Topologically sweeping the visibility complex of polygonal scenes

RIVIÈRE Stéphane *

1 Introduction

In order to do efficient visibility computations in the plane, one must deal with *maximal free line segments*, that is line segments of maximal length in free space. To that end, [PV93b] introduce, for scene of convex curved objects, a new data structure, the *visibility complex*, whose elements represent classes of maximal free segments having the same visibilities. In particular the visibility complex contains the visibility graph.

We have studied the visibility complex for general polygonal scenes, and have devised an algorithm for topologically sweeping the visibility complex (and so for computing the visibility graph with the same complexity) in optimal time $O(n \log n + m)$ and optimal space $O(n)$, where n is the total number of polygon vertices, and m the size of the visibility graph. Since it is a sweep algorithm, it improves the space storage of [GM87] for computing the visibility graph of polygonal scenes.

Moreover, we have implemented our algorithm and we have made experimental comparisons with two other sweep algorithms for computing visibility graphs : the algorithm of [OW88] and the algorithm of [PV93a], which both run in time $O(m \log n)$. They were tested on scenes composed of line segments. We see that the gain due to the suppression of the logarithmic cost is significant.

2 Sweeping the visibility complex

We consider a scene of obstacles in the plane. Given a point in free space and a direction, one can extend the point backward and forward along the direction, until the extremities get into an obstacle. We get a maximal free line segment (or ray), and the pair of obstacles delimiting it is its visibility. The visibility complex, is a 2-dimensional cellular complex which componants represent maximal connected sets of rays having the same visibility, such that one can pass continuously from one ray to another while keeping the visibility constant.

[PV93b] (resp. [PV95]) give an incremental (resp. sweep) algorithm for computing the visibility complex of a scene of n curved convex objects in optimal time $O(n \log n + m)$ (where m is the size of the corresponding visibility graph) and in $O(m)$ (resp $O(n)$) space. They use in both their algorithms the concept of *pseudo-triangulation*.

Here we deal with general disjoints polygons, and the elementary objects are the line segments, sides of the polygons. We use duality to study the sweep of the visibility complex. A line $y = ax + b$ in the scene is turned into a point $(a, -b)$ in dual space, and a vertex extremity (v_x, v_y) in the scene is turned into a line $y = -v_x + v_y$ in dual space, this dual line representing all the lines passing through the vertex in the scene.

Since many rays can be contained in one line, the structure of the complex is more involved than the structure of the dual arrangement of lines of the scene (that is the arrangement of dual lines of polygon vertices). For example, the complex has a non planar structure, and dual lines corresponding to polygon vertices hidden from each other in the scene, do not have an intersectioni in the complex.

To handle the sweep, we define a partial order on the vertices of the complex. The we show that a sweep of the complex compatible with this order can be achieved by a succession of discrete events :"passing" one vertex of the complex. Then we extend the concept of *horizon trees*, devised by [EG86] to sweep an arrangement of n lines in optimal time $O(n^2)$, to the complex. These trees allow to find vertices that can be passed at each step of the sweep, according to the order. Finally, using new methods in the complex for finding visibilities between vertices in the scene, we show that our algorithm runs in optimal time $O(n \log n + m)$ and space $O(n)$.

3 Experimental results

The algorithm has been implemented. Extending the notion of *rotation trees*, data structure used in [OW88] to compute the visibility graph of a scene of line segments, we have programmed the sweep in the polygonal scene. The programm also handles degenerate configurations.

Then, we have made experimental comparisons for computing visibility graphs of scenes of line segments between two algorithms and ours. The first one is the algorithm of [OW88]. The second is the more general algorithm of [PV93a] which deals with general curved objects, specially

*iMAGIS-IMAG - BP 53
38041 GRENOBLE CEDEX 09 - FRANCE
e-mail : Stephane.Riviere@imag.fr
iMAGIS is a joint project between CNRS/INRIA/INPG/UJF

adapted for the tests to deal with line segments only. Both algorithms make in fact a strait sweep of the visibility complex, and have a time complexity $O(m \log n)$. The strait sweep needs a priority queue that we have implemented here with a heap (for implementation details about these two algorithms, see [Riv93]).

The elementary operation of these algorithms is the comparison of the slope of two line segments, or equivalently testing if a point is above or below a line segment. The complexity measured is the number of these elementary operations, the initialization of the structures not being taken into account.

The algorithms are tested on scenes taken at random. For each line segment $[(x, y), (x + dx, y + dy)]$, the coordinates x and y (resp. dx and dy) are chosen randomly in the interval $[-a, a]$ (resp. $[-b, b]$), using a uniform distribution.

We fix the number n of line segments in the scene, and by tuning a and b we generate scenes such that the sizes of the visibility graphs range from sparse graphs ($m = 4(n-1)$) to dense graphs ($m = 2n(n-1)$). For a given n , the complexity curves are lines of equation $complexity = c(n) * m + m_0$ (see figure 1).

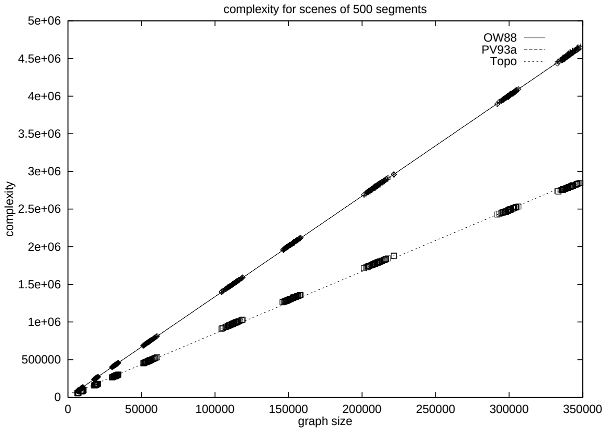

Figure 1: results for a fixed number of segment

By testing the algorithms on 500 scenes and using a linear regression, we compute their constant of complexity $c(n)$. Figure 2 shows the result : we plot the mesured $c(n)$ for $n = 100, 200, \ldots, 1000$.

The algorithms of [OW88] and of [PV93a] are very close, and their logarithmic nature is clearly shown (in fact the curves drawn for these algorithms are a "logarithmic regression"). Finally we see that our algorithm (named Topo in the figures) has better performances (its complexity is about the two thirds of the precedent above) for a same complexity of implementation.

4 Conclusion

We have devised an algorithm for topologically sweeping the visibility complex of a polygonal scene. It runs in optimal time and optimal working space. It is easy to implement : the data structures used are simple and the program is efficient in practice (the constants are small).

This algorithm allows the computation of the the visibility complex. Once built, the complex can be used to solve visibility requests. For example, one can compute a view from a point in time $O(m \log n)$ (n : size of scene, m size of the computed view) using the complex (see [PV93b]).

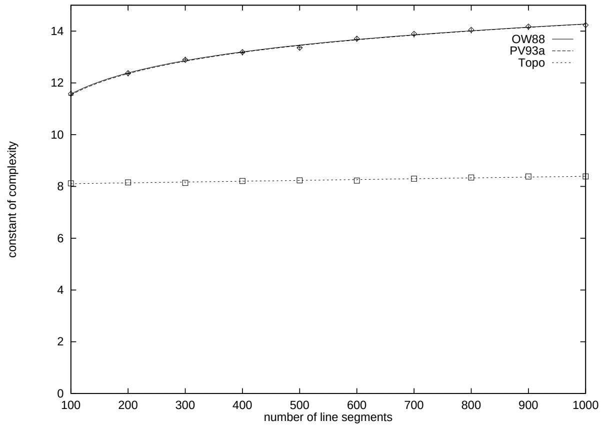

Figure 2: constant of complexity curves

Future research will study the visibility complex for dynamic scenes.

Author/Title Index

[EG86] H. Edelsbrunner and L. J. Guibas. Topologically sweeping an arrangement. In *Proc. 18th Annu. ACM Sympos. Theory Comput.*, pages 389–403, 1986.

[GM87] S. K. Ghosh and D. M. Mount. An output sensitive algorithm for computing visibility graphs. In *Proc. 28th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 11–19, 1987.

[OW88] M. H. Overmars and E. Welzl. New methods for computing visibility graphs. In *Proc. 4th Annu. ACM Sympos. Comput. Geom.*, pages 164–171, 1988.

[PV93a] M. Pocchiola and G. Vegter. Sweep algorithm for visibility graphs of curved obstacles. manuscrit, 1993.

[PV93b] M. Pocchiola and G. Vegter. The visibility complex. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 328–337, 1993.

[PV95] M. Pocchiola and G. Vegter. Computing the visibility graph via pseudo-triangulation. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, 1995.

[Riv93] S. Rivière. Experimental comparison of two algorithms for computing visibility graphs. manuscrit, 1993.