International Journal of Computational Geometry & Applications

Point and Line Segment Reconstruction from Visibility Information

S. K. Wismath

Department of Mathematics and Computer Science, University of Lethbridge, 4401 University Drive, Lethbridge, AB, Canada, T1K 3M4; E-mail: wismath@cs.uleth.ca. *

Received received date
Revised revised date
Communicated by Godfried Toussaint

ABSTRACT

In general, visibility reconstruction problems involve determining a set of objects in the plane that exhibit a specified set of visibility constraints. In this paper, an algorithm is presented for reconstructing a set of parallel line segments from specified visibility information contained in an extended endpoint visibility graph. The algorithm runs in polynomial time and relies on simple vector arithmetic to generate a system of linear inequalities. A related problem, solvable with the same technique, is the point reconstruction problem, in which the cyclic ordering and the x-coordinates of a set of points is specified.

A second contribution is the definition of an extension of the visibility graph called the Stab Graph, which contains extra visibility information.

Keywords: visibility graph, reconstruction, order types

1. Introduction

There are many problems in computer science that are directly or indirectly concerned with the visibilities inherent among a collection of objects in the plane. Such problems arise in graphics, motion planning, computational geometry, and VLSI design, for example. Although the type of objects and the definition of visibility frequently vary, most results that deal explicitly with visibility issues focus on either the computational or structural properties of visibility. One common theme in the visibility literature is to determine a concise yet sufficiently rich abstraction of visibility information of objects, and to investigate the informational content (and consequent algorithmic issues) provided by the abstraction. The most commonly considered objects in the plane are points, line segments and polygons. Pure visibility information includes the *order type* or various *visibility graphs*, to which is added extra information, for example distance specifications [9], or stabbing information [7],

*A preliminary version of this paper was presented at the 6th Canadian Conference on Computational Geometry, 1994. The author gratefully thanks N.S.E.R.C. for financial assistance.

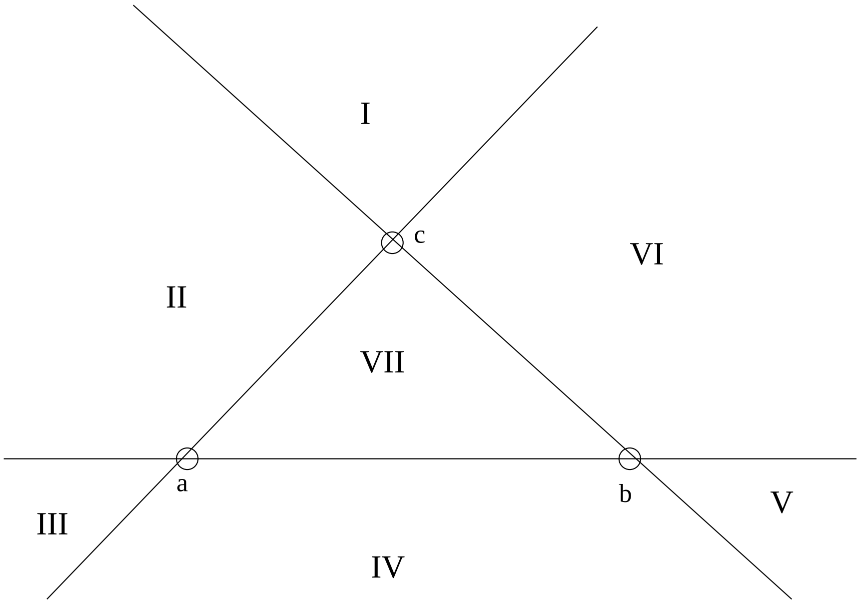

Figure 1: Indistinguishable regions

[11]. The *extraction* of such information from a given set of objects has been well studied; the converse problem has proven more difficult. *Reconstruction* problems require the production of a set of objects (for example, points, line segments, the sides of a polygon) consistent with specified visibility information. The layout of objects reconstructed from pure visibility information is, of course, not unique, but the goal is to group layouts of objects into reasonable equivalence classes that are similarly structured. See O'Rourke's survey paper [14] for a general introduction to the results in this area.

The main result of this paper is an algorithm that reconstructs a set of parallel line segments from visibility information. The degenerate case in which line segments reduce to points is also solved. Both these problems are described in the following subsection and their solution is detailed in section 3. An equally important contribution of this paper is the definition (in the following subsection) of an extended visibility structure called the *Stab Graph*, appropriate for points, line segments, and polygons.

2. Preliminaries and Definitions

Although the main focus of this paper is on visibility of line segments, the degenerate case relates to better known problems involving point sets, including a complexity result establishing the NP-Hardness of the general case.

2.1. Point Reconstruction

Planar configurations of *points* have been studied in a variety of contexts, including arrangements of lines [5], oriented matroids [10] [16], and as allowable sequences

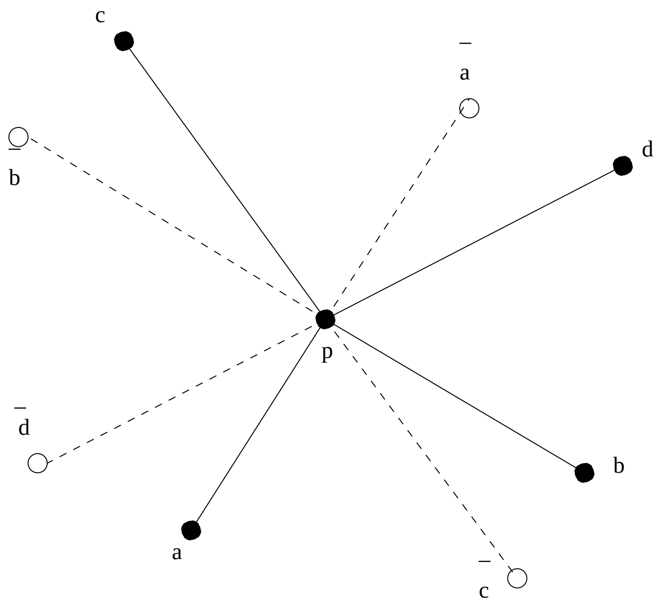

Figure 2: Ordering about point p: $\Pi^{\pi}(p) = c\bar{a}db\bar{c}a\bar{d}\bar{b}$

of permutations [2] [10]. A fundamental notion is that of *order type* which can be described as knowing for each triple of points p, q, r, whether r lies to the left or right of the oriented line $\overline{pq}$ (alternately, whether the triangle p, q, r is oriented clockwise or counterclockwise) †. The two standard encodings of order type information are via an n x n matrix [10] or in dual space as an arrangement of lines [5]. An alternate representation, and the one adopted here, is to specify for each vertex v, the cyclic ordering of the other vertices about v. Denote by $\Pi(p_i)$ the clockwise listing of all other points about p_i. The n Π sequences alone are *not* equivalent to the order type information as demonstrated by figure 1 for example - the lines through three non-collinear points a, b, c, divide the plane into seven regions. A point d placed in any of these regions is distinguishable from the order type information, but regions I, III, and V are indistinguishable from the Π sequences. Extra information is required.

The ordering information for a point p_i represents the cyclic order of all the other points about p_i. The supplemental information to be provided is how the *line* through $p_i p_j$ (for $j \neq i$) partitions the ordering information and is supplied as follows. For each point p_j about point p_i reflect p_j about p_i, thus defining a **shadow point** $\overline{p_j}$, π radians from p_j about p_i and define $\Pi^{\pi}(p_i)$ as the cyclic ordering of the other points *and their shadow points* about p_i. Refer to figure 2.

Remark: Goodman and Pollack [10] use a similar technique in an algorithm to compute their n x n λ matrix. Here the structure is used more directly.

The point reconstruction problem can now be defined as follows.

†Points are assumed to be in general position in this paper, for notational convenience; this assumption is easily removed at the expense of a slightly more detailed account.

Point Reconstruction Problem: Given the cyclic ordering constraints for a set of points : $\Pi^\pi(p_1), \ldots, \Pi^\pi(p_n)$, determine the x and y coordinates of a set of points $P = \{p_1, \ldots, p_n\}$ that satisfy the constraints.

Shor [16] showed that the realizability of points from their order types is NP-Hard via Pseudoline Stretchability. However as will be shown, the Point Reconstruction problem is solvable in polynomial time when extra information is provided, namely, the x coordinate of each point.

Tracked Point Reconstruction Problem: Given the n cyclic orderings of points and their shadows, $\Pi^\pi(p_1), \ldots, \Pi^\pi(p_n)$, and given the x coordinates $X(p_1), \ldots, X(p_n)$, determine y coordinates $Y(p_1), \ldots, Y(p_n)$ of points that satisfy the cyclic orderings.

A solution to the Tracked Point Reconstruction problem follows directly as a special case of a more general visibility problem involving line segments.

2.2. Line Segment Reconstruction

In this paper, it is the reconstruction problem that is solved for a set S of *line segments* to be embedded on specified parallel tracks when an extension of a visibility graph is given. There are very few reconstruction results of this form in the literature. For the case of line segments, related results can be found in [1], [17], and [18]. Variants of the *polygon* reconstruction problem have been considered by Coullard and Lubiw [4] who add distance information, and Everett, Lubiw and O'Rourke [9]. Colley [3] used linear programming to recognize uniform uni-monotone polygons from the visibility graph in a manner similar to that employed in this paper. Recently, Jackson [11] has extended the results of this paper by reconstructing orthogonal polygons from slightly weaker stabbing information.

A few papers have focussed on defining reasonable structures for extending the standard visibility information, most notably by: stabbing line information (Everett, Hurtado and Noy [7]), vertex-edge information (Streinu and O'Rourke [13]) and tangent information (Pocchiola and Vegter [15]). Efficient representations of visibility information were given by Kant [12]. An alternate structure which is a natural extension of the visibility graph and which is appropriate for points, line segments and polygons is the *Stab Graph* defined below.

Given a set S of n disjoint line segments in the plane, the **endpoint visibility graph** $G_e(S)$ consists of $2n$ vertices corresponding to the endpoints of S and an edge set representing pairs of visible endpoints: vertices u and v of $G_e(S)$ are joined by an edge if and only if the corresponding endpoints of S are **visible** (i.e. connectable by a line segment that intersects no other segment of S). Traditionally, this edge set is augmented by the segments themselves, i.e. if u and v are endpoints of the same segment of S, then (u, v) is included as an edge. A slightly stronger form of $G_e(S)$, and the one assumed here, is that for each endpoint v of S, the circular ordering of the endpoints visible to v is known.

For an endpoint u visible to v, define *Stab* (u, v) as the name of the first segment of S encountered by the ray from v in the direction of $\vec{uv}$; the symbol "∞" will be

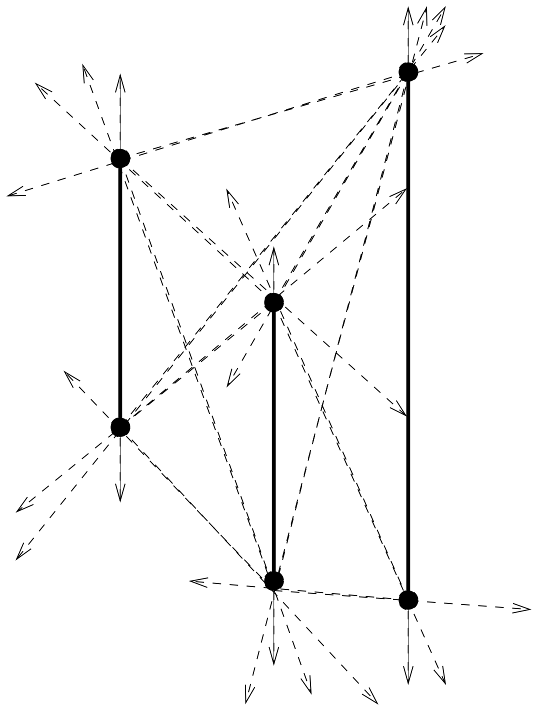

Figure 3: The Stab graph for a small set of segments

used, if this ray encounters no segment of S. Define the **Stab Graph** $G_e^{\pi}(S)$ as the graph $G_e(S)$ supplemented with this *stab* information, namely for each edge (u, v) of $G_e(S)$ the two stabs *stab* (u, v) and *stab* (v, u) are stored at vertices v and u respectively, inserted in sorted order. Thus, the cyclic ordering about each vertex v of $G_e^{\pi}(S)$ consists of pairs of "visibility vectors", spaced π radians apart. See figure 3 for an example of the stab graph. Given $G_e^{\pi}(S)$, the actual *visibility polygon* about any endpoint v, can be computed by a single rotational sweep about v.

Note that although G_e^{π} is not strictly speaking a graph, it is sufficiently graph-like that only minor changes to a data structure (for example an adjacency list representation with shadow links as described in the previous section) are required to accommodate the information and still retain the standard graph operations. We maintain this illusion and refer to both visibilities and stabs as *edges* of G_e^{π}, distinguishing between the two when necessary. Let E denote the number of edges in G_e; then G_e^{π} contains $3E$ edges.

The general line segment reconstruction problem is to determine a set $S = \{s_1, s_2, ...s_n\}$ of non-intersecting line segments in the plane, whose visibility graph is isomorphic to a given target visibility graph, and as a consequence of Shor's result [16], this problem is NP-Hard. In this paper, we consider the restricted problem in which the segments are to be embedded on specified vertical tracks, defined by the lines $X = x_1,\ X = x_2,\ \ldots\ , X = x_n,\ (x_i \neq x_j)$ and the target visibility graph is the stab graph G_e^{π} described in the previous section. Initially, assume that the lengths of each segment $l_1, l_2, \ldots\ l_n$ are also known, and then only the bottoms of the segments, denoted by $y_1, y_2, \ldots y_n$ must be determined, and segment s_i will then have bottom coordinates (x_i, y_i) and top coordinates $(x_i, y_i + l_i)$. In G_e^π it is the *endpoints* that are stored as vertices; an endpoint p may correspond to the top or bottom of a line segment. For notational convenience, let (x_p, y_p) represent the coordinates (yet to be determined) of the appropriate endpoint of S [‡].

> **Tracked Line Segment Reconstruction Problem:** Given a stab graph G_e^π, and a sequence of x-coordinates $x_1, x_2, ... x_n$, and a sequence of lengths $l_1, l_2, ... \ l_n$, determine a set of non-intersecting line segments $S = \{s_1, s_2, ... s_n\}$ such that s_i has length l_i and x-coordinate x_i and whose stab graph is isomorphic to G_e^π.

3. Line Segment Reconstruction from G_e^π

The technique for determining actual coordinates for the endpoints of the line segments involves generating a system of linear inequalities. First note that in vector terms, for any endpoint z, the intersection of the line through endpoints p and q with the track x_z is given by:

$$p + \frac{x_z - x_p}{x_q - x_p}(q - p)$$

Since the x values are known, it is the y values that are to be constrained. Given a target visibility graph G_e^π, the coordinates of the endpoints of S are determined as follows. The visibility information of G_e^π must be respected in two ways. In the embedding of S, visible endpoints must be visible, and non-visible endpoints must be blocked by some segment. The visibility edges and stab edges contribute similar but slightly different constraints. If u and v are endpoints of the same vertical line segment with u above v, those neighbours of u that lie strictly to the left of u appear clockwise between v and $\text{stab}(v, u)$ in G_e^π. Denote by $L(u)$ (respectively $R(u)$), the clockwise ordered set of neighbours (including stabs) of vertex u to the left (respectively right) of u in G_e^π. Consider $L(u)$, the ordered edges about u to the left.

First note that if $stab(v, u) = s_i$, as in Figure 4a, for example, then the top and bottom of segment s_i must be constrained relative to the line through u and v:

$$y_i < y_u + \frac{x_i - x_u}{x_v - x_u}(y_v - y_u)$$

$$y_i + l_i > y_u + \frac{x_i - x_u}{x_v - x_u}(y_v - y_u)$$

Stabs to ∞ may safely be ignored as they generate no direct constraint. Label the inequalities produced in this manner, Type 1 inequalities. Since each visible pair of vertices produces two stabs, there are at most $4E$ such linear inequalities and they can be generated in $O(E)$ total time.

[‡]More formally, the coordinates of the endpoint can be specified by introducing a function $seg(p)$ which returns the segment containing p

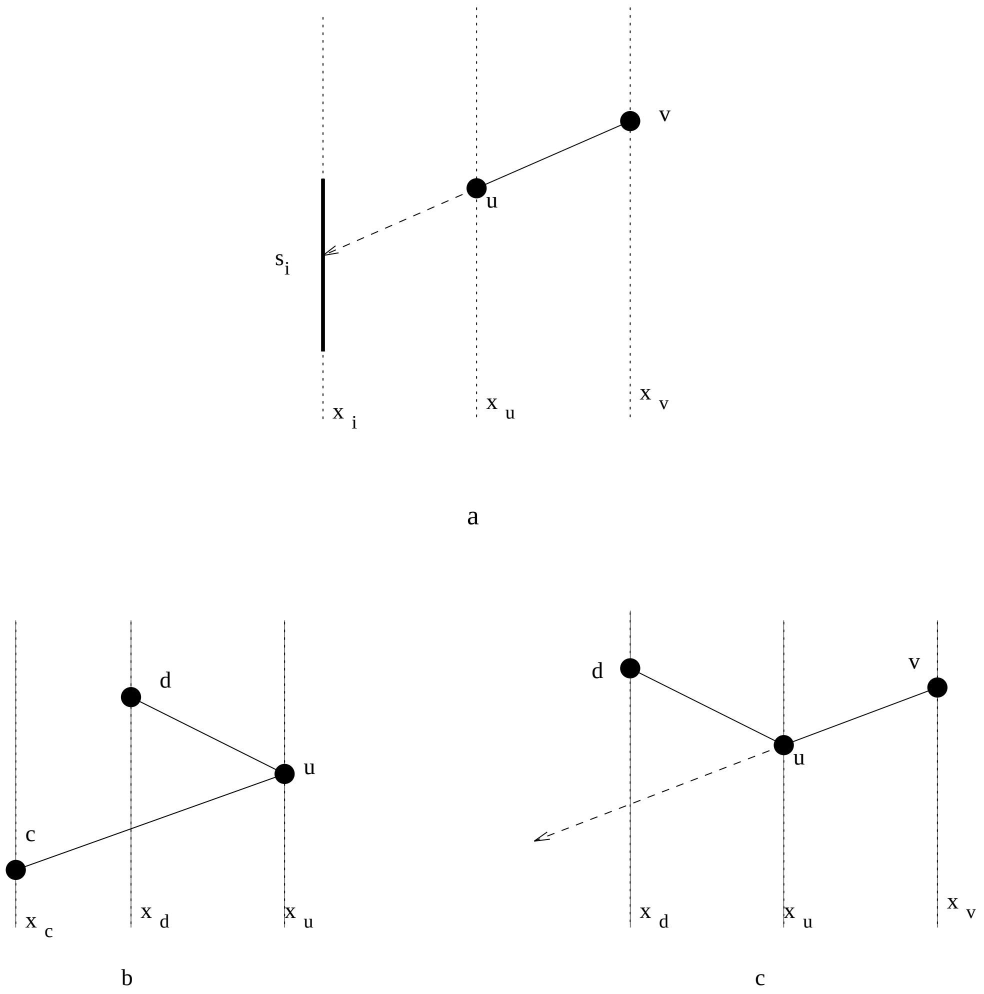

Figure 4: Constraints

Two consecutive visibility edges in $L(u)$, as in figure 4b, represent a relationship among the endpoints u, c, and d. Assume $x_c < x_d$. Then d must, in the final layout, lie above the line through c and u, which can be expressed as:

$$y_d > y_u + \frac{x_d - x_u}{x_c - x_u}(y_u - y_c)$$

Similarly, if a visibility edge (d, u) is clockwisely consecutive to a stab edge $stab(v, u)$ in $L(u)$, as in Figure 4c, then d is constrained as:

$$y_d > y_u + \frac{x_d - x_u}{x_v - x_u}(y_v - y_u)$$

Note that not all pairs of edges in $L(u)$ need be considered - only consecutive pairs of edges. Furthermore, pairs of consecutive *stab* edges do not generate inequalities of this form. Such inequalities are labelled Type 2 and can be generated in $O(E)$ total time by a rotational sweep about $L(u)$ and $R(u)$ for each vertex u of G_e^π. There are $3E$ edges in G_e^π and hence fewer than $3E$ type 2 inequalities.

In total, if there are E pairs of visible endpoints in G_e^π, then at most $7E$ linear inequalities are generated. However, obtaining a solution to this set of linear inequalities is not trivial. One technique is to create a fictitious objective function and solve the associated linear programming problem. Since linear programming is known to be in the complexity class P, thus the line segment reconstruction problem also has a polynomial time solution. Moreover since techniques, such as the simplex method, work efficiently in practice, a solution to these inequalities can, in fact, be obtained (on average) quickly.

Alternately, it is possible to convert the system of linear inequalities to a system of linear *equalities* by introducing a slack variable into each inequality. Each slack variable must be strictly greater than zero and has a geometric significance, namely the vertical distance of the constrained endpoint from the associated visibility stab or line. Solving the resulting system of equalities (by Gaussian Elimination, for example), yields a parametric solution to the given reconstruction problem which represents the set of *all* valid layouts. Unfortunately, determining even a single solution *with all slack variables strictly greater than zero*, is, in general, difficult (although solvable in polynomial time).

Finally, note that the inequalities generated are of a highly restricted form - each contains either three or four unknowns, and thus there may exist more efficient techniques for obtaining solutions directly.

3.1. Proof of Correctness

Let S' be the associated set of line segments for a particular solution to the set of linear inequalities generated by the algorithm. Note that since the segments of S' are embedded on parallel tracks, no pair of segments can intersect in S'. The following three lemmas prove that $G_e^\pi(S')$ is equivalent to the given target visibility graph G_e^π. Since G_e^π is the stab graph of some set of segments, let S be a set of line segments with $G_e^\pi(S){=}G_e^\pi$. We now treat S as a concrete realization of G_e^π. The following definition will be useful in the lemmas. Refer to figure 5. Let p and q be a pair of endpoints in S with $x_p < x_q$ and define **Upperchain**(p, q) as the sequence of endpoints $p = u_0, u_1, ..., u_k, u_{k+1} = q$ constructed as follows.

- $u_0 := p$
- If u_i is visible to q then $u_{i+1} := q$ else
- u_{i+1} is the first top endpoint z visible to u_i and counterclockwise of the line through $u_i q$ that satisfies:
  - x_z lies between the track of u_i and the track of u_q
  - and $stab(u_i, z) = s$, where $s = \infty$ or $x_s \geq x_q$.

Note that if p and q are not visible then $k \geq 1$. In a similar fashion a lower chain of vertices $l_0, ...l_{j+1}$ can be defined and then the rays $p\vec{u}_1$ and $p\vec{l}_1$ form a wedge shaped shadow region in which all endpoints w with $x_w \geq x_q$ are not visible to p.

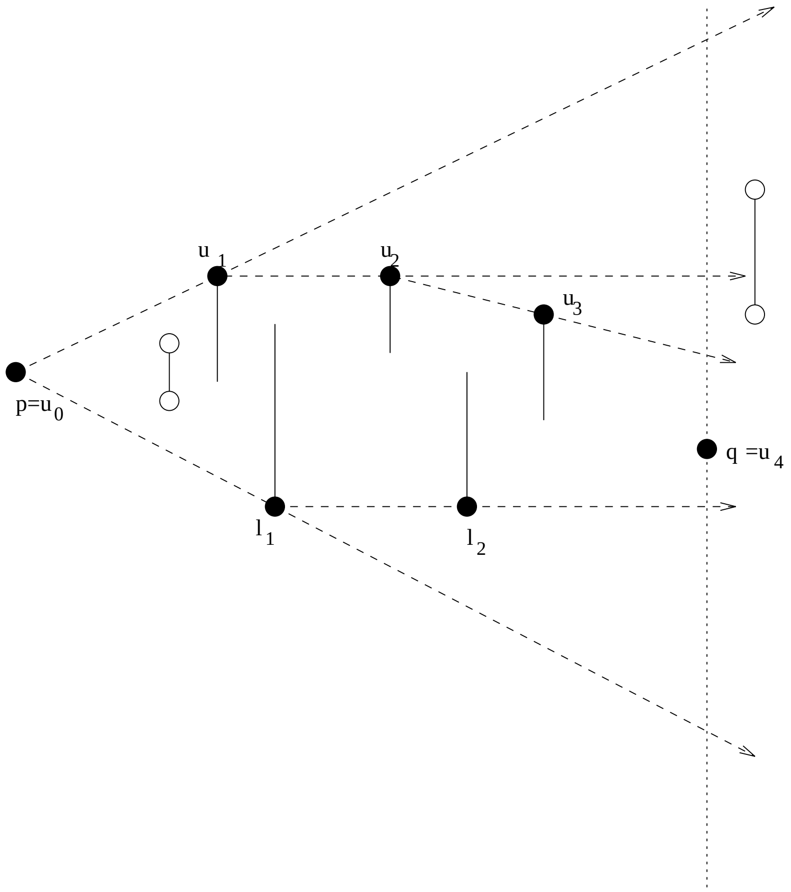

Figure 5: Upperchain(p, q)

Lemma 1 *If (p, q) is a visibility edge in G_e^π, then p' and q' are visible endpoints in S'.*

Proof: Let p and q be a pair of visible endpoints in S such that p' and q' are not visible in S'. Let the leftmost segment in S' that blocks the visibility of p' and q', have top and bottom endpoints u' and v'. Assume, without loss of generality, that in S, the segment (u, v) lies below the (p, q) visibility. If p sees u after q in $R(p)$, then there is a linear inequality forcing u to lie below the line through p, q, which contradicts the assumption that u' is above the line p', q' in S'. If p does not see u directly, then consider the Upperchain(p, u), as in figure 6. Then there is a sequence of type 2 linear inequalities generated at the u_i's, ultimately forcing u to be below the line through p, q. Hence u' is similarly constrained to be below the line segment (p', q') in S' and therefore, the segment (u', v') can not block the visibility of p' and q'.

□

Lemma 2 *If $stab(p, q) = u$ in G_e^π, then $stab(p', q') = u'$ in S'.*

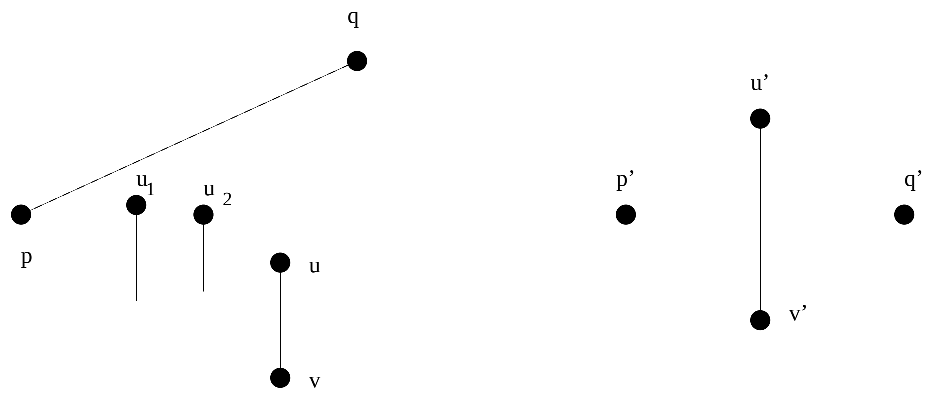

Figure 6: p and q visible; p' and q' not visible

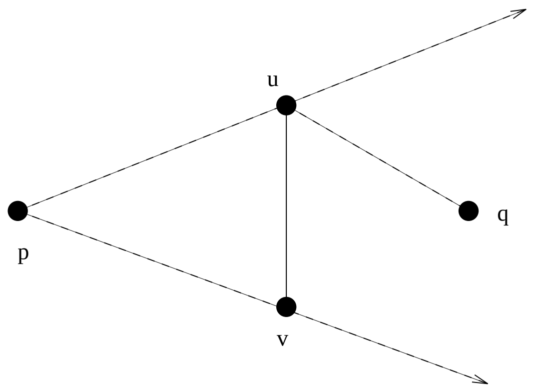

Figure 7: Simple case

Proof: The type 1 inequalities ensure that the line through $p'q'$ intersects segment u'. The inequalities generated when processing $R(q)$ ensure that the line segments between x_q and x_u are constrained to lie above or below the stab visibility.

□

It is more awkward to show that no new visibilities appear in S'. If (p, q) is not an edge in G_e^π, then it is not evident where the line through $p'q'$ would appear in the cyclic ordering of p'. Since this is equivalent to knowing the order type information, it is necessarily ambiguous in the system of linear inequalities if there are endpoints that are not visible.

Lemma 3 *If (p, q) is not an edge of G_e^π, then p', q' are not visible endpoints in S'.*

Proof: Consider a pair of endpoints p and q in S such that p is not visible to q. If the visibility between p and q is blocked by a single line segment, $\overline{uv}$, as in figure 7, then it is clear that there exists a constraint (generated at u), forcing q to lie below the line through $\overline{pu}$ and similarly above the line through $\overline{pv}$.

The general case in which u or v is not visible to q, involves a simple iterated version of this argument. Consider Upperchain(p, q) as in figure 5. The rays $u_i\vec{u_{i+1}}$ for $i := 0$ to k cross the line $X = x_q$ in a monotonely decreasing fashion and thus the corresponding linear inequalities generated by the stab graph G_e^π must ultimately constrain q to lie below the ray $p\vec{u}_1$. In a similar fashion, a lower chain can be determined in S, which constrains q to lie above the ray $p\vec{l}_1$, and hence in a shadow region not visible to p. The same sequence of constraints forces q' to lie in a shadow region not visible to p'.

□

As a consequence,

Theorem 1 *The Tracked Line Segment Reconstruction Problem is in the complexity class P.*

Note that the heights of each line segment are also linear in the resulting inequalities and thus could be considered as unknowns (with the restriction that $l_i > 0$).

4. Applications

As noted previously, the degenerate case of the problem (with $l_i = 0$) is an interesting point reconstruction problem. Note that for points in general position, no type 1 constraints are produced and thus there are at most $3\binom{n}{2}$ inequalities.

Corollary 1 *The Tracked Point Reconstruction Problem is in the complexity class P.*

A second application of the main result is as a generalization of a graph representation problem. Typically, such problems ask if some graph can be represented such that the vertices correspond to objects and the edges correspond to an underlying relation among objects; *characterizing* the graphs that are representable is generally the goal. The results presented here are relevant to the case when the objects are specified to be parallel line segments and the rotational view of each vertex endpoint is also given.

5. Conclusions and Open Problems

The visibility graph alone does not contain sufficient information for reconstruction purposes. A major contribution of this paper is the definition of a *stab graph* which extends the traditional visibility graph by including extra visibility information. The main result here is a polynomial time algorithm to reconstruct a set of parallel line segments from visibility and stab information when the segments lie on specified tracks. The associated point reconstruction problem also has a polynomial time solution.

The reconstruction problem for line segments is still open for the following two related cases:

- for parallel line segments without specified track information.

- for line segments on non-parallel tracks (for example on two sets of parallel tracks).

Acknowledgment The anonymous referees made several constructive suggestions to improve the presentation of the paper.

References

1. T. Andreae, Some Results on Visibility Graphs, *Discrete Applied Mathematics*, Vol. 40, No. 1, (1992), 5-18.
2. H. Bieri, P. Schmidt, On the Permutations Generated by Rotational Sweeps of Planar Point Sets, *Proc. 8th Canadian Conference on Computational Geometry*, Ottawa, 1996, 179-184.
3. P. Colley, Visibility Graphs of Uni-monotone Polygons, M.Sc. thesis, U. of Waterloo, 1991.
4. C. Coullard, A. Lubiw, Distance Visibility Graphs, *J. of Computational Geometry and Applications*, Vol. 2, No. 4 (1992) 349-362.
5. H. Edelsbrunner, J. O'Rourke, R. Seidel, Constructing Arrangements of Lines and Hyperplanes with Applications, *SIAM J. Computing* 15 (1986) 341-363.
6. H. Everett, D. Corneil, Recognizing Visibility Graphs of Spiral Polygons, *J. of Algorithms* **11** (1990), 1-26.
7. H. Everett, F. Hurtado, M. Noy, Stabbing Information of a Simple Polygon, *8th Canadian Conference on Computational Geometry*, Ottawa, 1996, 74-79.
8. L. Jackson, Polygon Reconstruction from Visibility Information, M. Sc. Thesis, April 1996, U. of Lethbridge Technical Report TR-CS-01-96.
9. H. Everett, A. Lubiw, J. O'Rourke, Recovery of Convex Hulls from External Visibility Graphs, *Proceedings of the 5th Canadian Conference on Computational Geometry*, Waterloo, (1993), 309-314.
10. J. Goodman, R. Pollack, Multi-dimensional Sorting, *SIAM J. Computing* 12 (1983) 484-507.
11. L. Jackson, S. Wismath, Orthogonal Polygon Reconstruction, *Proc. 8th Canadian Conference on Computational Geometry*, Ottawa, 1996, 44-49.
12. G. Kant, A More Compact Visibility Representation, *International Journal of Computational Geometry and Applications*, Vol. 7, 1997, 197-210.
13. J. O'Rourke, I. Streinu, The Vertex-Edge Visibility Graph of a Polygon, *Computational Geometry: Theory and Applications*, 10, 1998, 105-120.
14. J. O'Rourke, *Computational Geometry Column 18*, SIGACT News, Vol.24, No. 1 (1993), pp.20-25.
15. M. Pocchiola, G. Vegter, Order Types of Configurations of Disjoint Convex Plane Sets, *Proceedings of the 6th Canadian Conference on Computational Geometry,* Saskatoon, 1994.
16. P. W. Shor, Stretchability of Pseudolines is NP-Hard, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Vol. 4, 1991, pp. 531-554.
17. R. Tamassia, I. Tollis, A Unified Approach to Visibility Representations of Planar Graphs, *J. of Discrete and Computational Geometry* **1** (1986), 321-341.
18. S. K. Wismath, Characterizing Bar Line-of-Sight Graphs, *Proc. 1st ACM Symposium on Computational Geometry*, Baltimore (1985), 147-152.