Computing the Minimum Enclosing Sphere of Free-form Hypersurfaces in Arbitrary Dimensions

Ramanathan Muthuganapathy∗ † Gershon Elber† Gill Barequet†
Myung-Soo Kim‡

Abstract

The problem of computing the minimum enclosing sphere (MES) of a point set is a classical problem in Computational Geometry. As an LP-type problem, its expected running time on the average is linear in the number of points. In this paper, we generalize this approach to compute the minimum enclosing sphere of free-form hypersurfaces, in arbitrary dimensions. This paper makes the bridge between discrete point sets (for which indeed the results are well-known) and continuous curves and surfaces, showing that the general solution for the former can be adapted for the latter.

To compute the MES of a pair of hypersurfaces, each one having a contact point (a point at which the sphere touches the hypersurface), antipodal constraints are employed. For more than a pair, equidistance constraints along with tangency constraints are applied. These constraints yield a finite set of solution points which is used to identify the minimum enclosing sphere. The algorithm uses the LP-characteristic of the problem to process the input set. Furthermore, an optimization procedure that uses the convex hull of sampled points from the hypersurfaces is also described. Finally, results from our implementation are presented.

Keywords: minimum enclosing circle, minimum enclosing sphere, minimum enclosing sphere, LP-type problems, free-form hypersurfaces.

1 Introduction

Computing the minimum spanning/enclosing circle of a given set of points in the plane is considered a classic problem in computational geometry [4]. (The original reference goes back to [14].) The optimal solution (a minimum-radius circle) can be found in Θ(n) expected time, where n is the number of points. In fact, the minimum enclosing sphere of n points in any dimension can be found in expected Θ(n) time by using the same method. Sharir and Welzl [13] further generalized the technique and introduced the notion of LP-type (linear-programming-like) problems. In a nutshell, one can apply the LP-technique to an optimization problem in which adding a new constraint C does not change the solution P, or otherwise the new solution P′ is partially defined by C, thereby reducing...
the number of possible configurations of constraints defining $P'$. Amenta [2] later showed that special properties of these problems imply so-called Helly-type theorems. Fischer et al. [8, 9] developed an algorithm for computing the minimum enclosing ball of a set of $N$ balls in a $d$-dimensional space using the LP technique, while starting by explicitly addressing the small cases ($N \leq d + 1$). They appear to require rational arithmetic in their implementation. A deterministic algorithm for the smallest enclosing ball of balls has been provided in [11, 12].

It has been shown that the computation of the minimum enclosing circle of a set of smooth planar curves is possible to compute in expected linear time as it is also an LP-type problem [3] (by smooth curves we mean piecewise $C^1$ polynomial or rational curves of bounded degree.). This type of problem usually arises when there is a need to design efficient hierarchical spatial data structures for storing a large amount of geometric objects. These data structures can then support fast clipping, searching, collision detection, ray tracing, and other similar query-operations. For example, the sphere tree in [10] supports a hierarchical modeling of complex 3-dimensional scenes using bounding spheres of various 3-dimensional objects, and it has also been extensively used in 3-dimensional computer games [1, 5].

In this paper, we show that the generalized problem, in which one aims to find the minimum enclosing sphere in $d$ dimensions of an arbitrary set of free-form hypersurfaces, is possible. Amenta’s thesis [2] talks about relating Helly theorems with generalized LP and convex programming methods and gave some application algorithms on halfspaces, points (which can possibly be extended to compact sets), polygons/polytopes etc (refer [2, §9.1 to 11.3, pp. 64–94]). Though our work can be considered as based on the Helly theorems, we have also provided a methodology for computing the MES of a set of closed curves/surfaces. Typically, these problems are solved for point-sets/circles/ellipses. The purpose of this paper is to show that the discrete approach (for point sets) can be adapted to the continuous case, with the appropriate modifications and additional tools. The main contribution of the paper is, we have shown that this problem can be solved (through the multivariate constraint solver of Elber and Kim [7]) even if the input is of freeform curves/surfaces defined as Bezier/B-Spline through an algebraic technique. Moreover, the tangent and equidistance conditions can be generalized for any dimension.

1.1 Basic Idea

Given a set of $N$ closed $C^1$ continuous hypersurfaces $H_1, H_2, ..., H_N$, where a hypersurface can be a point, curve, surface, or a freeform entity in more than two parameters in $R^d$, the problem is to compute the minimum enclosing sphere (MES) of the set. Each $H_i = H_i(u_{i1}, \ldots, u_{in})$, where $u_{i1}, \ldots, u_{in}$ are the parameters representing the hypersurface, with $n \leq d - 1$. The MES is a hypersphere that encloses the given set and is of minimum radius. For example, in 2D, the input set
can consist of points and free-form curves (represented as $H_1(u_{11}), H_2(u_{21}), \ldots$), and the problem is to compute the minimum enclosing circle (Figure 1). For practical purposes, our implementation assumes that the MES is tangent to any hypersurface in at most one point. We explain in Section 6 how to avoid this assumption. Moreover, the hypersurfaces are assumed to be in general position i.e., they overlap neither partially nor completely. In addition, it is assumed that no “spherical” portions of any hypersurface lie completely on the MES.

The algorithm always starts with two input hypersurfaces. It is shown that, for the MES of two hypersurfaces, the points on the hypersurfaces touching the MES have to be antipodal, i.e., they lie in exactly opposite direction to each other with respect to the center of the MES. Based on the fact that the points have to be antipodal, constraint equations are then formulated. The finite set of solutions from these constraints are then used to compute the MES of the two hypersurfaces.

The rest of the input set is considered one by one (which is typical in an LP-type algorithm). Each hypersurface is tested for its containment in the current MES. When three or more hypersurfaces are involved, the algorithm uses the fact that their touching points have to be equidistant from the center of the sphere, and the sphere has to be tangent to all of them. Hence, the constraints are termed as equidistance and tangency constraints, respectively. Again, examining the finite set of solutions from these constraints, the MES can then be computed.

It is a well-known fact that, when the input set is a point set, it is sufficient to use the points on the boundary of the convex hull (hereafter, convex hull implies the boundary of the convex hull) of the input set to compute its MES. An optimization procedure uses this fact to eliminate hypersurfaces
that lie in the interior of the convex hull of points sampled from the input data. It is also shown that such an optimization procedure is indeed correct and speeds up the algorithm.

Our implementation presented here uses IRIT [6], and its multivariate piecewise-rational solver [7], to solve sets of algebraic constraints. This paper is an extension of [3], a work that deals with the computation of the minimum enclosing circle for inputs consisting only of planar curves.

The rest of the paper is organized as follows. The equidistance and tangency constraints for \(d + 1\) hypersurfaces in \(\mathbb{R}^d\) are provided in Section 2. A description of constraints for less than \(d + 1\) hypersurfaces in \(\mathbb{R}^d\) is described next. In particular, Section 3.1 describes the constraints for two hypersurfaces and Section 3.2 describes the constraints for triplets in three dimensions. The computation of the MES using the solution of the constraints is detailed in Section 4. Section 5 proves the LP characteristics of the problem. Details of the algorithm are described next in Section 6. An optimization procedure which uses the convex hull of the input is described in Section 7. The complexity of the algorithm is analyzed in Section 8. Experimental results are provided in Section 9, and we terminate with some concluding remarks in Section 10.

2 Equidistance and Tangency Constraints for \(d + 1\) Hypersurfaces in \(\mathbb{R}^d\)

Let \(H_1, H_2, \ldots, H_{d+1}\) be \(d + 1\) closed \(C^1\)-continuous hypersurfaces in \(\mathbb{R}^d\) (please note that \(H_i = H_i(u_{i1}, \ldots, u_{in})\), where \(u_{i1}, \ldots, u_{in}\) are the parameters and \(n \leq d - 1\)). Since the center of the MES, \(C = (C_1, C_2, \ldots, C_d)\), has to be equidistant from the touching points of all the hypersurfaces (we assume, for now, that the MES is indeed tangent to all \(H_i, i = 1, \ldots, d + 1\)), the equidistance constraints can be formulated as follows:

\[
\begin{align*}
\langle C - H_1, C - H_1 \rangle &= \langle C - H_2, C - H_2 \rangle, \\
\langle C - H_1, C - H_1 \rangle &= \langle C - H_3, C - H_3 \rangle, \\
&\vdots \\
\langle C - H_1, C - H_1 \rangle &= \langle C - H_{d+1}, C - H_{d+1} \rangle.
\end{align*}
\]

Equations (1) are equivalent to the following linear equations in \(C = (C_1, C_2, \ldots, C_d)\):

\[
2 \langle C, H_1 - H_2 \rangle = ||H_1||^2 - ||H_2||^2, \\
2 \langle C, H_1 - H_3 \rangle = ||H_1||^2 - ||H_3||^2, \\
&\vdots \\
2 \langle C, H_1 - H_{d+1} \rangle = ||H_1||^2 - ||H_{d+1}||^2.
\]

Noting that the MES also has to be tangent to the \(C^1\)-continuous closed hypersurfaces it touches,
Table 1: DOF vs. the number of constraints for the case of $d+1$ hypersurfaces in $R^d$, where $D_i$ is the dimension of the hypersurface $H_i, i = 1, ..., d+1$.

<table>
<thead>
<tr>
<th>Degree of Freedom (DOF)</th>
<th>Number of Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>$d$ coefficients of center of MES</td>
<td>$d$ equidistance constraints</td>
</tr>
<tr>
<td>$\sum D_i$</td>
<td>$\sum D_i$ tangency constraints</td>
</tr>
</tbody>
</table>

The tangency constraints can be written as follows:

$$\langle C - H_1, dH_1 \rangle = 0,$$

$$\langle C - H_2, dH_2 \rangle = 0,$$

$$\vdots$$

$$\langle C - H_{d+1}, dH_{d+1} \rangle = 0.$$ (2)

where $dH_i$ denotes a basis of vectors that spans the tangent vectors of $H_i$. Solving the equidistance and tangency constraints results in a finite set of solutions for $C$ which can be represented using the parameters of the hypersurfaces. It is easy to see that this set of constraints is well-defined and results in a zero dimensional solution space. It should be noted that, though the closed $C^1$-continuous hypersurfaces might have singular locations because of parameterization, we do assume regularity of the constraints when solving for their zeros. Nonetheless, the probability of a tangency/contact at a singularity, in a general arrangement, is close to zero. Practically, these cases could be dealt with as special cases.

Let $D_i = \dim(H_i)$ be the dimension of each hypersurface. First, the equidistance constraints (1) result in $d$ equations. Second, the number of tangency constraints (2) is $\sum D_i$, the total sum of the dimensions of the hypersurfaces. On the other hand, we have $d$ unknowns of the center $C$ of the sphere and $\sum D_i$ unknowns in all $H_i$ hypersurfaces. Table 1 shows the degrees of freedom available versus the number of constraints, $d$ and $\sum D_i$.

The following example shows how the equidistance and tangency constraints can be employed in practice.

**Example 2.1 Equidistance and Tangency Constraints for Three Curves in $R^2$**

Given three planar curves $H_1(u_{11}) (= C_1(t)), H_2(u_{21}) (= C_2(r))$, and $H_3(u_{31}) (= C_3(s))$, their equidistance constraints from the center of the MES, $C = (C_x, C_y)$ (see Figure 2 and Equations (1)) can be written as:

$$\langle C - C_1(t), C - C_1(t) \rangle = \langle C - C_2(r), C - C_2(r) \rangle,$$

$$\langle C - C_1(t), C - C_1(t) \rangle = \langle C - C_3(s), C - C_3(s) \rangle.$$ (3)
Figure 2: Constraints for three free-form planar curves in $\mathbb{R}^2$.

Equations (3) are equivalent to the following linear equations in $C = (C_x, C_y)$:

$$
2 \langle C, C_1(t) - C_2(r) \rangle = ||C_1(t)||^2 - ||C_2(r)||^2,
$$

$$
2 \langle C, C_1(t) - C_3(s) \rangle = ||C_1(t)||^2 - ||C_3(s)||^2.
$$

The tangency constraints (see Equation (2)) for the three curves can be written as:

$$
\langle C - C_1(t), C_1'(t) \rangle = 0,
$$

$$
\langle C - C_2(r), C_2'(r) \rangle = 0,
$$

$$
\langle C - C_3(s), C_3'(s) \rangle = 0.
$$

The equidistance constraints (Equations (4)) and tangency constraints (Equations (5)) for three curves in $\mathbb{R}^2$ result in five equations, in as many unknowns, $t$, $r$, $s$, and $C = (C_x, C_y)$.

3 Constraints for Less than $d + 1$ hypersurfaces (having less than $d + 1$ contact points) in $d$ Dimensions

When we have less than $d+1$ hypersurfaces in $d$ dimensions, the equidistance and tangency constraints toward the MES result in an under-constrained problem. For example, for two curves in 2D, there is one equidistance constraint and two tangency constraints, resulting in three equations in four unknowns. Yet, clearly there is a unique MES.
3.1 Antipodality Constraints for Two hypersurfaces

Lemma 1 A sphere, centered at $C$ in $\mathbb{R}^d$ enclosing two different hypersurfaces (each having a contact point) is minimum in radius if and only if the normals of the corresponding contact points are opposite to each other with respect to $C$ (i.e., antipodal.)

Proof: By contradiction. Assume that the unit vectors from the center $C$ to the contact points on the hypersurfaces are $V_1$ and $V_2$ and are not antipodal. There exists a small distance $\varepsilon > 0$ so that the sphere can be translated in the direction of $V_1 + V_2$, then the sphere has no contact point with the two hypersurfaces though it still bounds them. We can then shrink the sphere slightly so as to find an enclosing sphere which is smaller in radius. Hence, $V_1$ and $V_2$ have to be antipodal. $\Box$

The above antipodality constraint can be formulated in terms of equations. Let $H_1$ and $H_2$ be the two hypersurfaces. The antipodality constraint can be written as (noting that: $H_1 + H_2 = C$):

\[
\langle dH_1, H_1 - \frac{H_1 + H_2}{2} \rangle = 0, \\
\langle dH_2, H_2 - \frac{H_1 + H_2}{2} \rangle = 0,
\]

or, equivalently,

\[
\langle dH_1, H_1 - H_2 \rangle = 0, \\
\langle dH_2, H_2 - H_1 \rangle = 0.
\]

(6)

Equations (6) represent the antipodality constraint in any dimension $d$. Again, $dH_i$ denotes a basis of vectors that spans the tangent vectors of $H_i$ and is not necessarily prescribing a single constraint. The following two examples show the formulation of antipodal equations for two curves in $\mathbb{R}^2$ and two surfaces in $\mathbb{R}^3$, respectively.

Example 3.1.1 Antipodal constraint of two curves in $\mathbb{R}^2$

Consider two planar closed $C^1$-continuous curves $H_1(u_{11}) (= C_1(t))$ and $H_2(u_{21}) (= C_2(r))$. The antipodality constraint (Equations (6)) is:

\[
\left\langle C'_1(t), C_1(t) - \frac{C_1(t) + C_2(r)}{2} \right\rangle = 0, \\
\left\langle C'_2(r), C_2(r) - \frac{C_1(t) + C_2(r)}{2} \right\rangle = 0.
\]

(7)

Clearly, the two equations have two unknowns ($t$ and $r$). Figure 3 illustrates the antipodality constraint for two planar curves.

Example 3.1.2 Antipodality constraint of two surfaces in $\mathbb{R}^3$
Figure 3: Antipodality constraint for two free-form planar curves in $\mathbb{R}^2$.

Figure 4: Antipodality constraint for two free-form surfaces in $\mathbb{R}^3$. 
Consider two closed $C^1$-continuous surfaces $H_1(u_{11}, u_{12})$ ($= S_1(u_1, v_1)$) and $H_2(u_{21}, u_{22})$ ($= S_2(u_2, v_2)$) (Figure 4). Then, each of the antipodality constraint is represented by two equations.

\[
\langle \frac{\partial S_1(u_1, v_1)}{\partial u_1}, S_1(u_1, v_1) - \frac{S_1(u_1, v_1) + S_2(u_2, v_2)}{2} \rangle = 0,
\]

\[
\langle \frac{\partial S_1(u_1, v_1)}{\partial v_1}, S_1(u_1, v_1) - \frac{S_1(u_1, v_1) + S_2(u_2, v_2)}{2} \rangle = 0,
\]

\[
\langle \frac{\partial S_2(u_2, v_2)}{\partial u_2}, S_2(u_2, v_2) - \frac{S_1(u_1, v_1) + S_2(u_2, v_2)}{2} \rangle = 0,
\]

\[
\langle \frac{\partial S_2(u_2, v_2)}{\partial v_2}, S_2(u_2, v_2) - \frac{S_1(u_1, v_1) + S_2(u_2, v_2)}{2} \rangle = 0,
\]

Thus, in (8) we have four equations in four unknowns $(u_1, v_1, u_2$ and $v_2)$ in all. Note that in the antipodal cases, $C$ is easily deduced as $(S_1 + S_2)/2$. For the antipodal condition, the major degeneracy appears to be when the hypersurfaces having hyperplane portions that are parallel to each other. For example, when the input involves curves, this translates to having straight line portions in the curves. The MES though will have antipodal points that do not lie on the straight line portions.

### 3.2 Constraints for Triplets in Three and Higher Dimensions

Consider $H_1$, $H_2$, and $H_3$ (Figure 5). The equidistance constraints:

\[
\langle C - H_1, C - H_1 \rangle = \langle C - H_2, C - H_2 \rangle,
\]

\[
\langle C - H_1, C - H_1 \rangle = \langle C - H_3, C - H_3 \rangle.
\]
The corresponding tangency constraints are:

\[
\begin{align*}
\langle \mathbf{C} - H_1, dH_1 \rangle &= 0, \\
\langle \mathbf{C} - H_2, dH_2 \rangle &= 0, \\
\langle \mathbf{C} - H_3, dH_3 \rangle &= 0.
\end{align*}
\]

Let us examine the set of constraints assuming that the given hypersurfaces are surfaces (in two parameters). Clearly, the three tangency Equations (10) become six equations, taking into account the two parameters of each surface. This results in a total of eight equations (two equidistance constraints and six tangency constraints) in nine unknowns (the center point for \( \mathbf{C} \) and the parameters of each surface), which is an under-constrained problem. The remaining equation comes from the following lemma.

**Lemma 2** A sphere enclosing three different hypersurfaces (each having a contact point) in \( \mathbb{R}^3 \) is minimum in radius if and only if the vectors from the center of the sphere to the contact points on the hypersurfaces lie in a plane, i.e., the vectors are coplanar.

**Proof:** By contradiction. Let the vectors from the center \( \mathbf{C} \) of the MES to the contact points on the hypersurfaces be \( V_1, V_2 \) and \( V_3, ||V_i|| > 0 \), and assume they are not coplanar and let \( L \) denote the plane determined by the three contact points. (Note that the center \( \mathbf{C} \) is not contained in the plane \( L \).) There exists a small distance \( \varepsilon > 0 \) so that the sphere can be translated toward the plane \( L \) along the normal direction of the plane. The sphere will have no contact point with the three hypersurfaces though it still bounds them. We can then shrink the sphere so as to find a smaller enclosing sphere. Hence, the vectors have to lie on a plane. \( \square \)

The above is the generalization of the two hypersurfaces’ antipodality constraint (Lemma 1).

**Corollary 1** The three contact points of the MES of the three hypersurfaces in \( \mathbb{R}^3 \) define a great circle on the MES.

As the center \( \mathbf{C} \) and the contact points of the MES (which is a sphere in \( \mathbb{R}^3 \)) are on a plane (by Lemma 2, Corollary 1 follows).

Writing the coplanarity constraint in Lemma 2 as a ninth constraint results in

\[
\langle (\mathbf{C} - H_1) \times (\mathbf{C} - H_2), (\mathbf{C} - H_3) \rangle = 0.
\]

Equations (9), (10) and (11) completely determine \( \mathbf{C} \) in \( \mathbb{R}^3 \) and the parameters of the hypersurfaces, having nine equations in nine unknowns.
For $m$ hypersurfaces in $R^d$ and $m < d + 1$, the generalization is obtained from the following lemma (Lemma 3).

**Lemma 3** The contact points of the MES of $m$ different hypersurfaces in $R^d$, $m < d + 1$, are all on a sphere of dimension $m - 1$.

**Proof:** Consider Equations (9) and (10). The tangency Equations (10) determine the parameters of the hypersurfaces (see Table 1). The dimension of the sphere or the number of coordinates of $C$ (the sphere defined by the center $C$ and the radius defined by the center $C$ and a contact point) depend on Equations (9). If there are $m$ hypersurfaces, then there are $m - 1$ equidistance constraints. This implies that there are $m - 1$ coordinates for the sphere and hence it is of dimension $m - 1$. This sphere is also minimum in radius, otherwise, it violates Corollary 1 (though Corollary 1 is for $R^3$, the generalization to $R^d$ is rather straightforward).

4 Computing the MES

The solution of the constraint equations (detailed in Sections 2 and 3) results in a finite set of candidates. Each solution defines a sphere which is tangent to all the hypersurfaces, although some of the tangencies may be to the outside of the sphere as is clearly undesired. For example, in Figure 6, the solutions $B_1$, $B_2$, and $B_3$ are incorrect, while we seek the solution $B_4$. To identify the correct MES from the set of candidate solution spheres, the hypersurfaces in consideration are subjected to a containment test – of whether or not the hypersurface is contained in the sphere, i.e., all the points on the hypersurface lie within (or on the boundary of) the sphere. Obviously, only the MES will contain all the hypersurfaces.

Consider a hypersurface $H$ and a sphere $(C, R)$, centered at $C$ and of radius $R$ (see Figure 7). $H$ is inside the sphere if and only if

$$||D_m - C|| \leq R,$$

where $D_m$ is the maximum distance point of $H$ from the center point $C$. While one can indeed compute all radially-extreme locations from $H$ to $C$, the following would be examined.

$$F = ||H - C||^2 - R^2 \leq 0. \quad (12)$$

Given a piecewise rational hypersurface, $H$, Equation (12) is also rational, and hence one can examine all the coefficients of $F$. If all the coefficients are nonpositive, then, due to the convex-hull property of the NURBS representation, $F$ must be nonpositive as well. Otherwise, one needs to compute the zero set of $F$. If the zero-set is empty, then $H$ is either completely inside or completely outside the sphere, in which case a single evaluation of a point in $H$ could reveal its inclusion status.
Figure 6: Constraints yield multiple sets of solution points.

Figure 7: Containment test.
Figure 8: LP-type characteristics of the problem (see Theorem 1). $B_0 = \text{MES}(S)$ and $B_1 = \text{MES}(S \cup \{S\})$. $Q$ is an intersection point of $B_0$ and $B_1$

While this test does not affect the complexity of the algorithm, in practice it can greatly save a lot of running time.

5 LP Characteristics of the Problem

We now consider the computation of the MES of an arbitrary set of $C^1$ smooth surfaces as an LP-type problem. This is merely a generalization of Theorem 1 in [3].

Toward this end, we need to prove that given the MES of a set of surfaces, if a newly introduced surface is entirely or partially outside the current MES, then the MES of all the surfaces (including the new one) is partially defined by the new surface. In other words, the new MES must be in contact with the new surface.

More formally, we prove the following:

**Theorem 1** Let $S$ be a set of $C^1$ smooth surfaces, and let $\text{MES}(S)$ be the MES of $S$. Also, let $S \notin S$ be a newly introduced surface. Then, either

1. The surface $S$ is completely inside (possibly touching) $\text{MES}(S)$, in which case $\text{MES}(S \cup \{S\}) =$
2. Surface $S$ is partially or completely outside $\text{MES}(S)$, in which case $S$ must be in contact with $\text{MES}(S \cup \{S\})$.

Proof: The proof follows step by step the proof of Lemma 4.14 in [4], which makes the same assertion for points. Consider two possible cases:

1. $\text{MES}(S)$ fully contains $S$. Assume to the contrary that some sphere $B$ whose radius is less than that of $\text{MES}(S)$ contains $S \cup \{S\}$. In particular, $B$ contains $S$, which contradicts the minimality of $\text{MES}(S)$.

2. $S$ is partially (or fully) outside $\text{MES}(S)$. Clearly, $\text{MES}(S \cup \{S\}) \neq \text{MES}(S)$ and the two spheres must intersect; see Figure 8. Let $B_0 = \text{MES}(S)$ and $B_1 = \text{MES}(S \cup \{S\})$ with the respective center points $C_0$ and $C_1$. Denote by $Q$ some point on the intersection of the boundaries of $B_0$ and $B_1$. Define a continuous deformation between $B_0$ and $B_1$ as follows: the center of $B_t$ (for $0 \leq t \leq 1$) is $C_t = (1-t)C_0 + t(C_1)$, and its radius is $D(C_t, Q)$, where $D(\cdot, \cdot)$ is the Euclidean distance. Since $S$ is partially or fully outside $B_0$, and $S$ is fully contained in $B_1$, by continuity there must be some $0 < t^* \leq 1$ such that $B_{t^*}$ contains $S$ and touches it. The set $S$ is contained entirely in both spheres $B_0$ and $B_1$, therefore, $S$ is contained in $B_0 \cap B_1$, which, by definition, in turn is contained in $B_t$ (for any $0 \leq t \leq 1$). Also, the radius of any sphere $B_t$ (for any $0 < t < 1$) is strictly less than that of $B_1$. Therefore, we must have $t^* = 1$; otherwise, we have a contradiction to the minimality of $B_1$.

Remark: The proof of Theorem 1 holds for any dimension. The uniqueness of the MES of the set $S$ is shown by a similar argument of a continuous family of spheres. The reader is referred to [4, §4.7, Lemma 4.14] for the exact details.

6 The Algorithm

After establishing the LP characteristics of the problem, we apply the well-known machinery to solve it in an expected time that is linear in $n$, the number of hypersurfaces, on the average. We use the randomized algorithm of [4, §4.7, pp. 85–88] that originally finds the minimum enclosing circle of a planar point set. Assume for the moment that the MES of the hypersurfaces is defined by points that belong to different hypersurfaces. The algorithm (described in Figure 9) outlines the computation of the MES in three dimensions. In order to generalize it to $d$ dimensions, one needs to extend the algorithm to have $d$ levels (possibly by implementing all levels in one function).
ALGORITHM MinSphere ($S$)
Input: A set $S$ of $n$ hypersurfaces $H_1, H_2, \ldots, H_n$ in $\mathbb{R}^3$.
Output: The minimum-radius sphere that fully contains $S$.
begin
Compute a random permutation $H_1, \ldots, H_n$ of the hypersurfaces in $S$.
Let $B_0 = B_1 = B_2$ be the smallest sphere enclosing $H_1$ and $H_2$.
for $i = 3, \ldots, n$ do
if $H_i \subset B_{i-1}$
then $B_i := B_{i-1}$;
else $B_i := \text{MinSphereWithOneHS} (\{H_1, \ldots, H_{i-1}\}, H_i)$.
end if
end for
return $B_n$.
end MinSphere

FUNCTION MinSphereWithOneHS ($S, Q$)
Input: A set $S$ of $N$ hypersurfaces and a hypersurface $Q$, such that there exists an enclosing sphere of $S$ that touches $Q$.
Output: The minimum-radius sphere that fully contains $S$ and that touches $Q$.
begin
Compute a random permutation $H_1, \ldots, H_N$ of the hypersurfaces in $S$.
Let $B_0 = B_1$ be the smallest sphere enclosing $H_1$ and $Q$.
for $i = 2, \ldots, N$ do
if $H_i \subset B_{i-1}$
then $B_i := B_{i-1}$;
else $B_i := \text{MinSphereWithTwoHSs} (\{H_1, \ldots, H_{i-1}\}, Q, H_i)$.
end if
end for
return $B_n$.
end MinSphereWithOneHS

FUNCTION MinSphereWithTwoHSs ($S, Q_1, Q_2$)
Input: A set $S$ of $n$ hypersurfaces and two hypersurfaces $Q_1, Q_2$, such that there exists an enclosing sphere of $S$ that touches $Q_1$ and $Q_2$.
Output: The minimum-radius sphere that fully contains $S$ and that touches $Q_1$ and $Q_2$.
begin
Compute a random permutation $H_1, \ldots, H_n$ of the hypersurfaces in $S$.
Let $B_0$ be the smallest sphere enclosing $\{Q_1, Q_2\}$.
for $i = 1, \ldots, n$ do
if $H_i \subset B_{i-1}$
then $B_i := B_{i-1}$;
else $B_i := \text{MinSphereWithThreeHSs} (\{H_1, \ldots, H_{i-1}\}, Q_1, Q_2, H_i)$.
end if
end for
return $B_n$.
end MinSphereWithTwoHSs

FUNCTION MinSphereWithThreeHSs ($S, Q_1, Q_2, Q_3$)
Input: A set $S$ of $n$ hypersurfaces and three hypersurfaces $Q_1, Q_2, Q_3$, such that there exists an enclosing sphere of $S$ that touches $Q_1$ and $Q_2$, $Q_3$.
Output: The minimum-radius sphere that fully contains $S$ and that touches $Q_1$, $Q_2$ and $Q_3$.
begin
Compute a random permutation $H_1, \ldots, H_n$ of the hypersurfaces in $S$.
Let $B_0$ be the smallest sphere enclosing $\{Q_1, Q_2, Q_3\}$.
for $i = 1, \ldots, n$ do
if $H_i \subset B_{i-1}$
then $B_i := B_{i-1}$;
else $B_i :=$ the minimum enclosing sphere of $Q_1, Q_2, Q_3$, and $H_i$.
end if
end for
return $B_n$.
end MinSphereWithThreeHSs

Figure 9: Computing the minimum-radius enclosing sphere of a set of hypersurfaces (following closely the algorithm of [4] for enclosed points) in $\mathbb{R}^3$.
In the upper level of the algorithm, we iteratively compute the minimum enclosing sphere of the first \(i\) entities, where \(i\) goes from 2 to \(n\) (see function MinSphere in Figure 9). In the \(i\)th step, we check whether the hypersurface \(H_i\) lies inside \(B_{i-1}\), the minimum enclosing sphere of the first \(i-1\) hypersurfaces. If so, then \(B_i = B_{i-1}\). Otherwise, if \(H_i\) lies partially or entirely outside \(B_{i-1}\), we need to re-compute \(B_i\), but now it is guaranteed by Theorem 1 that \(B_i\) touches \(H_i\).

Accordingly, we now invoke a secondary function that performs the same task (namely, finding the minimum enclosing sphere of a set of hypersurfaces), with the only restriction that the sought-after sphere touches one specific hypersurface \(Q\) (see function MinSphereWithOneHS in Figure 9). Again, we add one hypersurface at a time, and check whether the newly-added hypersurface is contained in the previously-computed sphere. If this is not the case, we need to re-compute the sphere, but this time it is guaranteed that the new MES touches both \(Q\) and the newly-added hypersurface.

The algorithm then invokes the function MinSphereWithTwoHSs and possibly the function MinSphereWithThreeHSs (Figure 9), with the respective restrictions that the sought-after sphere touches three or four hypersurfaces. These two functions have the same logic as above. Note that we add one hypersurface at a time while performing these functions. At the bottom level we need the services of a function that computes the MES of four hypersurfaces.

We do not need to alter the algorithm to accommodate cases in which the MES is defined by two or more points that lie on the same entity. The invariant of Theorem 1 remains true. That is, if the \(i\)th entity lies (partially or entirely) outside the MES of the first through the \((i-1)\)st entity, then the new MES must touch the \(i\)th entity. The algorithm only needs to consider all possible multi-touching cases. For example, in the case of two points lying on the same entity, the constraint equations can be solved considering \(H_1 = H_2\). To accommodate all possible multi-touching situations, the algorithm must check all cases in increasing order of complexity. The correctness of the algorithm follows directly from the LP characteristics of the problem.

### 7 Optimization Using the Convex Hull

It is well known that, for a given point set in the plane, the MES is entirely determined by the convex hull (CH) of the point set. This is because the points defining the MES are always on the convex hull of the set.

In higher dimensions, when free-form curves, surfaces, hypersurfaces etc., are involved, an optimization procedure can be applied. Such a procedure will eliminate those hypersurfaces that do not contribute to the convex hull of the set. Since the convex hull of points in \(R^d\) is straightforward to compute, points lying on all hypersurfaces are sampled and their convex hull is computed. The optimization procedure, is the following:
Figure 10: The optimization procedure using the convex hull.

- Compute the convex hull of the sampled points (along with any input points).

- Eliminate all the hypersurfaces that are inside the convex hull. A simple test for this purpose is to examine whether or not all the control points of the hypersurface are inside the hull.

- The remaining hypersurfaces are used to compute the MES using the algorithm described in Section 6.

Figure 10 illustrates the optimization procedure. In the figure, the points sampled on the curves are shown in grey, whereas the input points are shown in black. Also shown is the convex hull of the sampled and input points. Applying the test for containment of hypersurfaces in the convex hull, all the input points and the curve $C$ are eliminated in this example. The remaining hypersurfaces are used for computing the MES.

Again, the correctness of the optimization procedure follows from the fact that $\text{CH}(\text{sampled + input points}) \subset \text{CH}(\text{the input set}) \subset \text{MES}(\text{the input set})$. 

17
8 Complexity Analysis

Denote by $N$ the number of hypersurfaces in $d$ dimensions, $d$ fixed, and by $p$ the maximum degree of the polynomials describing the hypersurfaces. Let $I(p)$ be the time needed to decide whether or not a hypersurface is fully contained in a given sphere, and $M(p)$ be the time needed to compute the MES of $d+1$ such hypersurfaces.

The main part of the algorithm, $\text{MinSphere}$, performs $N$ iterations, in each of which it either decides in $I(p)$ time whether or not the MES (computed so far) has to change. Occasionally, it also calls the function $\text{MinSphereWithOneHS}$. In the worst case, the main algorithm can call the latter $\Theta(N)$ times, in case all of the second through the $N$th hypersurfaces require an update of the MES. Similarly, the function $\text{MinSphereWithOneHS}$ performs $O(N)$ iterations. In each iteration it decides in $I(p)$ time whether or not to re-compute the MES by calling the function $\text{MinSphereWithTwoHSs}$. This also can occur in $O(N)$ iterations. Its running time is, therefore, $O(N (I(p) + M(p)))$. A similar argument holds for the functions $\text{MinSphereWithTwoHSs}$ and $\text{MinSphereWithThreeHSs}$. Overall, in the worst case, the entire algorithm requires $\Theta(N^4 (I(p) + M(p)))$ time. Note that $I(p)$ and $M(p)$, the practical running times of numerical methods, do not depend at all on $N$, but solely on $p$ and on the convergence parameters. When the latter are fixed or assumed constant, we have that both $I(p)$ and $M(p)$ are low-degree polynomials in $p$. If we further fix $p$, then $I(p)$ and $M(p)$ can be regarded as constants. Even if we take into consideration endpoints and $C^1$-discontinuity areas of the entities, the asymptotic running time of the algorithm does not change. Hence, the worst-case running time of a practical implementation of our algorithm will be $\Theta(N^4)$ in three dimensions. It is easy to see that the worst-case running time in $d$ dimensions will be $\Theta(N^{d+1})$.

The average case is much more favorable. A straightforward analysis, that follows closely the analysis in [4, §4.7, pp. 85–89], shows that the expected number of invocations of all functions is only $\Theta(N)$. Assuming that both $I(p)$ and $M(p)$ are constant, one obtains that the expected running time of the entire algorithm is also $\Theta(N)$, which is optimal.

The amount of space used by the algorithm is clearly $\Theta(Np)$, which is the space needed to store all the hypersurfaces. At all times the algorithm holds no more than a constant number of spheres needed to decide whether the MES of four hypersurfaces touches one, two, three, or all the four of them.

9 Experimental Results

The algorithm described in Figure 9 has been implemented for computing the MES in two and three dimensions. Optimization using the convex hull has also been implemented in two dimensions. The
Here are two representation examples. Figure 11 shows two test results for the MES computation in two dimensions. The input consists of points and free-form planar curves. Figure 12 shows some test results for computing the MES in three dimensions, where the input consists of points, and free-form curves and surfaces.

**9.1 $C^1$-discontinuous Hypersurfaces**

The discussion so far has been for hypersurfaces which are assumed to be $C^1$ continuous. If $C^1$ discontinuous hypersurfaces are present, then their support is straightforward. It involves the introduction of low-dimensional hypersurfaces where the discontinuity occurs in the input set and splitting the original hypersurface along the discontinuity into two pieces. For example, consider $C^1$ discontinuity points in the case of planar free-form curves. Figure 13 shows the MES computed for an input set having $C^1$ discontinuous planar curves.

**9.2 Optimization Procedure**

Figure 14 shows some test cases after applying the convex hull optimization procedure described in Section 7. Thicker points and curves are retained for computing the MES whereas the thinner are the eliminated ones. Table 2 shows the optimization results in terms of statistical data. Note that the number of points and curves that are eliminated are quite large and hence there is a significant
Figure 12: The MES in three dimensions. The surfaces and the open curves in the figure are of third order.
Figure 13: The MES of points and planar free-form curves having $C^1$ discontinuities.

<table>
<thead>
<tr>
<th>Curves</th>
<th>Degree of Curves</th>
<th>Control Points</th>
<th>Input points</th>
<th>Curves eliminated in opt.</th>
<th>Points eliminated in opt.</th>
<th>Running time (sec.) with opt.</th>
<th>Running time (sec.) w/o opt.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>16</td>
<td>3</td>
<td>5</td>
<td>15</td>
<td>9</td>
<td>2.0</td>
<td>2.1</td>
</tr>
<tr>
<td>(b)</td>
<td>26</td>
<td>3</td>
<td>5</td>
<td>100</td>
<td>17</td>
<td>2.53</td>
<td>10.5</td>
</tr>
<tr>
<td>(c)</td>
<td>56</td>
<td>4</td>
<td>7</td>
<td>50</td>
<td>32</td>
<td>16.7</td>
<td>16.9</td>
</tr>
<tr>
<td>(d)</td>
<td>96</td>
<td>3</td>
<td>5</td>
<td>1</td>
<td>61</td>
<td>0.84</td>
<td>6.9</td>
</tr>
<tr>
<td>(e)</td>
<td>46</td>
<td>3</td>
<td>3</td>
<td>11</td>
<td>24</td>
<td>6.42</td>
<td>20.6</td>
</tr>
<tr>
<td>(f)</td>
<td>50</td>
<td>3</td>
<td>3</td>
<td>11</td>
<td>30</td>
<td>4.9</td>
<td>12.9</td>
</tr>
</tbody>
</table>

Table 2: Statistics of the test cases shown in Figure 14

change in the running times (the last two columns in the table give the running times with and without optimization). Figure 14(a) also shows the convex hull of the input points and the points sampled on the curves.

9.3 Large Data Sets

We have also experimented with the algorithm on several large data sets (of up to 40,000 curves), indicating the robustness of the implementation. Figure 15(a) and 15(b) show the computed MESs of 1,000 and 2,000 curves, respectively. The large data sets contain only curves. Table 3 shows the average time taken to compute MES for all data sets (Each experiment was repeated five times.). Figure 16 plots the running time as a function of the number of curves. It appears experimentally that the running time is roughly linear in the number of curves when the convex-hull optimization is not employed. However, when the optimization is performed, the average running time may not be linear as the procedure will eliminate an arbitrary number of curves (which depends on the specific
Figure 14: Test cases showing the effect of using the convex-hull optimization procedure. Convex hull is also shown in Figure 14(a).
Figure 15: Test cases for large data set without using optimization.

<table>
<thead>
<tr>
<th>Number of curves</th>
<th>1,000</th>
<th>5,000</th>
<th>10,000</th>
<th>15,000</th>
<th>20,000</th>
<th>30,000</th>
<th>40,000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg Running Time (w/o opt.) (sec.)</td>
<td>48.2</td>
<td>73.2</td>
<td>105.9</td>
<td>130.2</td>
<td>171.6</td>
<td>223.5</td>
<td>362.4</td>
</tr>
<tr>
<td>Avg Running Time (with opt.) (sec.)</td>
<td>20.3</td>
<td>14.96</td>
<td>39.37</td>
<td>50.64</td>
<td>71.4</td>
<td>79.18</td>
<td>183.9</td>
</tr>
</tbody>
</table>

Table 3: Average running time for large data sets (curves).

10 Conclusion

In this paper, we presented an optimal-time (expected) solution for the problem of computing the minimum spanning (enclosing) sphere of a set of points, free-form curves, surfaces, etc. It has been shown that the theory holds for any dimensions, though our implementation was for two and three dimensions only. The solution is inherently the same as for computing the minimum enclosing circle (or sphere) of a set of points; however, the implementation details of our solution are significantly more complicated. The expected running time of the algorithm is linear in the number of input hypersurfaces, but also contingent on the other details, such as the maximum degree of the hypersurfaces. It has been shown that the well-known results for MES of point sets can be adapted to continuous curves, surfaces, and hypersurfaces. In future, we also seek to overcome the assumption that the MES is tangent to a hypersurface at most once. Finally, a similar algorithm could be sought for hierarchical minimum enclosing computations, such as minimum enclosing ellipsoids, a problem we are now investigating.
Figure 16: Plot of the number of curves vs the average running time

Acknowledgements

Work on this paper has been supported in part by the Israel Science Foundation Grant 346/07, and in part by a grant for Korean-Israeli Research Cooperation. Work on this paper by the first author has been supported in part by a Lady Davis Fellowship, Israel. This work was also supported in part by KICOS through the Korean-Israeli Binational Research Grant (K20717000006) provided by MEST in 2007.

References


