Convex hull of Curved Objects via Duality – a General Framework and an Optimal 2-D Algorithm

Chao-Kuei Hung  Doug Ierardi
ckhung@usc.edu  ierardi@usc.edu

November 10, 1993

Abstract

We address the problem of finding the convex hull of curved polyhedra in Euclidean spaces of finite dimension. Optimal algorithms for simple curved polygons in $E^2$ exist but the more general version of this problem seems missing from the literature.

A precise definition of the objects under consideration is given. It encompasses most interesting solids that could arise in practical applications of geometric modeling. Based on the decomposition theorem for the polar set transformation [HI93], we generalize one of the component transformations and obtain a framework for computing the hull in $E^n$. Taking advantage of adjacency information, we are able to improve the general framework in the special case of simple curved polygons in $E^2$ and devise a new optimal algorithm. Its relatively simpler logic is contrasted to the existing optimal algorithms.

1 Introduction

In a previous report [HI93] we showed that the classical polar set transformation can be decomposed into three more primitive transformations. As the former is found to be closely related to convex sets, we generalize one of the component transformations in this report and show that convex hull finding of a large class of piecewise smooth compact objects in $E^n$ can be reduced to the construction of a particular cell in an arrangement of hypersurfaces induced by this generalized transform. A simple 2-D algorithm is spelled out to illustrate how the logic is simplified using the ideas which we develop.

Section 2 defines precisely what we mean by curved objects as the foundation of our discussion. We then generalize the scheme for imaging hyperplanes introduced in [HI93] to smooth faces of any dimension in Section 3. There we also prove the major theorem and point out a strategy for computing the convex hull of an $n$-dimensional curved polyhedron by calling upon duality and existing algorithms for cell construction in an arrangement of hyperplanes. Section 5 presents a combinatorially optimal algorithm for $E^2$, which compares favorably
against other existing optimal algorithms in terms of simplicity. Some concluding remarks are
given in Section 6. Appendix A lists a few facts on convex sets, the decomposition theorem,
and its corollaries. Algebraic details needed for the 2-d algorithm are omitted from the text
but are discussed in Appendix B.

2 Objects under Consideration

Our aim is to include in our definition of curved objects not only polyhedra, but also things
which are “smoothly deformed” from polyhedra. In the following the interior of a set $A$ is
denoted $\overline{A}$; its closure $\overline{\overline{A}}$. We begin by recalling the following definition taken from [Cai68, p. 86], which generalizes simplexes.

**Definition 2.1** A convex open polyhedral $n$-cell $c^n \subset E^n$ is a bounded set which is the inter-
section of a finite collection of open half-spaces. Its closure $\overline{c^n}$ is a convex closed polyhedron.
Its boundary $\partial c^n = \overline{c^n} - c^n$ falls naturally into convex open $j$-cells ($j < n$), called $j$-faces of
$c^n$. Its interior is considered to be a (the only) $n$-face. A $m$-face is proper if $m < n$.

Just as simplexes are the building blocks of complexes, so are convex polyhedra the building
blocks of general polyhedra complexes [Cai68, p. 87].

**Definition 2.2** A (finite) polyhedral $k$-complex $P^k \subset E^n$ is a finite set $\Theta$ of convex open
polyhedral $m$-cells ($m \leq k$) such that

1. every face of a member of $\Theta$ is a member of $\Theta$,
2. for every pair of members of $\Theta$, the intersection of their closures is either empty or the
closure of a common face, and
3. $\Theta$ contains at least one $k$-cell.

The point set union $|P^k| \equiv \bigcup \Theta$ is called a polyhedron.

We will drop the word finite since we won’t be interested in infinite polyhedral complexes. An
immediate consequence of this definition is:

**Lemma 2.3** A polyhedral complex is compact.

The incidence relations between the faces of a polyhedral $k$-complex can be thought of as
a graph. It is then natural to define isomorphism between polyhedral complexes as an iso-
morphism between the graphs representing the incidence relations which also preserves the
dimension of the faces and their neighbors (see [Cai68, p. 80] for isomorphism between sim-
plcial complexes). Note that according to this definition a polyhedron may have “dangling
pieces” which have a lower dimension than the “main body”.
Figure 1: (a) A convex open polyhedral 2-cell and (b) a polyhedral 2-complex. Note that the latter has two connected components.

**Definition 2.4 (Curved Polyhedron)** Let $P^k \subset E^n$ be a closed polyhedral $k$-complex and $C$ a subspace of $E^n$ homeomorphic to $|P^k|$ with smooth faces (each being the image of a face under a continuously differentiable function). Then $C$ is called a curved polyhedron of dimension $k$. $P^k$ is said to be a topological subdivision of $C$ and $C$ a smooth deformation of $|P^k|$. The subsets of $C$ corresponding to the faces of $P^k$ will be called the faces of $C$ with respect to the topological subdivision $P^k$. The boundary of $C$ will be denoted by $\partial C$.

The choice of the the term “curved polyhedron” is motivated by the more general term abstract polyhedron. Roughly speaking the latter refers to a collection of “abstract faces” having only labels but not necessarily any corresponding point sets in Euclidean space. A curved polyhedron in our definition is just a realization [Cai68] of such abstract polyhedra in $E^n$, characterized by smoothness constraints on the bounding faces, etc.

Our definition is actually weaker than what “smooth deformation” would be intuitively. We require only that each individual face be smooth after the deformation. (Compare with smooth $k$-manifold with boundary in [Mil81, p.12] which insists on the deformation being smooth.) With respect to the class of the representable Euclidean subsets this class of objects lies between those of constructive solid geometry and constructive non-regularized geometry [RR91, RR92].

A small technical problem arises here. A curved polyhedron $C$ may have (in fact infinitely) many distinct (non-isomorphic) subdivisions. We will however, be more interested in its properties that are invariant among all subdivisions. For example, given a subdivision $P^k$ of $C$, the image of the faces under the homeomorphism contains $\partial C$ but usually not the other way around. A more careful definition would identify $\partial C$ alone; but for our computational
purposes, we will be contented with fixing any particular subdivision of $C$ and considering all of its faces. (It will turn out that different subdivisions may affect at most the execution time of our algorithm, but all give the correct result.) Thus when we refer to a face of $C$ we actually mean the face of a particular subdivision of $C$. As we shall see later, the extra “interior faces” will cause no computational problem. Following the terminology in [PS85, p. 91], we will call a 0-dimensional face a vertex, a 1-dimensional face an edge, and a $(k - 1)$ dimensional face a facet. (Our definition subsumes the algebraic curves in [BK91] and [KYP92].)

We conclude this section with a few observations leading to the design of the desired output data structure from a computational point of view.

**Observation 2.5** The convex hull of a curved polyhedron $A$ of dimension $k$ is a compact, connected manifold with boundary. Its dimension is at least $k$.

**Observation 2.6** Let $A$ be a star-shaped curved polyhedron of dimension $d$ in $E^n$ and $P$ a polyhedral subdivision of $\partial A$. Then there exists $A'$ a curved polyhedron on the unit hypersphere homeomorphic to $\partial A$ and having a polyhedral subdivision isomorphic to $P$.

Now imagine we map the boundary of $A$ to the hypersphere above and then punch a hole in the relative interior of a face, from which we cast rays to form a standard projection. We have effectively blown up the face containing the hole and arrived at the

**Observation 2.7** Let $A$ be a star-shaped curved polyhedron of dimension $n$ in $E^n$, $P$ be a polyhedral subdivision of $\partial A$ and $f$ be a facet of $A$. Then there exists $A' \subseteq E^{n-1}$ a curved polyhedron homeomorphic to $\partial A - \mathring{f}$ which has a polyhedral subdivision isomorphic to $P$.

As a special case of star-shaped objects, a convex object, in particular the convex hull of a curved polyhedron also enjoys this property. Two consequences of much interest immediately follow:

**Observation 2.8** In $E^2$ the convex hull can be represented as a (linearly) ordered set of vertices and curved arcs.

**Observation 2.9** In $E^3$ the convex hull can be represented as a planar graph. If desired, it may be modified to admit a straight-line planar embedding, possibly with the addition of vertices and/or edges.

Observation 2.8 may seem trivial but it is interesting to see how much of the literature we review implicitly agree on this point. In view of Euler’s formula for planar graphs, Observation 2.9 gives an optimal representation for 3-D convex hulls since the size of the data structure can be made linear in the number of “interesting features” (vertices, arcs, and curved or planar facets) of the hull using “standard representations”. [MP78, GS85]
3 Generalized $\text{IH}$

We defined $\text{IH}$ in a previous report and showed that to compute the convex hull of a set $A$, one may find all the bounding and supporting hyperplanes of $A$, compute their $\text{IH}$, and repeat these two steps on the resulting set again (see also Appendix A). Though the computation of $\text{IH}$ is straightforward (Lemma A.3), it is not clear how we can find the set of all supporting planes (if we are just interested in the boundary of the convex hull) of $A$ when $A$ is curved and has more than one face$^1$. We remarked in [HI93] that by considering the supporting hyperplanes of a convex solid with smooth boundary, we are just rewriting $\partial \text{conv}(A)$ in a way more reminiscent of its identity as the envelope of its tangent hyperplanes. The remark does not generalize to concave solids, however. Nonetheless, it seems to be a good starting point as the computation of the tangent hyperplanes of a smooth surface is straightforward (that is, purely algebraic). For the moment let’s not worry about the combinatorial feature of the problem and neglect the troubles concavity might bring about by concentrating on the entirety of a single (smooth) face (only to the extent as it appears on an curved polyhedron).

**Definition 3.1** Let $\phi \subset E^n$ be an $m$-face of some curved polyhedron, $m < n$. Define $\Pi_r(\phi) \equiv \{ \pi \in \Pi_0 : \pi \text{ contains a tangent flat to } \phi \}$. In particular when $\phi$ is a vertex (singleton), it is considered its own tangent flat (of dimension 0) and hence $\Pi_r(\phi)$ is defined to be the set of hyperplanes containing $\phi$.

**Definition 3.2** Let $\phi \subset E^n$ be an $m$-face of some curved polyhedron, $m < n$ and $\mathcal{F}$ be the collection of its faces. Define

$$
\Pi_r(\tilde{\phi}) \equiv \bigcup_{\psi \in \mathcal{F}} \Pi_r(\psi)
$$

**Definition 3.3** Let $\phi \subset E^n$ be an $m$-face of some curved polyhedron, $m < n$. Define $\text{IH}(\phi) \equiv \text{IH}(\Pi_r(\phi))$ and denote its inversion by $\text{IH}(\phi)$. Similarly for $\tilde{\phi}$.

It is straightforward to verify that the definitions do not depend on the choice of the underlying curved polyhedron. They are natural extensions of the original $\text{IH}$ for hyperplanes. Incidentally, $\text{IH}(\gamma)$ is the pedal curve of $\gamma$ for $\gamma$ a plane curve [BG92]. The following lemma is another crucial one to the development of our algorithm. Now that the precision (in terms of what we really wanted) is sacrificed for ease of computation in the definition, is there still any hope to compute the exact boundary of the convex hull?

**Lemma 3.4** Let $\phi \subset E^n$ be an $m$-face of some curved polyhedron, $m < n$. Then

$$
\Pi_r(\tilde{\phi}) \subseteq \Pi_r(\phi) \subseteq \Pi_r(\tilde{\phi}) \cup \Pi_r(\tilde{\phi}).
$$

$^1$The polar set property, however, has been found useful in computing the convex hull of a finite set of points [Rag89, Ede87].
Speaking informally, by looking at \( \Pi_i(\mathcal{F}) \) we have included everything (all of \( \Pi_i(\phi) \)) we definitely want; yet the extra things we accidentally include all fall on the same side of \( \mathcal{H}(\Pi_i) \). Similar definitions and results carry over to an entire curved polyhedron as a whole if we generalize \( \mathcal{H} \) appropriately.

**Definition 3.5** Let \( A \subset E^n \) be a curved polyhedron and \( \mathcal{F} \) the set of its faces. Define
\[
\pi(A) = \bigcup_{\phi \in \mathcal{F}} \Pi_i(\phi)
\]
We also write \( \mathcal{H}(A) \) for \( \mathcal{H}(\pi(A)) \) and \( \mathcal{H}(A) \) for \( \mathcal{H}(\pi(A)) \).

**Corollary 3.6** Let \( A \subset E^n \) be a curved polyhedron. Then \( \Pi_i(A) \subset \pi_i(A) \subset \Pi_i(A) \cup \pi_i(A) \).

With luck, we hope to be able to further remove the extra pieces, thus obtaining exactly all the supporting hyperplanes. We will return to this point presently. Before delving into that, let’s give formulas for computing generalized \( \mathcal{H} \) and look at a few of its interesting geometrical properties. In the following let \( n \) be the dimension of the Euclidean space and \( m \) be the dimension of the face. The \( i \)-th row vector of array \( (a_{ij}) \) is denoted \( a_{i*} \) and \( j \)-th column \( a_{*j} \).

**Lemma 3.7** Let \( \pi \) be the flat whose points \( x \) satisfy:
\[
a_{i*} \cdot x = 1, \quad i = 1, 2, \ldots, m
\]
where \( a_{i*} \) are linearly independent. Then \( \mathcal{H}(\pi) \) is the flat
\[
\left\{ \sum_{i=1}^{m} \lambda_i a_{i*} : \sum_{i=1}^{n} \lambda_i = 1 \right\}.
\]

*Proof:*

A point in the flat claimed to be \( \mathcal{H}(\pi) \) corresponds to a hyperplane of the form
\[
\pi^* = \left\{ x : \sum_{i=1}^{m} (\lambda_i a_{i*}) \cdot x = 1 \right\}
\]
for some fixed choice of \( \lambda_i \) which sum to 1. If \( x_0 \in \pi \) then \( \sum (\lambda_i a_{i*}) \cdot x_0 = \sum \lambda_i (a_{i*} \cdot x_0) = \sum \lambda_i = 1 \) and hence \( x_0 \in \pi^* \). That is, \( \pi^* \in \mathcal{H}(\pi) \).

On the other hand for \( \pi \) to be contained in some hyperplane \( \pi^* \), the latter must have as its defining equation linear combinations of those defining \( \pi \). Hence \( \mathcal{H}(\pi^*) \) necessarily belongs to the claimed flat.

**Corollary 3.8** Let \( A \subset E^n \) be a linear \( k \)-face. Then \( \mathcal{H}(A) \) is a \((d-k-1)\)-flat. In particular, vertices are mapped to hyperplanes, flat facets to points, and straight line edges in \( E^3 \) to straight lines.
Proof:

Although the proof for the general case is just straightforward algebraic computation, we will give a more intuitive geometric argument for the special case of straight line edges in $E^3$. Consider a straight line segment $l \subset E^3$ and the point $p \in l$ closest to $O$. An elementary geometric argument shows that the perpendicular foot of each plane containing $l$ is on the (unique) circle $C$ of which $pO$ is a diameter; and conversely $C$ consists of only such points. To see what $C$ is mapped to by inversion, consider its restriction to the plane containing $C$. Certainly the restriction is an inversion in $E^2$ and $C$ lies in its domain. But then clearly $C$ is mapped to a straight line, and hence $l$. We can actually see that $f_{IH}(l)$ is orthogonal to the plane containing $l$ and $O$. \hfill $\square$

Lemma 3.7 says that the $\widehat{IH}$ of the intersection of some (finitely many) hyperplanes in general position is the affine hull of the $\widehat{IH}$ of these hyperplanes. Now consider a smooth face $\phi$ of dimension $n - m$ defined by the set of $m$ equations

$$f_i(x) = 0, \quad i = 1, 2, \ldots, m.$$ 

The tangent flat to $\phi$ at $x_0$ has equations

$$\nabla f_i(x_0) \cdot x = \nabla f_i(x_0) \cdot x_0, \quad i = 1, 2, \ldots, m$$

where

$$\nabla f(x) = \left( \frac{\partial f(x)}{\partial x_1}, \frac{\partial f(x)}{\partial x_2}, \ldots, \frac{\partial f(x)}{\partial x_n} \right)^T$$

Thus by Lemma 3.7, $x_0$ contributes to $\widehat{IH}$ the affine hull of the points

$$a_i = \frac{\nabla f_i(x_0)}{\nabla f_i(x_0) \cdot x_0}, \quad i = 1, \ldots, m.$$ 

To obtain an implicit equation we may consider each point in the above flat as the sum of $a_{s_m}$ and a linear combination of the vectors $a_j - a_m$, $j = 1, 2, \ldots, m - 1$. With a little linear algebra we can summarize the above discussion in the following

**Lemma 3.9** Let $\phi$ be a smooth face of dimension $n - m$ defined by the set of $m$ equations

$$f_i(x) = 0, \quad i = 1, 2, \ldots, m$$

and let

$$a_i(x_0) = \frac{\nabla f_i(x_0)}{\nabla f_i(x_0) \cdot x_0}, \quad i = 1, 2, \ldots m.$$ 

Let $A(x_0)$ be the $(m - 1) \times n$ matrix whose $i$-th row vector is $(a_i(x_0) - a_m(x_0))^T$ and let $c_i(x_0)$ for $i = 1, 2, \ldots, n - m + 1$ be a basis for its null space. Then a set of implicit equations representing $\widehat{IH}(\phi)$ is given by eliminating $x_0$ from

$$(x - a_m(x_0)) \cdot c_i(x_0) = 0, \quad i = 1, 2, \ldots, n - m + 1$$

$${f_i(x_0) = 0, \quad i = 1, 2, \ldots, m}$$
Here we make use of the assumption that every point on the face is regular (and that we have a “regular representation”) and the fact that the rank of a linear transformation plus the dimension of its null space equals the dimension of its domain.

**Corollary 3.10** In general the dimension of $\tilde{\text{IH}}$ of a $k$-face in $E^n$ is $n - 1$ regardless of $k$ (i.e., excluding developable surfaces and their appropriate generalizations).

*Proof:*

The difference between the dimension of a hyperplane and the object gives the degrees of freedom that a hyperplane can “hinge” at a fixed point on the face. The dimension of the face itself adds the remaining degrees of freedom to the resulting family of hyperplanes. (A more precise argument will invoke the Grassmanians.)

**Lemma 3.11** Let $p$ be a point on a $(d - 1)$-face $A \subset E^n$. Let $t_A(p)$ denote the tangent hyperplane to $A$ at $p$. Then

$$\tilde{\text{IH}}(p) = t_A*(\tilde{\text{IH}}(t_A(p)))$$

*Proof:*

We sketch the proof in $E^2$. Let $A$ be described by the parametric equation $p(t) = (x(t), y(t))$. Then $\tilde{\text{IH}}(A)$ is described by

$$p^*(t) = \frac{1}{x(t)y'(t) - y(t)x'(t)}(y'(t), -x'(t))$$

The tangent vector to $p^*(t)$ is

$$\frac{x''y' - x'y''}{(xy' - x'y)^2} \cdot (y, -x).$$

It has the same direction as $(p(t))^*$. But both share the point $p^*(t)$ so they must coincide.

Summarized briefly, the above lemma says that a tangent upstairs (i.e., in the inverted $\text{IH}$ space) corresponds to a point downstairs\(^2\) (i.e., in the original $E^n$). This is dual to the definition of $\text{IH}$, which says that a a tangent downstairs corresponds to a point upstairs.

**Corollary 3.12** A common support downstairs corresponds to an intersection upstairs; an intersection downstairs corresponds to a common support upstairs.

Please also refer to the figures showing several simple curves together with their $\tilde{\text{IH}}^3$. Except for figures 6 and 7, each curve is described by a single smooth equation, offset from its “canonical positions” (the origin is marked by a cross) to give more interesting features in its $\tilde{\text{IH}}$. In addition to the tangents to the smooth curve, the end points are also mapped (to a straight line) and shown as its $\tilde{\text{IH}}$.

\(^2\)From now on we shall freely use the terms *downstairs* and *upstairs* with these connotations.

\(^3\)The correspondence between points on the original curve and those on the $\tilde{\text{IH}}$ is clearer if you are viewing a color output of the postscript version of this report.
Figure 2: Sine curve at an offset, its $\overline{IH}$, and its convex hull. The vertices of the original curve and their images (straight lines) are labeled. So are the intersections (two vertices on the cell containing the origin) in $\overline{IH}$ corresponding to the supports of the hull.
Figure 3: The cardioid and its $\tilde{\mathbf{H}}$. The horizontal line in the $\tilde{\mathbf{H}}$, tangent to an inflection point, is the image of the cusp in the original curve. The cusp being inside the hull, its image lies outside the cell containing the origin.

Figure 4: The cycloid and its $\tilde{\mathbf{H}}$. In the arrangement created by its $\tilde{\mathbf{H}}$, notice how the three vertices around the cell containing the origin (there seem to be five, but two are actually places where tangent curves meet) correspond to the three supports needed to complete the hull.
Figure 5: The clover leaf and its \( \tilde{\mathbf{H}} \). The vertical line in the \( \tilde{\mathbf{H}} \), tangent to the \( \tilde{\mathbf{H}} \) at three places, is the image of the intersection of the three branches of the clover leaf.

Figure 6: An S-shaped figure and its \( \tilde{\mathbf{H}} \). The two straight lines are completely outside the cell containing the origin, since they are the images of the two end points, which are inside the convex hull.
Figure 7: A headset and its $\widehat{H}$. Images of the vertices are removed to simplify the picture. (Their presence wouldn’t affect the cell containing the origin since the vertices are inside the hull.)

**Lemma 3.13** In $E^3$ a edge is mapped to a ruled surface.

*Sketch of proof:*

At each point of the (curved) edge the tangent hyperplanes consist of all hyperplanes containing the tangent straight line at that point. By the previous lemma this gives rise to a straight line. Repeating the argument for every point on the curve gives us a one-parameter family of straight lines in space, which constitute a ruled surface.  

**Lemma 3.14** In $E^3$ a developable surface is mapped to a curve.

*Sketch of proof:*

A developable surface is the envelope of a one-parameter family of planes. The image is therefore a curve.

We now return to algorithmic considerations and present the crux of our development which leads to the development of an $n$-dimensional convex hull algorithm.

**Theorem 3.15** Let $A \subset E^n$ be a curved polyhedron whose convex hull contains the origin and let $\mathcal{F}$ be the collection of its faces. Then

$$\partial(\text{conv}(A)^*) = \{x \in \widehat{H}(A) : \text{x } \mathcal{C} \cap \widehat{H}(A) = x\}$$

That is, the boundary of the polar set of $\text{conv}(A)$ consists of every first intersection of each ray shooting from the origin with $\widehat{H}(A)$.  

12
In view of Lemma A.2 we may assume that $A$ is convex, or equivalently that $A = \text{conv}(A)$ without loss of generality. In view of Corollary A.5 we need only show instead that the described set (call it $B$) is equal to $\overline{\mathbb{H}}(\Pi_s(A))$. In view of Corollary 3.6 (and the comments preceding/following it) all we need to show is that the cutting planes account for exactly all the planes excluded by the condition. A simple geometric argument verifies this. \qed

**Observation 3.16** Let $A \subseteq E^n$ be a curved polyhedron and $\mathcal{F}$ its faces. Consider the arrangement of (the hypersurfaces) $\{\overline{\mathbb{H}}(\phi) : \phi \in \mathcal{F}\}$. Then $(A \cup \{O\})^*$ is the cell in this arrangement whose interior contains the origin. It is bounded if and only if $O \in \text{relint}A$.

4 Complexity of Convex Hulls in $E^n$ and a Near-Optimal Algorithm in $E^3$

In this section we briefly argue that the $\overline{\mathbb{H}}$ transformation enables us to compute the convex hull of a curved polyhedron in $E^3$ by way of reduction to Sharir’s algorithm for computing lower envelopes [Sha93]. In that paper it was shown that the combinatorial complexity of the lower envelope of $N$ surface patches is $O(N^{2+\epsilon})$ for all $\epsilon > 0$ under some reasonable assumptions and the assumption that the vertical projection of each patch onto the $xy$-plane is bounded by a constant number of algebraic arcs of constant maximum degree. In $E^n$ a similar result holds and the complexity is $O(N^{d-1+\epsilon})$. A randomized algorithm with expected running time $O(N^{2+\epsilon})$ for constructing the lower envelope was also presented in the case of $E^3$.

Recall that by curved polyhedron we mean a realization of a closed polyhedral complex, which need not be connected nor a manifold. The randomized algorithm with expected running time $O(N^{2+\epsilon})$ is reasonably good since the complexity of the convex hull of even $N$ spheres can be as large as $\theta(N^2)$ [BCD+$^92]$.

**Lemma 4.1** The problem of finding a particular star-shaped cell $\sigma$ in $E^n$ in an arrangement of algebraic hypersurface patches, given a point $p$ in its kernel, is linearly transformable to the problem of finding the lower envelope of an arrangement of hypersurface patches.

**Proof:**

First we translate the origin of the coordinate system to $p$. Fix an arbitrary unit vector $e$ rooted at the origin and consider the perspective projection

$$\mathcal{P} : x \mapsto \frac{1}{x \cdot e} x$$
which maps a point in the halfspace passing \( p \) with (inward) normal \( e \) to a point on the hyperplane passing \( e \) with the same normal. The projection does not change the direction of a vector but is “highly non-injective”. We can however make it injective by “blowing” points away from the hyperplane according to the length of their pre-image. Specifically let

\[
\mathcal{T} : x \mapsto \frac{1}{x \cdot e} x + |x|^2 e.
\]

It is not difficult to verify that \( \mathcal{T} \) is injective (over the halfspace). Moreover, \( \mathcal{T}(x) \) is on the lower envelope if and only if \( x \) is on the boundary of the star-shaped cell containing \( p \). The above transformation is algebraic and hence can be carried out in time proportional to the number of the given hypersurface patches (up to a constant determined by their maximum degree). The reduction is completed by a similar transformation that takes care of the other half space.

\[\square\]

**Corollary 4.2** The combinatorial complexity of any star-shaped cell of \( N \) hypersurface patches in \( E^n \) satisfying the conditions in [Sha93] is \( O(N^{n-1+\epsilon}) \) for all \( \epsilon > 0 \), where the constant of proportionality depends on the maximum degree and the connectivity of the surface patches.

**Corollary 4.3** Given a point in the kernel of a star-shaped cell in an arrangement of \( N \) surface patches in \( E^3 \) satisfying the conditions in [Sha93], the cell can be computed in time \( O(N^{2+\epsilon}) \).

The following corollaries are now immediate by Theorem 3.15 and by the fact that there is a direct one-to-one correspondence between \( \overline{\mathcal{H}}(A) \) of a curved polyhedron \( A \) and its convex hull.

**Corollary 4.4** Let \( P \) be a curved polyhedron in \( E^n \) whose faces are algebraic of bounded degree and have bounded number of subfaces. Then the combinatorial complexity of its convex hull is \( O(N^{n-1+\epsilon}) \) for any \( \epsilon > 0 \) for any \( N \), where the constant of proportionality depends on the maximum degree and the connectivity of the surface patches.

**Corollary 4.5** The convex hull of a curved polyhedron having \( N \) faces in \( E^3 \) can be computed in time \( O(N^{2+\epsilon}) \), provided that each face has a bounded number of subfaces.

### 5 Simple Curved Polygons in \( E^2 \)

Not until recently did the problem of convex-hulling curved objects receive some attention [SW87, DS90, BK91, KYP92]. Bajaj and Kim [BK91] consider areas bounded by implicit algebraic curves. The edges are segmented at singular and inflection points so that each piece becomes monotone. The algorithm follows the one-time-push-pop structure similar to that for ordinary simple polygon [PS85], with more complicated cases carefully examined.
The complexity is analyzed to be linear in the number of (segmented) edges and polynomial in the maximum degree of the curves. Dobkin and Souvaine [DS90] assume oracles for the algebraic computations and deal with splinegons whose edges are assumed to be either convex or concave. Then a bounding polygon is used to help compute the hull following the similar pocket-lid analysis in the ordinary simple polygon case. The complexity is the product of the time for an oracle query and the number of edges.

As pointed out in the previous section, the algorithm which the general strategy may lead to is not likely to be efficient. In $E^2$, however, if we restrict ourselves to the more interesting curved polyhedra such as the analogs of simple polygons, we may hope to obtain a much more efficient algorithm by utilizing adjacency information. Among other things, such objects enjoy the properties of being connected, having no “holes” inside, no self intersections, and no dangling faces. Specifically we assume that the edges $e_1, e_2, \ldots, e_n$ of the given curved polygon $A$ are given in clockwise order around its interior (boundary representation). The information associated with each individual edge is assumed to be reasonably complete so that we can

- find its tangents and normals;
- decide whether a point is on a given edge;
- at each point on the edge, identify one of the two normals that points towards the interior of the polygon

each in $O(1)$ time. The algorithm has (combinatorial) complexity linear in the number of edges (times an algebraic factor) thought it still requires a significant amount of equation solving, which is inevitable. The logic is, however, relatively simpler compared to the existing optimal algorithms. The simplicity is due to the segmentation of the edges at the turning points as opposed to the inflection points. This idea is in turn a rather natural choice once we see the IH analysis, where the origin plays the “central” role. We assume a subroutine for solving systems of equations numerically or symbolically [Pet89, Ost66, SS92, MD92].

We begin by constructing an arbitrary convex circumscribing polygon $\Delta$ of $A$. What we really want here is a circumscribing triangle although any convex circumscribing polygon will do. A circumscribing rectangle whose edges parallel to the coordinate axes is probably easiest to compute. We find on all edges the points at which the tangent is horizontal. Among these and all vertices we pick the two with extremal $y$-coordinates. They define the upper and lower sides of the rectangle. Similarly for the other two sides of the rectangle.

Each side of the circumscribing polygon contacts $A$ at one or more points. Let’s number the points of contact $p_0, p_1, \ldots, p_{K-1}$ in the order in which they are visited as we traverse the edges of $A$ in clockwise order. At each point $p_i$ which is not a vertex of $A$ we break the edge by inserting $p_i$ as an additional vertex. It is not difficult to verify, either directly or by our previous results, that

- $p_0p_1 \cdots p_{K-1}$ is a simple, convex polygon;
**Figure 8: Efficient Convex Hull of Curved Polygon in \( E^2 \)**

**Input:** A planar simple curved polygon \( A \) described as an ordered list of edges and vertices.  
**Output:** Its convex hull \( ch \) as an ordered list of edges and vertices.

1. Find a convex circumscribing polygon \( \Delta \) of \( A \).
2. Break the edges in contact with \( \Delta \) into halves at the points of contact \( p_0, p_1, \ldots, p_{K-1} \),  
   breaking the edge list into sublists \( Q_0, Q_1, \ldots, Q_{K-1} \) at the same time.
3. Pick a point \( \mathcal{O} \) inside the polygon \( p_0p_1 \cdots p_{K-1} \) as the origin.
4. \textbf{foreach} edge \( e_i \) \textbf{do}
   
   - break \( e_i \) at the turning points.

   \( ch \leftarrow \{ \} \)
5. \textbf{foreach} \( Q_i \) \textbf{do begin}
   
   - \( S \) will be a stack of edges contributed by \( Q_i \) to the hull.
   - \( S \leftarrow \text{push}(\text{deleteq}(Q_i), \{ \}) \)
   - initialize \( \omega[e_0] \) as in the text
6. \textbf{while} \( Q \neq \{ \} \) \textbf{do begin}
   
   - \( e_i \leftarrow \text{copy-top}(S) \)
   - \( e_j \leftarrow \text{deleteq}(Q) \)
   - \textbf{if} \( e_j \) goes backwards \textbf{then}
     
     - \textbf{if} \( e_j \) goes above \textbf{then}
       
       - \textbf{repeat} \( e_j \leftarrow \text{deleteq}(Q) \) \textbf{until} \( e_j \) goes forwards
       
       - \textbf{else} (* \( e_j \) goes below *)
         
         - \textbf{repeat} \( e_j \leftarrow \text{deleteq}(Q) \) \textbf{until} \( e_j \) goes forwards \textbf{and} \( e_j \cap v_i\mathcal{O} \neq \{ \} \)
   
   - \textbf{loop}
     
     - compute \( \omega[e_j] \) (see text)
     
     - \textbf{if} \( \omega[e_j] \) is ahead of \( \omega[e_i] \) \textbf{then} \textbf{break}
     
     - \( e_i \leftarrow \text{pop}(S) \)
   
   - \textbf{forever}
     
     - \textbf{push}(e_j, S)
   
   - \textbf{end}
7. \textbf{while} \( S \neq \{ \} \) \textbf{do begin}
   
   - \( e_i \leftarrow \text{pop}(S) \)
   
   - \textbf{append} \( ch, \partial(\text{conv}(e_i \cup \{ \mathcal{O} \})) \)
8. \textbf{end}
9. \textbf{end}

10. \textbf{end}
- the circular order of \( p_0, p_1, \ldots, p_{K-1} \) around the curved polyhedron is the same as the angular order around any point inside the polygon \( p_0 p_1 \cdots p_{K-1} \).

- There is no common support between the respective sector of the convex hulls of the adjacent sublists except at the shared vertex. This implies that upstairs they either intersect at a single shared vertex or a shared straight line edge.

Next we pick a point \( O \) inside the polygon \( p_0 p_1 \cdots p_{K-1} \) as the origin. Clearly

- \( O \) lies in \( \text{conv}(A) \);

- The angle \( \angle p_i Op_{i+1} \) is less than \( \pi \).

In order that each edge have a definite orientation with respect to \( O \) (either clockwise or counterclockwise), we further break the edges at the turning points (line 4). Take the implicit equation representation for example, we may first solve for the zeros of

\[
\begin{align*}
\frac{\partial}{\partial y} f(x, y) &= 0 \\
x \cdot \frac{\partial}{\partial x} f(x, y) &= y \cdot \frac{\partial}{\partial x} f(x, y)
\end{align*}
\]

That is, find the points on the curve at which the position vector has the same direction as the that of the tangent vector. The solutions need be checked against the predicates to see if they really fall on the edge. Finally the neighborhood needs to be examined (perhaps by taking the quadratic form) to take care of the (unlikely) case that it is also an inflection point.

In step 5 we initialize the output \( \text{ch} \) of the algorithm which will eventually be the boundary representation of \( A \) consisting of the list of edges and vertices of \( \text{conv}(A) \) in clockwise order. Now each pair of adjacent points of contact found in steps 1–2 \( p_i, p_{i+1} \) define a sublist of curved edges of \( A \). We will process each sublist in the same manner as described in the following paragraphs, and then concatenate the results.

Call the edges and vertices in the current sublist \( v_0, e_0, v_1, e_1, \ldots, v_k \) (in the order they are encountered when one travels around \( A \) in a clockwise direction). Note that \( v_0 = p_M \) and \( v_k = p_{M+1} \) for some \( M \) (the index of the current sublist). We first initialize a stack \( S \) of the “contributing edges” (see below) and a queue \( Q \) of the unvisited edges.

In addition we initialize \( \omega \) an array indexed by the edges. Intuitively, \( \omega[E] \) is the \( \tilde{\text{IH}} \) of the “first” (clockwise) supporting straight line contributed by \( E \). If \( e_1 \) is forward we let \( \omega[e_1] \) equal to the \( \tilde{\text{IH}} \) of the “first tight support” of \( e_1 \) (see below for how other entries of \( \omega \) are obtained similarly), otherwise \( \omega[e_1] \) is assigned the ideal point which is the limit of the \( \tilde{\text{IH}} \) of the straight line passing \( v_0 \) as it rotates clockwise around \( v_0 \) towards the origin. This vector upstairs will be used later in deciding whether or not to pop \( e_0 \) off \( S \) (see below).

Then we enter the main loop (line 12) to process all edges in this sublist in order, taking one element \( e_j \) from \( Q \) at a time. We maintain a slightly different invariant from those in other papers. If the stack \( S \) contains the edges \( e_{i_0}, e_{i_1}, \ldots, e_{i_m} = e_i \), with \( e_i \) on the top, then at this point of any iteration,
• $c_0 < c_1 < \cdots < c_m$

• The edges in $S$ are exactly those which contribute to the part of the convex hull of $O$ and the edges $e_0, e_1, \ldots, e_i$ that lies in the sector $\angle v_0 O v_i$.

• The first unvisited edge is the immediate successor of the edge on the top of stack (i.e., $j = i + 1$).

Moreover, all edges in the stack are forward edges.

We first skip all edges that cannot possibly contribute to the convex hull (line 15). If the first edge $e_j$ in just removed from the queue $Q$ is forward, we skip to line 20, described in the next paragraph; otherwise we proceed here. Certainly a backward (counterclockwise) edge can never contribute; depending on whether the first backward edge goes “above” or “below” $e_i$ (top of stack, the immediate predecessor), there may be other edges which may be discarded without much computation. This can be determined by checking whether the boundary is making a left turn or right turn here using the bit of information that distinguishes between the inside and outside of $A$ in the boundary representation. If $A$ folds on top of itself with respect to $O$ at $e_j$, namely $A$ is locally concave at $v_i = v_{j-1}$, we simply eliminate all subsequent backward edges. Otherwise $v_i$ is a forward pointing vertex of $A$ ($A$ is locally convex at $v_i$). In the latter case we eliminate in addition all edges that are completely outside $\angle v_0 O v_k$ the sector of our concern. This is to avoid considering awkward edges that go around the origin in the right direction but are irrelevant. Whether an edge is completely outside the sector can be determined by checking if it intersects the ray $O \overrightarrow{v_i}$ since the edge goes around $O$ in a definite direction.

After skipping all the non-contributing edges (possibly none), we arrive at a forward edge at least part of which is inside the sector of our interest (step 20). Following the algorithm we will call it $e_j$ (now $j = i + 1$ is not necessarily true). We will then pop off the stack $S$ all edges that no longer contribute to the convex hull due to the introduction of $e_j$. The criterion for popping off the top element is as follows. Intuitively, $\omega$ is the $\widetilde{H}$ of the “first” (most counterclockwise) supporting straight line contributed by $e_i$. We now compute $\omega[e_j]$, the $\widetilde{H}$ of the “first” supporting straight line contributed by $e_j$, which is also the “last” (most clockwise) supporting line contributed by $e_i$. This is done by finding the most counterclockwise intersection of $\widetilde{H}(e_i)$ and $\widetilde{H}(e_j)$. If $\omega[e_j]$ is ahead of $\omega[e_i]$, i.e., $\omega[e_j]$ is more clockwise than $\omega[e_i]$, then at least part of $e_i$ is not occluded, i.e., part of $e_i$ still contributes to the convex hull. Otherwise, no part of $e_i$ can appear in the convex hull.

Finally we have in $S$ exactly all the edges that contribute to the portion of $\text{conv}(A)$ within the sector $\angle p_M O p_{M+1}$ (line 26). We may now append to the output list $ch$ the actual hull contributed by each edge, in reverse order. Let $e_i$ be the edge we are processing. Working upstairs, we compute $\widetilde{H}(e_i)$ and find the cell containing the origin. The vector $\omega[e_i]$ points to the direction around this cell with which $e_i$ begins to contribute. The ending direction is either on $\widetilde{H}(p_{M+1})$ (for $e_i$ top of stack) or $\omega[e_j]$ where $e_j$ is the edge previously popped off the stack.
As pointed out earlier the logic of our algorithm is definitely much simpler than that of the other existing optimal algorithms. In the special case when the input is an ordinary polygon, we have a simpler alternative to the polygon convex hull algorithm [PS85]. A full analysis of the algebraic factor in the complexity (such as the effects of the degrees of the edges on the number of inflection points vs turning points, etc.) is required for a comparison in terms of efficiency.

6 Conclusions

We have defined a broad class of curved objects in Euclidean $d$-space and given a transformation on them which leads to the reduction of the convex hull finding problem to the cell arrangement problem. A (combinatorially) optimal algorithm for the special case of planar simple curved polygons is presented, its simple logic contrasted to the existing optimal algorithms.

The general result is of theoretical importance in that it is independent of the dimension of the space; whereas the 2-d algorithm looks like a promising candidate for actual implementations. We are currently working to improve the general strategy in the 3-d case and reduce the number of equations to be solved.

Acknowledgement

We thank David Kriegman for sending us useful references on the subject of convex hull finding for curves in the plane. The 3-d problem was first stated to one of the authors by Ken Goldberg. We are also indebted to the following people for their comments: Francis Bonahon, Yuibin Chen, Ming-Deh Huang, Alfred Inselberg, Ram Nevatia, and Aristides Requicha.

The softwares used to produce the pictures include Rlab, gnuplot, ghostscript, dvips and \LaTeX. These quality freewares made our research not only easier, but also enjoyable.

A Imaging the Hyperplanes and the Decomposition of the Polar Set transformation – a Brief Review

In this section we list the major results from [HI93].

Definition A.1 Let $A \subset E^n$. Define the following sets of hyperplanes:

\[
\begin{align*}
\Pi_b(A) & \equiv \{ \pi : \pi \text{ strictly bounds } A \} \\
\Pi_s(A) & \equiv \{ \pi : \pi \text{ supports } A \} \\
\Pi_c(A) & \equiv \{ \pi : \pi \text{ cuts } A \} \\
\Pi_{bs}(A) & \equiv \Pi_b(A) \cup \Pi_s(A)
\end{align*}
\]
Lemma A.2 Let $A \subset E^n$. Then $A^* = \text{conv}(A)^*$.

Lemma A.3 Let $\pi$ be the hyperplane $\{x \in R^d : n \cdot x = \delta\}$. Then $\tilde{\Pi}(\pi) = n/\delta$.

Theorem A.4 (Decomposition of the Polar Set Transformation) Let $A \subset E^n$ and $O \in A$. Then $A^* \setminus \{O\} = \tilde{\Pi}(\Pi_b(A) \cap \Pi_0)$.

Corollary A.5 Let $A \subset E^n$ be a convex closed set containing $O$. Then $A^*$ is convex and closed, and it contains the origin. Furthermore,

\[
\begin{align*}
(A^*)^c & = \tilde{\Pi}(\Pi_b(A)) \\
\partial(A^*) & = \tilde{\Pi}(\Pi_i(A)) \\
\text{exterior}(A^*) & = \tilde{\Pi}(\Pi_e(A))
\end{align*}
\]

Corollary A.6 Let $A \subset E^n$ be a compact set containing the origin. Then

\[
\text{conv}(A) = \tilde{\Pi}(\Pi_b(\tilde{\Pi}(\Pi_b(A)))).
\]

B Algebraic Details

The major detail omitted in the description of the algorithm is the exact representation of an individual edge.

Obviously a parametric equation with two bounding parameters corresponding to the end points is sufficient for our purpose. We argue briefly that with reasonable assumptions implicit equations are also acceptable. Assume edge $e$ is given in implicit equation $f = 0$ together with two of its zeros (in clockwise order around the polygon) as the end points $a$ and $b$. In addition assume that we are given a predicate $P$ for testing the membership of an arbitrary zero of $f$. Finally suppose all point on $e$ except at most one of the end points are regular. Clearly the computation of the transformation is straightforward (though implicit in the report), either symbolically or numerically. The only problem would be to decide which one of the two normals at any point on $e$ points inwards. We can easily determine this at the regular end point, say $a$. We simply look at the two zeros of $f$ near $a$ and apply $P$. This enables us to figure out which of the two tangents points to the “relevant” zeros of $f$ that are part of $e$. With the assumption that all (relative) interior points of $e$ are regular, the decision of which tangent to choose is unanimous with that at $a$. The choice of the normal is then trivial.

References


