Approximation and randomized method for Visibility Counting Problem

Sharareh Alipour Mohammad Ghodsi

October 5, 2013

Abstract

For a set of n disjoint line segments S in R^2, the visibility counting problem (VCP) is to preprocess S such that the number of visible segments in S from a query point p can be computed quickly. This problem can be solved in logarithmic query time using $O(n^4)$ preprocessing time and space. In this paper, we propose a randomized approximation algorithm for this problem. The space of our algorithm is $O_\epsilon(n^{4-3\beta})$ and the query time is $O_\epsilon(n^\beta)$. The complexity of our algorithm is superior compare to best known algorithm for this problem.

Keywords. Computational geometry, visibility, approximation and randomized algorithm.

1 Introduction

Suppose that S is a set of n disjoint line segments in the plane. For simplicity, we assume that there is a bounding box containing all segments. Two points $p, q \in R^2$ are visible to each other with respect to (w.r.t.) S if the line segment $\overline{pq}$ does not intersect any segment of S. A segment $\overline{st} \in S$ is also said to be visible (w.r.t. S) from a point p if there exists a point $q \in \overline{st}$ such that p and q are visible to each other. The visibility counting problem (VCP) is to find the number of segments of S visible from any query point p.

Definition 1.1. *The visibility polygon of a given point $p \in R^2$ is defined as*

$$V_S(p) = \{q \in R^2 : p \textit{ and } q \textit{ are visible (w.r.t. } S)\}. \textit{ (Fig. 1.(a)).}$$

Definition 1.2. *The visibility polygon of a given segment $\overline{st}$ is defined as*

$$V_S(\overline{st}) = \bigcup_{q\in\overline{st}} V_S(q) = \{p \in R^2 : \overline{st} \textit{ and } p \textit{ are visible (w.r.t } S)\}. \textit{ (Fig. 1.(b))}.$$

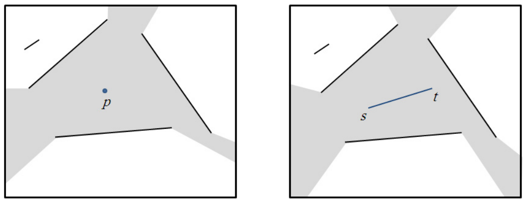

Figure 1: (a) The visibility polygon of a point p and (b) The visibility polygon of a line segment $\overline{st}$.

Consider the $2n$ endpoints of the segments of S as vertices of a graph. Add an edge between each pair of vertices, if they see each other. The resulting graph is *the visibility graph of* S or $VG(S)$ (Fig. 2.(a)). If we extend each edge of $VG(S)$ in both directions until it intersects the segments in S (or the bounding box), at most two new vertices and some edges are produced. Adding all these vertices and edges to $VG(S)$ results in a new graph called *the extended visibility graph* or $EVG(S)$ (Fig. 2.(b)).

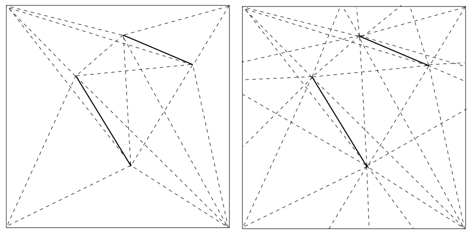

Figure 2: (a) A visibility graph. (b) The extended visibility graph of (a).

$V_S(p)$ can be computed in an optimal $O(n \log n)$ time using $O(n)$ storage [3, 12]. VCP can also be solved using $EVG(S)$. There is an optimal $O(n \log n + m)$ time algorithm to compute $VG(S)$, where $m = O(n^2)$ is the number of the edges of $VG(S)$ [7]. This algorithm can also be used to compute $EVG(S)$ in $O(n \log n + m)$ time [7]. Considering the planar arrangement of the edges of $EVG(S)$ as a planar graph which is called *the visibility space partition*, $VSP(S)$ of S, all points in any face of this arrangement have the same number of visible segments. Therefore, we can compute these numbers for each face in the preprocessing step and then, locate the face containing the query point. There are $O(n^4)$ faces in the planar arrangement of $EVG(S)$, so a point location structure of size $O(n^4)$ can answer each query in $O(\log n)$ time. Obviously, the preprocessing time and space of $O(n^4)$ is high. On the other hand, without any preprocessing, the query can be answered in $O(n \log n)$ time which is also too high. There are several other results with a tradeoff between the preprocessing cost and the query time [2, 4, 11, 14] (see the visibility book by Kumar Ghosh for a complete survey [8]). Therefore, the main concern of VCP can be to obtain approximation algorithms with a lower preprocessing cost in expense of higher than logarithmic query time. Moreover, the preference is to obtain an estimation close to the exact problem answer.

Suri and O'Rourke introduced the first 3-approximation algorithm for VCP based on representing a region by the union of a set of convex (triangular) regions [12]. Gudmundsson and Morin improved this result to a 2-approximation algorithm using an improved covering scheme [9]. Their method builds a data structure of size $O_\epsilon(n^{2(1+\alpha)})$ in the preprocessing time of $O_\epsilon(n^{2(1+\alpha)})$ and performs a query in $O_\epsilon(n^{(1-\alpha)})$ time, where $0 < \alpha \leq 1$ and $O_\epsilon(f(n)) = O(f(n)n^\epsilon)$ (Here, and throughout the remainder of the paper, $\epsilon, \delta > 0$ are constants that can be made arbitrarily small). If the exact answer to the query is m_p, the algorithm by [9] returns m'_p such that $m_p \leq m'_p \leq 2m_p$. There are two other approximation algorithms for VCP by Fischer *et.al.* [5, 6]. One of these algorithms uses a data structure of size $O((m/r)^2)$ to build a (r/m)-cutting for the $EVG(S)$ by which the queries are answered in $O(\log n)$ time with an absolute error of r compared to the exact answer $(1 \leq r \leq n)$. The second algorithm uses a random sampling method and builds a data structure of size $O((m^2 \log^{O(1)} n)/l)$ to answer any query in $O(l \log^{O(1)} n)$ time, where $1 \leq l \leq n$. In the latter method, the answer of VCP is approximated up to an absolute value of δn for any constant $\delta > 0$ (the constant δ affects the constant factor of both data structure size and the query time).

In this paper, we present a $(1+\delta)$-approximation algorithm with $O_\epsilon(n^{(1-\alpha)})$ query time, which needs a data structure of size and the preprocessing time of $O_\epsilon(n^{(2+1.5\alpha)})$, where $0 < \alpha \leq 1$.

2 The new method

In this section after defining some new concepts, we present our randomized algorithm to approximate the answer of VCP.

2.1 Definitions and the new algorithm

For each end-point of S, consider a vertex and for each pair of the end-points of S, which are visible to each other, add an edge between them with the probability of $1/n^\alpha$. We call this graph *Probabilistic Visibility Graph* and denote it by $VG'(S)$. Obviously $VG'(S)$ is a subgraph of $VG(S)$. Similar to $EVG(S)$, we extend the edges of $VG'(S)$ in both directions to intersect a segment of S or the bounding box. This new graph is called *Probabilistic Extended Visibility Graph* and is denoted by $EVG'(S)$. Also, we consider the plane graph of $EVG'(S)$ that partitions the plane into a set of regions, called *Probabilistic Visibility Space Partition*, i.e., $VSP'(S) = \{r'_1, ..., r'_{m'}\}$ (Fig. 3).

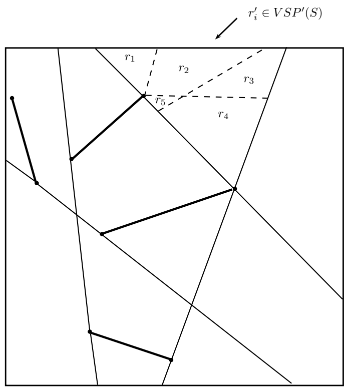

Figure 3: $VSP'(S)$ and $r'_i \in VSP'(S)$. For $r_i \in VSP'(S)$, we have $a_i = 2$ and $b_i = 3$.

Lemma 2.1. *The expected size of* $VSP'(S)$*,* $E(|VSP'(S)|)$*, is* $O(n^{4-2\alpha})$*.*

Proof. Let $E_{EVG'} = \{e_1, e_2, ..., e''_m\}$ be the set of edges in $EVG'(S)$. We consider a random variable X_i for each edge of $EVG(S)$, such that $X_i = 1$ if e_i is chosen and otherwise $X_i = 0$, then

$$E(m'') = \textstyle\sum_{i=1}^{m''} X_i = n^2.1/n^\alpha = n^{2-\alpha}.$$

As each pair of edges in $EVG'(S)$ may intersect, then the expected number of edges, vertices and $E(|VSP'(S)|)$ are $O(n^{4-2\alpha})$.

Consider a region $r' \in VSP'(S)$. r_i is a combination of some regions in $VSP(S)$. So, we could achieve $VSP'(S)$ by merging some adjacent regions of $VSP(S)$. Two regions are adjacent, if they have a mutual edge and merging two adjacent regions means to combine them by omitting their mutual edge. Let r be a region of $VSP(S)$. According to the definitions, some of its edges or vertices are omitted in $VSP'(S)$. Let r' be a region of $VSP'(S)$ containing all point of r, then we say r is a sub-region of r'.

Consider a region $r'_i \in VSP'(S)$. r'_i is a combination of some regions of $VSP(S)$ that are merged in $VSP'(s)$. As it was noted, we can compute the exact answer of VCP for each of these sub-regions. Suppose that the minimum value of VCP among these regions is a_i in $a \in VSP(S)$ and the maximum value is b_i in $b \in VSP(S)$ (Fig. 3). We know that a and b are sub-regions of r'_i. For each $r'_i \in VSP'(S)$, we save a_i and b_i. As the difference of two adjacent regions is at most 1, then there should be a set of sub-regions $SR = \{a = r_1, ..., b = r_k\}$ such that r_i and r_{i+1} are adjacent and $|SR| \geq b_i - a_i$. This means that $O(b_i - a_i)$ edges are not chosen in $EVG'(S)$. As each edge is chosen with the probability of $1/n^\alpha$, the probability that there exists r'_i is at most

$$P(r'_i) \leq (1 - 1/n^\alpha)^{b_i - a_i}.$$

and for each $r'_i \in VSP'(S)$

$$P(b_i - a_i \geq \delta n^\beta) \leq (1 - 1/\sqrt{n})^{\delta n^\beta}.$$

Suppose that we construct $VSP'(S)$, $z = O(n^\gamma)$ times, denoted by $VSP'_i(S)$, $i = 1.., z$. Now consider a query point p. In each $VSP'_i(S)$, let $r'_i \in VSP'_i(S)$ be the region containing p. If $b_i/a_i \leq 1 + \delta$ then, we report b_i as the approximated value of m_p. In this condition we have

$$m_p \leq b_i \leq (1 + \delta) m_p$$

Which is a $1 + \delta$ approximation factor answer to VCP. We prove that, if $m_p \geq n^\beta$ and $\beta + \gamma > \alpha$, then there is a region $r_j \in VSP'_j(S)$ such that with high probability $b_j/a_j \leq 1 + \delta$. Assume that for each $r'_i \in VSP'_i(S)$, $i = 1.., z$, $b_i/a_i \geq 1 + \delta$. This means that in each r_i, at least $\delta O(\delta n^\beta)$ edges are not chosen and the probability that this happens is at least $O(1 - (1/n^\alpha))^{\delta n^\beta})$. So, the probability that in all the $VSP'_i(S)$ this happens is at least

$$P = O((1 - (1/n^\alpha))^{\delta n^{\beta+\gamma}}).$$

Since δ is a constant, it is obvious that, if $\beta + \gamma > \alpha$, then $P \to 0$. If there is not any $r_i \in VSP'_j(S)$ such that $b_j/a_j \leq 1 + \delta$, then with the probability of $1 - P$, $m_p \leq n^\beta$, so we use the method of [13] to calculate the exact value of m_p (Algorithm 1).

Algorithm 1 $(1+\delta)$-approximation factor algorithm for VCP

```
1: Input: $S$ (a set of segments) and $p$ (a query point).
2: Output: The number of visible segments from $p$.
3: For each query:
4: for each $VSP'_i(S)$ do
5:     if $b_i/a_i \leq 1+\delta$ then
6:         return $b_i$ as the approximated value of $m_p$
7:         Break
8:     end if
9: end for
10: Use the method of [13] to compute $V_S(p)$
11: return the exact value of $m_p$.
```

2.2 Analysis of time and space

According to Lemma 2.1, the space for each $VSP'_i(S)$ is $O(n^{4-2\alpha})$. On the other hand, the space for constructing the data structure of [13] is $O(n^2)$. So, the total space is $O(n^{4-2\alpha+\gamma}+n^2)$. We choose α and γ such that $4-2\alpha+\gamma \geq 2$, then the space is $O(n^{4-2\alpha+\gamma})$.

At the query time, with logarithmic time for each $VSP'_i(S)$, we can find the location of query point p and as we do this at most $O(n^\gamma)$ times, the query time is $O_\epsilon(n^\gamma)$. If there is no r_i with $b_i/a_i \leq 1+\delta$, then we have to use the method of [13] with the query time of $O_\epsilon(n^\beta)$ to compute $V_S(p)$ and calculate the exact answer. So, in the worst case the query time is $O_\epsilon(n^\beta + n^\gamma)$.

Theorem 2.1. *We can answer VPC in $O_\epsilon(n^\beta)$ by using $O_\epsilon(n^{4-3\beta})$ space $(0 \leq \beta \leq 2/3)$. Our algorithm returns a value m'_p such that $m_p \leq m'_p \leq (1+\delta)m_p$ when $m_p \geq n^\beta$ and the exact value when $m_p \leq \Theta(n^\beta)$.*

Proof. If we choose $\alpha = 2\beta - \epsilon = 2\gamma - \epsilon$, then we achieve the desired space and query time.

Compared to the best known result for VCP, according to [9], if we set $k = n^{4-3\beta}$, then the space would be $O_\epsilon(k)$ and the query time would be $O_\epsilon(n^{1.5\beta})$, where in our method with this space the query time would be $O_\epsilon(n^\beta)$. Their method gives a 2-approximation factor algorithm, where in our method we can achieve a $1+\delta$-approximation factor algorithm where δ can be chosen as little as we want.

3 Conclusion

In this paper we proposed a randomized algorithm to approximate the answer of VCP. Compared with the best known results for this problem, our algorithm improves the approximation factor and the query time. But we have not yet find any method to reduce the construction time for calculating a_i and b_i for each region. So, working on these issues is of our interest.

References

[1] Alipour, S., Zarei, A. Visibility Testing and Counting. *FAW-AAIM 2011, Jinhua, China, LNCS*, (Volume 6681) by Springer-Verlag: 343-351, 2011.

[2] Aronov, B., Guibas, L. J., Teichmann M. and Zhang L. Visibility queries and maintenance in simple polygons. *Discrete and Computational Geometry*, 27:461–483, 2002.

[3] Asano, T. An efficient algorithm for finding the visibility polygon for a polygonal region with holes. *IEICE Transactions*, 557–589, 1985.

[4] Bose, P., Lubiw, A. and Munro, J. I. Eficient visibility queries in simple polygons. *Computational Geometry Theory and Applications*, 23(7):313–335, 2002.

[5] Fischer, M., Hilbig, M., Jahn, C., Meyer auf der Heide F. and Ziegler M. Planar visibility counting. *CoRR, abs/0810.0052*, 2008.

[6] Fischer, M., Hilbig, M., Jahn, C., Meyer auf der Heide F. and Ziegler M. Planar visibility counting. *In Proceedings of the 25th European Workshop on Computational Geometry(EuroCG 2009)* , 203–206, 2009.

[7] Ghosh, S. K. and Mount, D. An output sensitive algorithm for computing visibility graphs. *SIAM Journal on Computing*, 20:888–910, 1991.

[8] Ghosh, S. K. Visibility algorithms in the plane. *Cambridge university press*, 2007.

[9] Gudmundsson, J., Morin, P. Planar visibility: testing and counting. *Annual Symposium on Computational Geometry*, 77–86, 2010.

[10] Nouri, M. and Ghodsi, M. Space/query-time tradeoff for computing the visibility polygon. *Computational Geometry*, 46(3), 371–381, 2013.

[11] Pocchiola, M. and Vegter, G. The visibility complex. *International Journal of Computational Geometry and Applications*, 6(3):279–308, 1996.

[12] Suri, S. and O'Rourke, J. Worst-case optimal algorithms for constructing visibility polygons with holes. *In Proceedings of the Second Annual Symposium on Computational Geometry (SCG 84)*, 14–23, 1984.

[13] Vegter, G. The visibility diagram: A data structure for visibility problems and motion planning. *In: Gilbert, J.R., Karlsson, R. (eds.) SWAT(1990). LNCS,* 447:97–110. Springer, Heidelberg, 1990.

[14] Zarei, A. and Ghodsi, M. Efficient computation of query point visibility in polygons with holes. *In Proceedings of the 21st Annual ACM Symposium on Computational Geometry. (SCG 2005)*, 2005.