Dent and Staircase Visibility*

(Extended Abstract)

Derick Wood† Peter Yamamoto‡

1 Introduction

Computing the visibility polygon from a point in a polygon, sometimes called the hidden line problem, is a fundamental problem in computational geometry. This problem has been studied in the framework of various types of visibility, including staircase, link, and restricted-orientation. Our research is a generalization of two recent lines of research: orthogonal staircase visibility problems [SRW91] and link distance problems [dB91].

We define the notions of **dent length** and **staircase length** of orthogonal paths. We introduce **dent visibility** and extend staircase visibility to multi-staircase visibility. We provide an algorithm to compute dent regions that runs in $O(n + d\log d)$ time and $O(n)$ space where d is the number of dents in a polygon with n vertices. We use the algorithm to compute the dent-visibility polygon, $VP(p)$, of a query point p, in $O(|VP(p)| + \log(n) + d)$ time, and also to solve several other fundamental dent-visibility problems. The concepts of dent-length and staircase-length raise other geometric and visibility issues.

2 Dents and Staircases

In a simple orthogonal polygon, a **dent edge** is any edge with two reflex vertices. A **dent chord**, formed by extending a dent edge in both directions with **dent rays** until they hit the polygon boundary, divides the interior of the polygon into three types of regions: the regions **above** the dent edge, and the regions below and to each side of the dent edge. The problem of determining these regions and their intersections is called the dent decomposition problem and is used in some polygon covering problems [CR89, BS92].

We extend the notion of a dent in a polygon to a dent in a path. In this case, a **dent segment** is a segment of the path such that its vertices are both right turning or both left turning. The **dent length** of a path is the number of dent segments in the path. We define the **orthogonal dent distance**, or simply dent distance, of two points as the minimum dent length over all orthogonal paths connecting the two points.

An **orthogonal staircase path**, or simply staircase, is an orthogonal path such that the path is monotone with respect to the axes [SRW91]. We define the **staircase length**

[1]This work was supported by the Natural Sciences and Engineering Research Council of Canada and the Information Technology Research Centre of Ontario.

[2]Department of Computer Science, University of Western Ontario, London, Ontario, N6A 5B7 Canada. email: dwood@@watdragon.uwaterloo.ca.

[3]Department of Computer Science, University of Waterloo, Waterloo, Ontario, N2L 3G1 Canada. email: pjyamamoto@@watdragon.uwaterloo.ca.

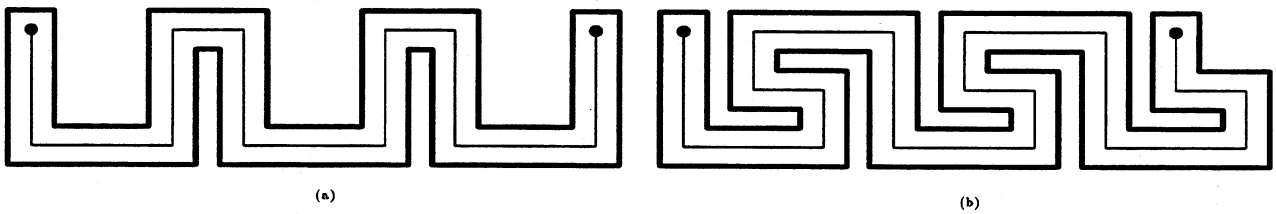

Figure 1: A comparison of the staircase length and the dent length of path reveals geometric information about the path. The paths in the two polygons have a staircase length of 6; however, the path in polygon (a) has a dent length of 5 while the path in polygon (b) has a dent length of 10. This tells us that the path in (a) has no oppositely oriented staircases while the path in (b) has all oppositely oriented staircases.

of a path as the minimum number of staircases joined at their endpoints which decompose the path. We define the **staircase distance** of a path between two points as the minimum staircase length over all orthogonal paths between the two points. Except for 0-dent and 1-staircase length paths, the dent-length and staircase-length of a path are, in general, not equal since there may be one or two dents for two consecutive staircases (see Figure 1).

We define **staircase visibility** between two points as follows: two points a and b staircase-see each other if there exists a staircase connecting the two points which lies entirely in the polygon. This definition may be parameterized by a length as is done for k-link visibility. Two points s-staircase see each other if the staircase distance between them is at most s. **Dent visibility** is similarly defined: two points d-dent see each other if the dent distance between them is at most d.

In order to solve dent-visibility problems, we first identify the set of dent regions by using a $O(n+d\log d)$-time and $O(n)$-space topological sweep of the polygon, where d is the number of dents in the polygon. While d may be $O(n)$, the algorithm illustrates how the algorithm is dependent on the geometric complexity of the polygon. We use this preprocessing algorithm to solve several fundamental dent visibility problems.

3 Sweeping Dents

First note that we are only interested in computing the dent regions for each dent. We do not determine the intersections among the dent regions as in the dent decomposition of the polygon. In this sense, the preprocessing algorithm is a special ray-shooting query problem where the rays are shot out from dent edges (or more arbitrarily anywhere from the boundary) parallel to the axes. A naive approach to the problem would have a logarithmic cost in the size of the polygon for each query. Our algorithm shows that the actual complexity of the query is dependent on the geometric structure of the polygon; in particular, it is dependent on the geometry of the dents. The polygon sweep is topological in the sense that a set of vertical lines independently sweep over disjoint subpolygons but synchronize at special

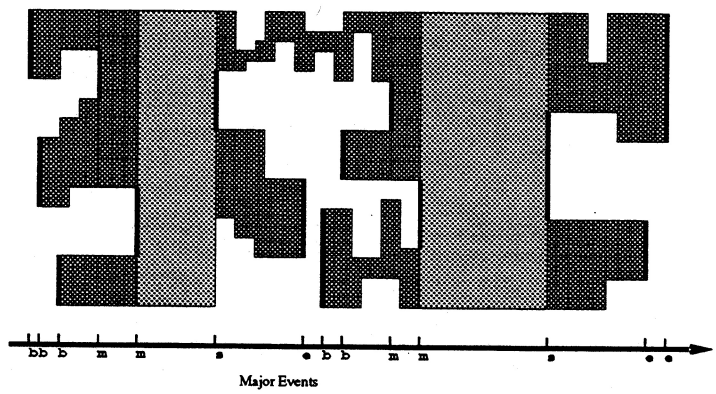

Figure 2: The channel decomposition is determined by four types of edges (drawn as thick edges): **b** (begin), **e** (end), **m** (merge), and **s** (split) edges. Channel sweeps may be performed independently of each other although all sweeps must be synchronized at major events.

events.

In the following discussion, we make the following assumptions. We assume the polygon boundary is labeled in a clockwise orientation. Hence an edge may be pointing up, down, left, and right, and has a **previous** and **next** edge. When we refer to a dent ray, we assume that it is pointing in the direction of the sweep.

3.1 Major Events

In order to compute the set of dent regions, we sweep the polygon in two directions: left-to-right and right-to-left. Since the two sweeps are similar, we assume that the sweep direction is left-to-right. The goal is to find all dent rays pointing in the direction of the sweep and to find the first polygon edge which each ray intersects (called the obstructing edge). As a consequence of the sweep, we also compute the intersection of vertical dent rays and the polygon boundary. We assume pseudo-vertices are inserted at the intersection points; this introduces new pseudo-edges but we make no distinction between such edges and the original edges of the polygon.

The polygon is decomposed by vertical chords into disjoint x-monotone subpolygons called **channels** which are swept independently of each other (see Figure 2). Channels start, end, split, or merge at **major events** which are associated with four, respectively named, types of edges in the polygon.

Convex up edges are **begin edges**. Convex down edges are **end edges**. Reflex down edges are **split edges**. Reflex up edges are **merge edges**. These edges are easily identified by a sequential scan of the polygon edges (see Figure 2). Also note that the split and merge edges are exactly the set of all vertical dent edges.

The next and previous edges of a begin edge are the first edges in the upper and lower chains, respectively, of a new channel (see Figure 3). The channel extends until the next

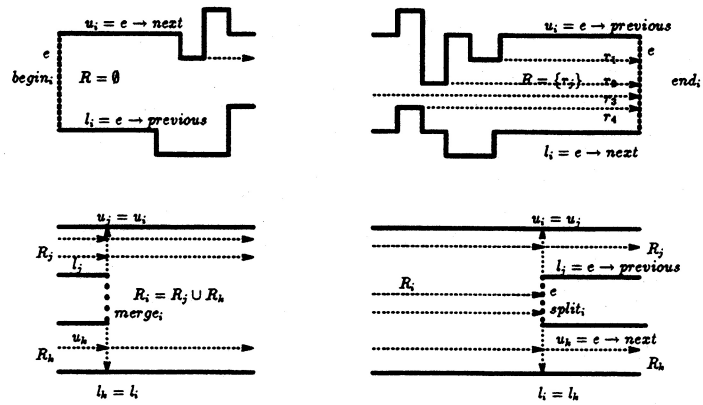

Figure 3: begin, end, merge, and split edges determine the beginnings and ends of channels.

edge on one of the chains is an end or a merge edge, or until an edge of the polygon lies in between the upper and lower chain. An end edge ends a channel since it is an edge in between the upper and lower chains. Since both upper and lower chains terminate in the end edge no new channel is created.

A split edge ends the current channel since it lies between the two chains, but it also defines two new channels. One channel with an upper chain starting on the current edge on the upper chain and the lower chain starting with the edge previous to the split edge. The other channel with the upper chain starting with the edge next to the split edge and the lower chain starting with the current edge on the lower chain.

Suppose that the next edge on an upper chain of the current channel is a merge edge. In this case, the edge next to the merge edge is the last edge in the upper chain of an **adjacent channel**. The upper and lower chains of the adjacent channels terminate at the merge edge and the two channels merge into one new channel defined by the lower chain of the current channel and the upper chain of the adjacent channel.

3.2 Channel Sweeps

During a channel sweep we maintain a set of dent rays: we insert into the set new dent rays and we delete from the set dent rays which hit the channel boundary. The potential obstructing edges can be divided into three distinct groups: end edges, split edges, and others. End and split edges are handled during the processing of major events. The other edges are non-convex and non-reflex edges of the channel boundary.

Some, or all, of the dent rays may pass through the channel unobstructed. The end of a channel can be caused by an end edge in which case all the remaining rays must hit the end edge, or by a merge edge in which the set of unobstructed rays from the two adjacent channels merge, or by a split edge in which case some of the rays may intersect the split edge while others pass above or below the split edge. In the latter two cases, we may start sweeping a channel with a set of rays from a previous channel.

We assume we are given a (possibly empty) set of dent rays, R, upper and lower starting points on edges of the polygon, and the end coordinate of the channel. Finally, we assume that R initially contains all unobstructed rays lying between the two start points. In the following discussion we assume without loss of generality that a (possibly) artificial vertex has been inserted at the end coordinate on the upper and lower chains; this just lets the discussion refer to an "edge" without having to worry about the end cases in which only a portion of an original polygon edge may be considered.

The sweep proceeds by sequentially processing an edge at a time along the two chains in order of their rightmost vertices (ties can be broken arbitrarily). If the current edge initiates a dent ray, then insert the dent ray into R. Otherwise check if the edge is a potential blocking edge (down edges on the upper chain, and up edges on the lower chain) and, if so, then determine the intersections with rays in R and delete those rays from R. Otherwise the edge is not of interest. Proceed with the next leftmost edge.

3.3 Sweeping the polygon

We now describe how the channels sweeps are synchronized and how the channels and set of rays are merged, split, and terminated at major events. First, we examine the edges of the polygon noting the start, end, split and merge, edges. We sort these edges with respect to the sweep direction and process them in order. Note that the leftmost edge is a begin edge. The procedure depends on the the type of major event being processed. After the procedure is finished, we sweep all current channels up until the sweep coordinate of the next major event.

At a begin edge we initialize a new channel, $C_i = (c_{i_u}, c_{i_l}, R_i)$, with the adjacent edges c_{i_u} and c_{i_l} (upper and lower chains respectively) and an empty dent ray set, R_i. We then insert it into the set of current channels $C = \{C_i\}$.

At an end edge, we determine the appropriate channel C_i of C, and if the dent ray set R_i is not empty, we compute the intersections of the rays with the end edge and discard the channel.

At a merge edge, we need to determine the two channels above, C_i, and below, C_j, the merge edge. We compute the intersection of the vertical dent rays from the merge edge with the current edges of the chains c_{i_u} and c_{j_l}. We then create a new channel $C_k = (c_{i_u}, c_{j_l}, R_i \cup R_j)$ and delete the channels C_i and C_j.

At a split edge with previous edge e_l and next edge e_u, we first need to determine the channel C_i to be split. Next, we need to divide R_i into three sets: rays intersecting the split edge, rays R_{i_u} above the split edge, and rays R_{i_l} below the split edge. We compute the intersections of the rays and the split edge. We create two new channels $C_j = (c_{i_u}, e_u, R_{i_u})$ and $C_k = (e_l, c_{i_l}, R_{i_l})$ and delete channel C_i.

4 Applications

We use the preprocessing algorithm to solve several fundamental dent-visibility problems. First, with minor modifications to the preprocessing algorithm, we compute the dent decomposition intersections during the sweep; once again, the worst case time complexity is $O(n^2)$ but the algorithm's performance, $O(n + d^2)$, is sensitive to the dent complexity of the polygon. Second, we compute the d-dent or s-staircase visibility polygon $VP(p)$, of a query point p, in $O(|VP(p)| + \log(n) + d)$ time. The algorithm follows only the actual boundary of the visibility polygon $VP(p)$ requiring $O(|VP(p)|)$ time. The $log(n)$ term is the initial point location and the d term is due to "false checks" which may not be avoidable without affecting the preprocessing time.

5 Conclusions

We introduce a new type of visibility based on dents and provide a topological sweep algorithm to compute dent regions that runs in $O(n + d\log d)$ time and $O(n)$ space. We use the dent region computation as a preprocessing step to solve several dent-visibility and staircase-visibility problems. The dent decomposition of a polygon provides essential visibility information in algorithms for computing covers for orthogonal polygons. Our research further emphasizes the relationship between dents and the geometric complexity of orthogonal polygons.

Many open dent visibility problems remain to be considered. The algorithms presented here are not necessarily optimal and it would be interesting to either provide improvements or lower bounds to the problems. We are currently working on the dent kernel, dent diameter, and dent center problems.

References

[BS92] David Bremmer and Thomas Shermer. Properties and applications of restricted-orientation point visibility graphs. Manuscript, School of Computing Science, Simon Fraser University, 1992.

[CR89] J. Culberson and R. A. Reckhow. Orthogonally convex coverings of orthogonal polygons without holes. *J. Comput. Syst. Sci.*, 39:166–204, 1989.

[dB91] M. de Berg. On rectilinear link distance. *Comput. Geom. Theory Appl.*, 1:13–34, 1991.

[SRW91] S. Schuierer, G. J. E. Rawlins, and D. Wood. A generalization of staircase visibility. In *Proc. 3rd Canad. Conf. Comput. Geom.*, pages 96–99, 1991.