note that the self-bisector event is not relevant here.

In the next section, we presents some examples of both the Hausdorff distance computation and the minimal distance testing.

5 Examples

In this section, we present a few examples of the implemented-so-far portion of the computation of distances portrayed in Sections 2, 3 and 4. We present results of deriving the Hausdorff distance and minimal distance between curves in $\mathbb{R}^2$ and $\mathbb{R}^3$.

Figures 4 to 7 presents four examples of increasing complexity starting from an approximation of a sine function in the plane (Figure 4), a circular function in $\mathbb{R}^3$ (Figure 5), a helical function in $\mathbb{R}^3$ (Figure 6) and a general space curve (Figure 7). All curves are B-spline curves of degrees 3 or 4. The Hausdorff distance computation times are between a few seconds to several dozens seconds for the most complex example of Figure 7, on a modern PC workstation. The minimal distance computation took a small fraction of that.

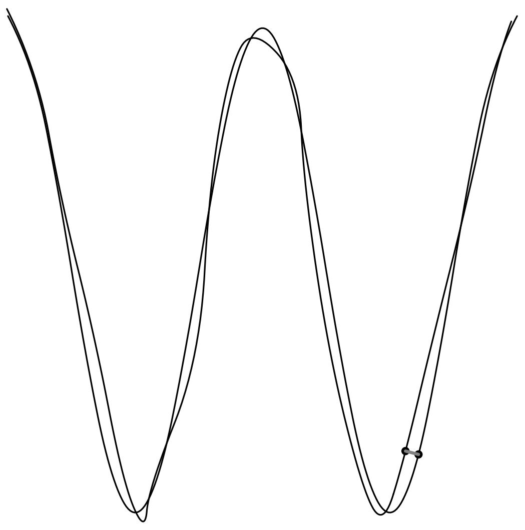

Fig. 4. The Hausdorff distance between a polynomial sine function approximation and a perturbed sine function, in the plane. Both curves are quadratic with 21 control points.

Figures 8 to 10 presents computations of minimal distances between two shapes, for the same pairs of curves as in Figures 5 to 7, for completion. In all cases, the minimal distance is *not* zero.

6 Extensions and Computational Comments

We have derived conditions for the computations of the events where the Hausdorff distance and/or minimal distance between two regular C^1 freeform para-

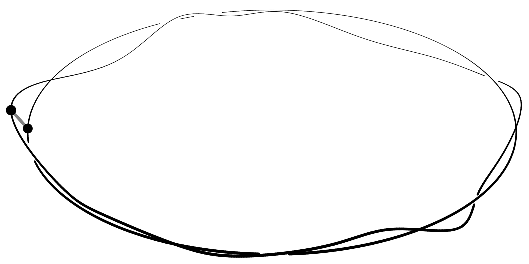

Fig. 5. The Hausdorff distance between a polynomial circular function approximation and a perturbed circular function, in $\mathbb{R}^3$. Both curves are cubic with 10 and 24 control points, respectively.

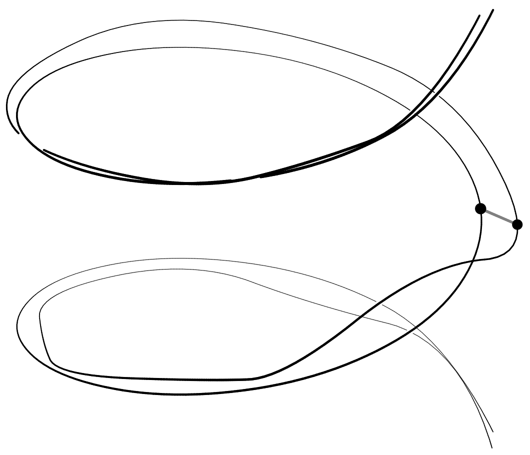

Fig. 6. The Hausdorff distance between a polynomial helical function approximation and a perturbed helical function. Both curves are quadratic with 21 control points.

metric shapes in $\mathbb{R}^2$ and $\mathbb{R}^3$ can take place. So far, we have implemented and tested all the cases for curves in $\mathbb{R}^2$ and $\mathbb{R}^3$, as was demonstrated in Section 5.

The presented constraints, even for curves, impose major computational burdens, when attempting to solve them. Consider a pair of curves, each with $O(n)$ coefficients. The addition/subtraction and/or product operations between this pair of curves, in all presented constraints, are typically derived as outer (tensor) products. Hence, any constraint that holds only a pair of independent curves (i.e. Constraint (2)) will possess $O(n^2)$ coefficients whereas a constraint involving three independent curves (i.e. Constraint (4)) will contain $O(n^3)$ coefficients.

In general, having k independent parametric forms, would yield constraints with $O(n^k)$ coefficients. This exponential growth renders this tensor product representation futile, when more than a few independent variables are involved. While beyond this writeup, we are working on an approach that reduces this exponential complexity from $O(n^k)$ to $O(np)$, where p is the number of operators (i.e. addition, subtraction, or product) in the constraint. p is typically small

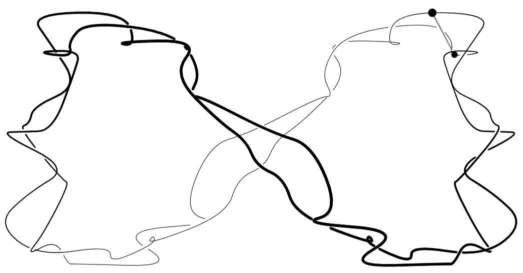

Fig. 7. The Hausdorff distance between two similar yet general space curves. Both curves are quadratic with 53 control points.

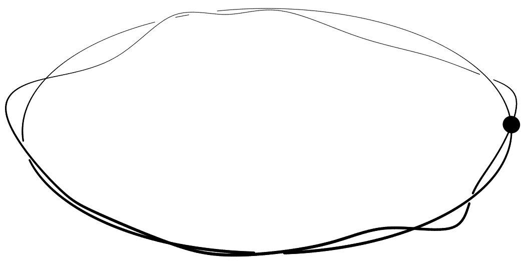

Fig. 8. The minimal distance between a polynomial circular function approximation and a perturbed circular function, in $I\!R^3$. Both curves are cubic with 10 and 24 control points, respectively. See also Figure 5.

and in the order of k. To get a hint at the expected benefit from using this approach, both in speedup and in memory consumption reduction, the set of Equations (3) was solved in both the traditional, tensor product, way and using the new approach, for a few curves of different sizes. Table 1 summarizes the result. Using up to 1470 Mbytes when solving for the Hausdorff distance between two curves of 50 coefficients, the traditional, tensor product, approach converts the first constraint in Equations (3) into a tensor product trivariate of $O(50^3)$ coefficients. Due to the fact that the constraint also involves (inner) products, it ends up with around one million coefficients. With 8 bytes per double, a single constraint will consume around 10 Mbytes of memory!

7 Conclusions

In this paper, we have presented algorithms to derive the precise Hausdorff and minimal distance between regular C^1 freeform shapes in $I\!R^2$ and $I\!R^3$. We hope to continue and completely implement all cases in $I\!R^3$, including surfaces, in the future.

Clearly some of the posed constraints, like in Equations (1), (2) and (3), could be extended with ease to $I\!R^n$. Others, like (6) and (7), are more difficult to extend. One can expect that the way Equations (6) (Equations (11)) could

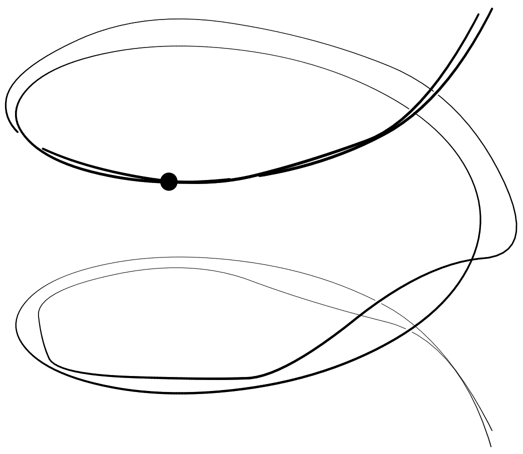

Fig. 9. The minimal distance between a polynomial helical function approximation and a perturbed helical function. Both curves are quadratic with 21 control points. See also Figure 6.

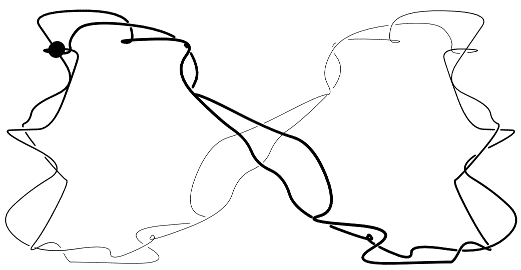

Fig. 10. The minimal distance between two similar yet general space curves. Both curves are quadratic with 53 control points. See also Figure 7.

	Traditional		*New Approach*	
Num. of Coeffs.	Time (Secs.)	Size (MB.)	Time (Secs.)	Size (MB.)
10	2.1	14.5	1.8	7
20	8.4	77	2.9	7.5
50	326	1470	23	11.5

Table 1. A comparison of solving Equations (3) for different pairs of curves, of different sizes, using the traditional, tensor product, approach and the new approach. Benefits are clearly significant in both speedup and memory consumption, as the complexity is increased.

be augmented with an extreme condition (7) (Equations (12)) to yield a zero dimensional solution space could also be applied to higher dimensions as well.

In all the above constraints, the implicit assumption was that the shapes do not (self-) interest. Many of the presented constraints vanish at an intersection, and hence, an implicit preprocessing step to all the above computation should examine for intersections first and preclude these intersection locations from the computation.

The overall computation is not fast. The need to solve for the simultaneous zeros of several piecewise polynomials of many coefficients, makes the computations consumes seconds of processing, even for curves. Further methods to make this computation more efficient are to be sought.

8 Acknowledgment

This research was partly supported by the New York metropolitan research fund, Technion, in part by the Israel Science Foundation (grants no. 346/07) and in part by the Israeli Ministry of Science Grant No. 3-4642.

References

1. H. Alt and L. Scharf. Computing the Hausdorff distance between sets of curves. Proceedings of the 20th European Workshop on Computational Geometry (EWCG), Seville, Spain, Pages 233 - 236, 2004.
2. P. Cignoni, C. Rocchini, R. Scopigno. Metro: Measuring error on simplified surfaces. Computer Graphics Forum, 17(2):167-174, 1998.
3. G. Elber and M. S. Kim. Geometric constraint solver using multivariate rational spline functions. *Proc. of the ACM symposium on Solid Modeling and Applications*, 1–10, 2001.
4. G. Elber, T. Grandine and M. S. Kim. Surface Self-Intersection Computation via Algebraic Decomposition. functions. Submitted for publication in SPM07.
5. I. Hanniel and G. Elber. Subdivision Termination Criteria in Subdivision Multivariate Solvers. Computer Aided Design, Vol 39, pp 369-378, 2007.
6. *IRIT 9.5 User's Manual*, 2005, Technion. http://www.cs.technion.ac.il/∼irit.
7. D. Johnson. Minimum distance queries for haptic rendering. PhD thesis, Computer Science Department, University of Utah, 2005.
8. B. Juttler. Bounding the Hausdorff Distance of Implicitely Defined and/or parametric Curves. Matematical Methods in CAGD, Tom Lyche and Larry L. Schumaker (eds.), pp 1-10, Oslo 2000.
9. D. Pekerman, G. Elber, and M.-S. Kim. Self-Intersection Detection and Elimination in freeform Curves. Accepted for publication in Computer Aided Design.
10. SYNAPS (SYmbolic Numeric ApplicationS). http://www-sop.inria.fr/galaad/software/synaps.