The Visual Computer

Computing spanning line segments in three dimensions

Ming-En Wang[1], Tony C. Woo[2], Lin-Lin Chen[3], Shuo-Yan Chou[3]

[1] Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI 48109-2117, U.S.A.
[2] Department of Industrial Engineering, University of Washington, P.O. Box 352650, Seattle, WA 98195-2650 U.S.A.
[3] National Taiwan Institute of Technology, 43 Section 4, Keelung Road, Taipei 106, Taiwan

A set of spanning line segments $\mathcal{S}$ in a polyhedron P preserves the property of intersection; that is, a plane intersects P if and only if it also intersects $\mathcal{S}$. This paper gives a linear time algorithm for constructing $\mathcal{S}$ for a polyhedron with N extreme vertices. If N is odd, the algorithm is optimal in yielding $\lfloor N/2 \rfloor + 1$ spanning line segments. If N is even, it gives $(N/2) + 1$, which is optimal in some cases and nearly optimal in others.

Key words: Spanning line segments – Skeletons – Plane polyhedron intersections

e-mail: twoo@u.washington.edu
Correspondence to: Tony C. Woo

1 Introduction

Spanning line segments (SLSs) in a polyhedral object are a set of line segments, each connecting two vertices of an object, so that any plane intersecting the polyhedron intersects at least one segment in the set, and vice versa. A set of line segments forms SLSs of a polyhedral object if and only if it satisfies two properties (Wang et al. 1993a):

1. each vertex of the object must be covered by (i.e., be adjacent to) at least one line segment (the property of *completeness*).
2. The set of line segments is inseparable by any plane, i.e., no plane can separate the set into two nonempty subsets without intersecting any line segment (the property of *inseparability*).

The intersection of a polyhedron with a plane implies and is implied by the intersection of its convex hull with that plane. Therefore, we develop algorithms on computing SLSs on convex polyhedra with the understanding that if a polyhedron is not convex and not simply connected, its convex hull is taken.

In computer graphics and computer-aided design, an intersection between polyhedra is often detected via the "box text", for which the box is constituted by the maxima and minima of the vertices in the x, y, and z directions. A box with six faces has three diagonals, the intersections of which reduce the computation time by 50% (Ratschek and O'Rourke 1993). Generalizing such a notion to that of SLSs offers an interesting representation for polyhedra. While there is no equivalent, the closest notion is perhaps that of the "skeletons" (Blum 1967), which has led to a number of developments. Chief among them is the medial axis transform (Lee 1982), which has stimulated considerable interest in computational geometry (Goldak et al. 1991; Hoffmann 1990; Kirkpatrick 1979) and in pattern recognition (Blum and Nagel 1977; Bookstein 1979; Matheron 1988; Wu et al. 1986). There are differences, to be sure.

The SLSs for a given polyhedron are not unique. Fig. 1a shows a polyhedron with ten vertices, eight of which (the extreme vertices) are on the convex hull shown in Fig. 1b. An SLS set that satisfies completeness and inseparability is the set of line segments emitting from one of the extreme vertices, connecting to all other extreme vertices, as shown in Fig. 1c. If N denotes the total number of *extreme* vertices, such an SLS set has a size of

© Springer-Verlag 1996

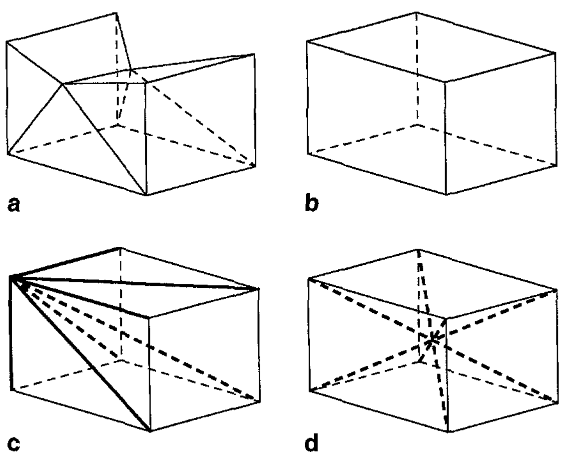


Fig. 1a–d. Two SLS sets for a polyhedron

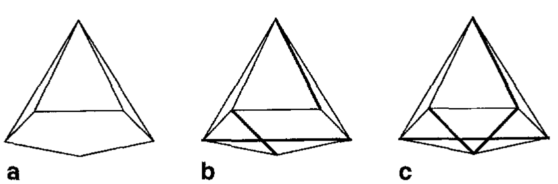


Fig. 2a–c. An even-vertex polyhedron with minimum size $(N/2)+1$ for its SLS

$(N-1) = 7$. A smaller set is shown with 4 diagonals in Fig. 1d. Clearly, an SLS set of the smallest size is desirable.

By noting that adding extra line segments that connect vertices to an SLS set generates another SLS set, the largest SLS set for a polyhedron with N vertices is the set of all the line segments connecting the vertices, which is of size $N(N-1)/2$. In contrast, according to the property of completeness, the smallest possible size of an SLS set is $\lceil N/2 \rceil$. However, exceptions exist, as some polyhedra with an even number of vertices have a minimum size $(N/2)+1$ instead of $N/2$. An example of such polyhedra, a polyhedral cone with a pentagonal base (facing the viewer), is shown in Fig. 2a. Consider generating an SLS set with $N/2 = 3$ line segments for the cone. As shown in Fig. 2b, to satisfy the property of completeness, one line segment is used to connect the top vertex to a base vertex, and two lines segment the remaining four base vertices in two pairs. However, as also illustrated in Fig. 2b, the segment covering the top vertex can be "sliced off" by a plane, which violates the property of inseparability. Hence, no set of three line segments qualifies as an SLS set for a hexahedral cone. (This result can be generalized, i.e., any polyhedral cone with a even number of vertices has a minimum size of $(N/2)+1$). By adding an extra line segment to those in Fig. 2b to increase the connectivity of the base vertices, an SLS set is generated and shown in Fig. 2c. We call an SLS set of the minimum size a *minimum SLS set*. These observations lead to the following problem definition.

Problem minimum SLS (mSLS) set. Given a polyhedron P, determine an SLS set of the minimum size.

It is understood that P is convex, for otherwise its convex hull would be taken. A recent linear time algorithm to compute an SLS set was based on a subalgorithm that computed the SLS set for a simple polyhedron with a fixed number (say, $k = 5$) of vertices (Wang et al. 1993b). Then the SLS set was grown by repeatedly applying the subalgorithm, adding $(k-1)$ vertices each time to the set already covered. The idea was to reuse a single covered vertex each time to ensure the inseparability of the enlarged set. Doing so, however, forfeits the goal of achieving the minimum size. In fact, the size of an SLS set thus obtained is about $[1 + 1/(k-1)]$ times the minimum size. To reduce this overhead, subalgorithms for an increasingly larger k need be developed. It is noted that subalgorithms for $k > 7$ vertices can be quite tedious and even difficult.

The current paper presents a linear time algorithm that computes near minimum SLSs. Two critical observations contribute to the development. First, suppose that two nonadjacent vertices are removed from a polyhedron that is then reconstructed. It is noted that adding the line segment connecting the two removed vertices to an SLS set of the reconstructed polyhedron forms an SLS set for the original polyhedron. Inseparability is automatically satisfied without reusing a vertex. This observation also suggests recursively removing nonadjacent vertex pairs from a given polyhedron, for completeness. The second observation, which states that there is always a pair of nonadjacent vertices of degrees less than or equal to five for any polyhedron with more than four vertices, provides a basis for a linear time complexity. No convex hull is taken to reconstruct the polyhedron. This algorithm obtains minimum SLS sets for polyhedra with an odd number of vertices and SLS sets of sizes at most one larger than minimum for polyhedra with an even number of vertices. The rest of this paper is organized as follows: the two observations are presented in Sect. 2, the algorithm in Sect. 3, and a brief conclusion in Sect. 4.

2 Computing near minimum SLS sets

Lemma 1. *The smallest possible size S^* of an SLS set for a polyhedron with N vertices is:*

$S^* = \lceil N/2 \rceil = \lfloor N/2 \rfloor + 1$ *if N is odd*

$S^* = \lceil N/2 \rceil = N/2$ *if N is even.*

Proof. As a line segment covers two vertices, at least $\lceil N/2 \rceil$ line segments are needed to satisfy the property of completeness, where $\lceil N/2 \rceil$ equals $\lfloor N/2 + 1 \rfloor$ if N is odd and $N/2$ if N is even. □

As illustrated in the previous section, not all polyhedra have a minimum SLS set with a size equal to the smallest possible size S^*. Some, such as the hexahedral cone shown in Fig. 2, have a minimum size of $(N/2) + 1$ instead of $N/2$. With this in mind, the rest of this section presents a method to compute the minimum SLS set. Let P denote an arbitrary polyhedron with a vertex set $\mathscr{V}$ containing N vertices.

Lemma 2. *For a pair of nonadjacent vertices p and q in $\mathscr{V}$, let $\tilde{P}$ and $\tilde{\mathscr{S}}$ denote respectively the convex hull $CH(\mathscr{V} - \{p, q\})$ and its SLS set. Adding the line segment (p, q) to $\tilde{\mathscr{S}}$ forms an SLS set for P.*

Proof. By definition, for $\tilde{\mathscr{S}} \cup \{(p, q)\}$ to be an SLS set for P, it needs to cover all the vertices of P and be inseparable by any plane. As the segment (p, q) covers p and q with $\tilde{\mathscr{S}}$ spanning the remaining vertices in $\mathscr{V}$, the union $\tilde{\mathscr{S}} \cup \{(p, q)\}$ satisfies the property of completeness. The property of inseparability is now shown to be satisfied by combining two arguments. First, being an SLS set of $\tilde{P}$, $\tilde{\mathscr{S}}$ is itself inseparable. Second, since P is convex, and

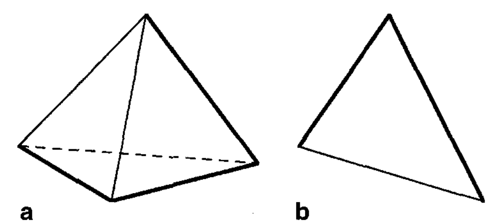


Fig. 3a, b. Computing the SLS set for the final convex hull

p and q are non-adjacent, the line segment (p, q) intersects polyhedron $\tilde{P}$. This implies the inseparability of polyhedron $\tilde{P}$ and the line segment (p, q). [In the case that p and q share a face of P, edge (p, q) "touches" either a face or an edge of $\tilde{P}$. The lemma also holds for this case.] □

Lemma 2 suggests an algorithm:

1. Remove a pair of nonadjacent vertices from P and add the line segment connecting them to an initially empty set $\mathscr{S}$.
2. Reconstruct the convex hull for the remaining vertices.
3. Repeat this procedure until the reconstructed convex hull contains no more than four vertices.

Since exactly two vertices are removed in each iteration, the final convex hull has either three or four vertices; its SLS set is easy to obtain. (An SLS set for the final convex hull, shown as bold line segments in Fig. 3, can be obtained by connecting the vertices in an arbitrary order with a chain of edges.) An SLS set of size three or of size two is then added to $\mathscr{S}$. We call the algorithm mSLS.

Lemma 3. *For polyhedra with N vertices, algorithm mSLS yields the number $\mathscr{S}$ of the SLS sets, where:*

$S = \lfloor N/2 \rfloor + 1.$

Proof. Provided that any polyhedron with more than four vertices has a pair of nonadjacent vertices, implied by Lemma 4, which is to appear shortly, algorithm mSLS is applicable to any polyhedron. Since two vertices are removed in each iteration, exactly $\lceil (N - 4)/2) \rceil$ iterations are executed before the final convex hull is obtained. Each iteration adds a line segment to the SLS set. After adding an SLS set for the final hull, the overall SLS set has a size of $\lceil N/2 \rceil + 1$ line segments. □

Two examples are presented now to illustrate the application of the algorithm mSLS. The first example, as shown in Fig. 4, illustrates the application of algorithm mSLS to an odd-vertex polyhedra where N equals 9. In Fig. 4b, the line segment (p, q) is seen to intersect the convex hull $\tilde{P}$ of the remaining seven vertices. From $\tilde{P}$ two nonadjacent vertices p and q are chosen and a line segment is formed; the configuration of (p, q) and $\tilde{P}$ is shown in Fig. 4c. As the convex hull $\tilde{P}$ still has more than four vertices, another iteration is run, and thus results in the configuration in Fig. 4d. Since $\tilde{P}$ now has three vertices, its vertices are joined by two line segments, as shown in Fig. 4e. The resulting SLS set for $N = 9$ consists of $S = 5$ line segments, which is minimum. The second example, as shown in Fig. 5, illustrates the application of algorithm mSLS to the even-vertex polyhedron shown in Fig. 1b. After iterations similar to the ones in the previous example, the resulting SLS set (Fig. 4d) for $N = 8$ consists of $S = 5$ line segments, rather than the minimum $N/2$ or $S^* = 4$. For even-vertex polyhedra, algorithm mSLS is not always suboptimal. Applying it to the polyhedron in Fig. 1a would give a diagonal in the base and three SLSs for the remaining tetrahedron, resulting in an SLS set of the minimum size four. The time complexity is examined next. At a first glance, the time-dominant step of algorithm mSLS, the reconstruction of P, could take $O(N \log N)$ time due to the convex hull operation (Edelsbrunner 1987). An observation to be presented shortly reduces the reconstruction time to a constant by providing a criterion for choosing nonadjacent vertex pairs. Before presenting the observation, two supporting lemmae need to be established.

Lemma 4. *For a polyhedron with N vertices, there are at most 3N–6 edges.*

Proof. See, for example, p. 120 of O'Rourke (1994). □

Lemma 5. *There are at least four vertices of a degree less than or equal to 5 for any polyhedron.*

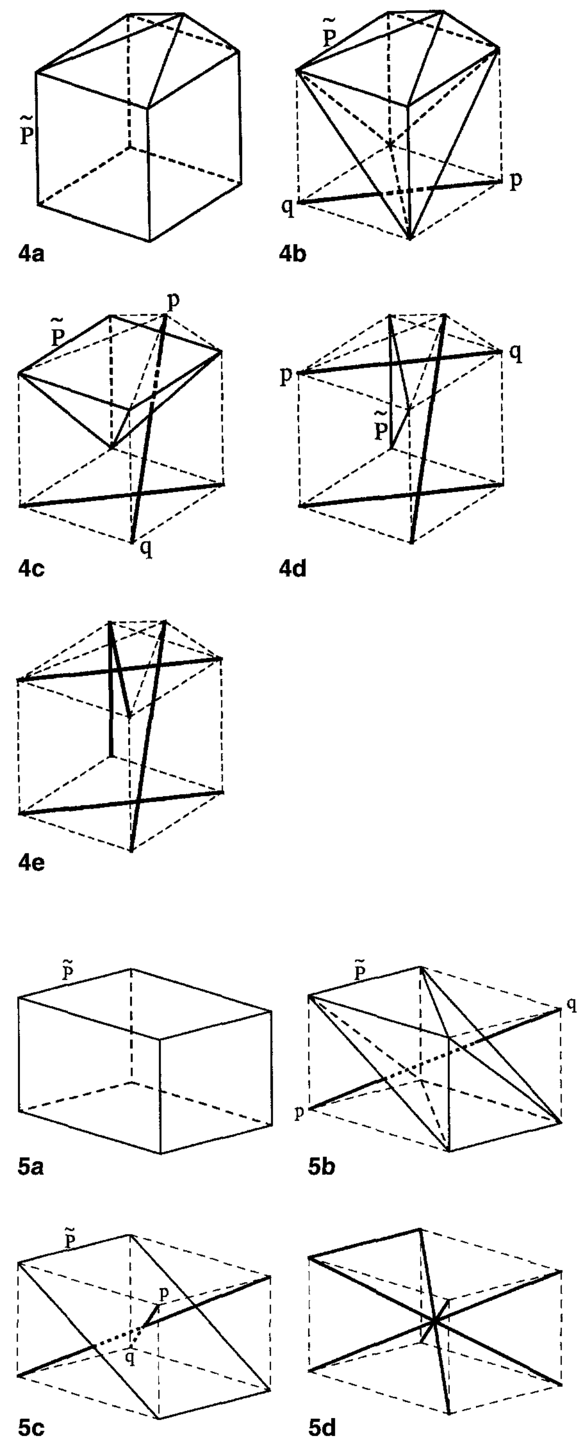


Fig. 4a–e. Applying algorithm mSLS to an odd-vertex polyhedron

Fig. 5a–d. Applying algorithm mSLS to an even-vertex polyhedron

Proof. If the degrees of all vertices but three were greater than 5, then the sum of the degrees, which equals twice the number of edges, would be at least $6(N-3)+3(3)$, since every vertex of a polyhedron must have a degree no less than 3. This sum is $6N-9>2(3N-6)$, which would violate Lemma 4. □

Lemma 6. *For any polyhedron with five or more vertices, there are always a pair of nonadjacent vertices of degrees less than or equal to five.*

Proof. Let $v_{\leq 5}$ denote the subset of vertices containing those of degrees less than or equal to 5, and let $N_{\leq 5}$ denote its size. Also, let p be a vertex in $v_{\leq 5}$ with the smallest degree d^*. A sufficiency claim, $v_{\leq 5}$ *contains a vertex q nonadjacent to p*, is now proved. Because p is adjacent to exactly d^* vertices, the set $v_{\leq 5}$ must contain a vertex nonadjacent to p if $N_{\leq 5}$ is greater than d^*+1. By Lemma 5 and the fact that every nondegenerate vertex has a degree greater than or equal to 3, only three cases are possible: d^* equals 3, 4, or 5. First consider the case $d^*=3$. The lemma holds if $v_{\leq 5}$ contains more than four vertices. By Lemma 5, $N_{\leq 5}$ is at least 4, and $N_{\leq 5}$ is 4 only when $v_{\leq 5}$ contains exactly four vertices of degree 3. To make these four vertices adjacent to each other would require them to form a complete graph and force a disconnected component for any polyhedron with five or more vertices. Therefore, $v_{\leq 5}$ always contains a pair of nonadjacent vertices for $d^*=3$. By Lemma 4, $N_{\leq 5}$ is at least 6, if $d^*=4$, and $N_{\leq 5}$ is at least 12, if $d^*=5$. Thus the result follows immediately for these two cases. □

The reconstruction of a convex hull after removing a vertex requires only the geometric and topological information of the removed vertex and its adjacent vertices. Edges and faces not adjacent to the removed vertex are not affected by the removal and should remain in the reconstructed convex hull. Therefore, the reconstruction after each removal needs to consider only information local to the removed vertex. Thus, the updating can be done in constant time.

3 Algorithm

The algorithm is first described conceptually, followed by details in data structure requirements and analysis for time complexity.

Algorithm mSLS: Minimum spanning line segments

Input: a convex polyhedron P, represented by a set $\mathscr{V}=\{v_1, v_2, \ldots, v_N\}$ of N vertices and their adjacency relations.
Output: $\mathscr{S}$, an SLS set for P containing $\lfloor N/2 \rfloor + 1$ line segments.

begin
  Step 0. Initialize $\mathscr{S}$.
  Step 1. Traverse the representation of P.
    1.1 For each vertex v_i, compute its degree d_i.
    1.2 Classify v_i according to its degree d_i.
  Step 2. Find two nonadjacent vertices v_a and v_b by searching the classified vertices in ascending order of the degrees.
    2.1 Add line segment (v_a, v_b) to $\mathscr{S}$.
    2.2 Delete first the adjacencies of v_a and v_b, and then the two vertices themselves from $\mathscr{V}$.
    2.3 Reconstruct P (i.e., $CH(\mathscr{V}-\{v_a, v_b\})$).
  Repeat this step until the number of vertices in P is no more than 4.
  Step 3. Compute an SLS set for P with no more than four vertices; add the line segments to $\mathscr{S}$.
end

To achieve efficient computation, a data structure containing a dictionary D and an array of linked lists L [3..6] is devised. Each entry $D[i]$ keeps the degree of vertex i, as well as a pointer into L for the vertex. The elements in $L[i]$ are vertices of degree i for $i=3,4,5$ and vertices of degree 6 or more in L [6]. Now algorithm mSLS is described in detail, along with the analysis for the time complexity of each step.

Step 1. Initially D and L are empty. In Step 1, degrees of the vertices are first computed, in linear time by traversing the representation of P, and entered into D. Each vertex v_i is then inserted in L according to its degree stored at $D[i]$, and the pointer at $D[i]$ is assigned. These initial insertions take linear time. An example of this data structure is shown in Fig. 6.

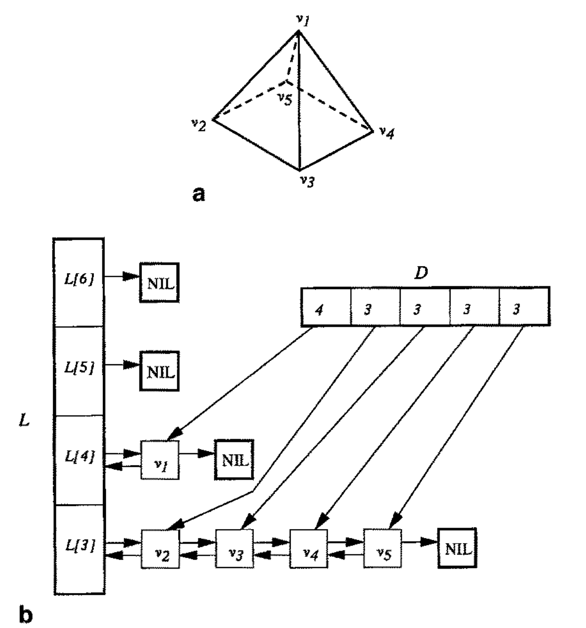


Fig. 6a, b. A data structure to facilitate finding nonadjacent vertices of degrees ≤5: **a** a polyhedral cone; **b** its filled data structure

Step 2. To find a pair of nonadjacent vertices, traverse L [3], L [4], and L [5] in succession until a pair is found. By Lemma 4, finding a pair of such vertices takes a constant time, as at most, six vertices are visited. This selection process is illustrated by Fig. 6. List L [3] is examined, and the first vertex v_2 is picked as v_a. The second vertex stored in L [3], v_3, is next visited and found to be adjacent to v_a. The search then goes on to visit the next vertex v_4, which is not adjacent to v_2 and is thus taken as vertex v_b.

Step 2.1. Inserting a line segment to $\mathscr{S}$, which is initially empty, takes a constant time.
Step 2.2. For some representations of polyhedra such as the winged-edge data structure (Hoffmann 1989), deleting the two vertices v_a, v_b and their adjacencies takes a constant time. As for maintaining D and L, v_a and v_b need to be deleted from L, which takes a constant time by construction. Depending on the implementation, the degrees of v_a and v_b can either be left in D or be deleted in a constant time.

Step 2.3. After a vertex is removed, faces are formed from its adjacent vertices to restore convexity. Since each vertex removed is of a degree no more than 5, and a face needs at least three vertices, there are at most $C(5,3)+C(5,4)+C(5,5)=16$ combinations. Therefore, determining the new convex-hull faces and, in turn, updating the adjacencies in P can both be completed in a constant time. After restoring the convexity for P, up to ten vertices may need to be relocated in L for changes in their degrees. Relocating a vertex in L involves two steps: deleting the vertex at the old position and reinserting it according to its updated degree. Both of these require only a constant time. Therefore, updating the data structure after a reconstruction takes a constant time.
Overall, Step 2 is repeated exactly $\lceil (N-4)/2 \rceil$ times, as two vertices are removed in each execution. As explained in Step 2.1–2.3, each execution of Step 2 takes a constant time. Therefore, the total time complexity for Step 2 is $O(N)$.

Step 3. SLS sets consisting of three and two line segments can be computed in a constant time for polyhedra with four and three vertices, respectively.

This completes the description of the algorithm for computing the SLS sets for polyhedra with N extreme vertices.

Theorem 1. *For a polyhedron of N extreme vertices, algorithm mSLS computes an SLS set of size S, where*:

$$S = \lfloor N/2 \rfloor + 1 \quad \text{if } N \text{ is odd}$$

$$S = (N/2) + 1 \quad \text{if } N \text{ is even}$$

in $O(N)$ time.

4 Concluding remarks

Of particular interest to computer graphics is the notion of spatial arrangements of lines (Edelsbrunner 1987; O'Rourke 1994), which arises in various contexts such as visibility graphs (O'Rourke 1987) and hidden surface removal

(Mckenna 1987). As a novel arrangement, the SLSs capture the essential property of intersection of polyhedral objects and thus provide a new representation.

This paper offers an algorithm that yields the minimum $\lfloor N/2 \rfloor + 1$ number of SLSs when N, the number of extreme vertices, is odd, and a near minimum $(N/2) + 1$ when N is even. This paper also establishes a tight upper bound for the minimum size of an SLS set. Clearly, an algorithm with an output independent of the parity would be desirable. However, such an algorithm implies a procedure to determine whether a polyhedron with an even number of vertices has the minimum size $(N/2)$, which, as dictated by the spatial arrangement of the vertices, might not have a fast solution.

Acknowledgements. The authors thank the referees for their careful reading of the manuscript, insightful comments, and helpful suggestions, which helped shape the presentation.

References

Blum H (1967) A transform for extracting new descriptors of shape. In: Whaten-Dunn W (ed) Models for the perception of speech and visual form. MIT Press, Cambridge, Mass

Blum H, Nagel R (1977) Shape description using weighted symmetric axis features. Patt Recogn 10: 167–180

Bookstein FL (1979) The line skeleton. Comput Graph Image Processing 11: 123–137

Edelsbrunner H (1987) Algorithms in combinatorial geometry, Springer, Berlin Heidelberg New York

Goldak JA, Yu X, Knight A, Dong L (1991) Constructing discrete medial axis of 3-D objects. Int J Comput Geom Appl 1: 327–339

Hoffmann CM (1989) Geometric and solid modeling: an introduction. Morgan Kaufmann, San Mateo, Calif

Hoffmann CM (1990) How to construct the skeleton of CSG objects. Technical Report CSD-TR-1014, Computer Science Department, Purdue University, West Lafayette, IN 47907, USA

Kirkpatrick DG (1979) Efficient computation of continuous skeletons. IEEE 20th Annual Symposium on Foundations of Computer Science. San Juan, Puerto Rico: 18–27

Lee DT (1982) Medial axis transformation of a planar shape. IEEE Trans Patt Anal Machine Intell 4: 363–369

Matheron G (1988) Examples of topological properties of skeletons. In: Serra J (ed) Image analysis and mathematical morphology, vol 2: theoretical advances. Academic Press, London, pp 217–238

Mckenna M (1987) Worst-case optimal hidden-surface removal. ACM Trans Graph 6: 19–28

O'Rourke J (1987) Art gallery theorems and algorithms. Oxford University Press, Oxford, UK

O'Rourke J (1994) Computational geometry in C, Cambridge University Press, Cambridge, UK

Ratschek H, O'Rourke J (1993) Test for intersection between plane and box. Comput Aided Design 25: 249–250

Wang JY, Yuen LD, Woo TC (1993a) A method for testing intersection between plane and polyhedron. Technical Report, GINTIC Institute of Manufacturing Technology, Nanyang Technological University, Singapore

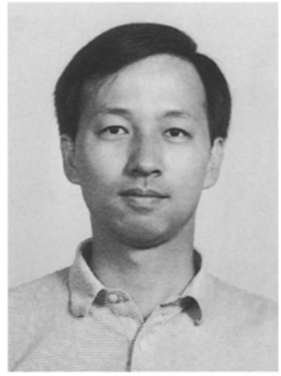
Wang JY, Yuen LD, Woo TC, Chen XM (1993b) The spanning line sgments of a polyhedron. To appear: ASME Trans J Mechanical Design (March 1996) Vol. 118 also available as Technical Report TR93-42, Department of Industrial & Operations Engineering, University of Michigan, Ann Arbor, Mich

Wu AY, Bhaskar SK, Rosenfeld A (1986) Computation of geometric properties from the medial axis transform in $O(n \log n)$ time. Comput Vision Graph Image Processing 34: 76–92

MING-EN WANG is a PhD candidate of Industrial and Operations Engineering at the University of Michigan. His research interest is in the geometric problems that arise in design, manufacturing, and other areas. He received a BS in Mechanical Engineering from the Chiao-Tung University, Taiwan, in 1987, and an MS in Industrial Engineering from Iowa State University, USA, in 1993.

TONY C. WOO is a Professor of Industrial Engineering and holds the John M. Fluke Distinguished Chair in Manufacturing Engineering at the University of Washington. His research is in computational geometry as applied to design and manufacturing.

LIN-LIN CHEN is an Associate Professor of Engineering and Technology. Her research interests are in design for manufacturing, computational geometry, solid modeling, and geometric tolerancing. She received a BS in Industrial Design from Cheng-Kung University, Taiwan, in 1984, and a PhD in Industrial and Operations Engineering from the University of Michigan, USA, in 1992. She worked as a design engineer at Sampo Corporation, Taiwan, from 1984 to 1986, and as a part-time computer programmer in the Computer-aided Engineering Department of the Ford Motor Company from 1987 to 1989.



SHUO-YAN-CHOU is an Associate Professor of industrial management. His research interests are in computational geometry and its applications in design, process planning, manufacturing, and inspection. He received a BBA in Industrial Management from Cheng-Kung University, Taiwan, in 1983, and an MS (1987) and a PhD (1992) in Industrial and Operations Engineering from the University of Michigan, USA.