Visibility Testing and Counting for Uncertain Segments

Mohammad Ali Abam * Sharareh Alipour† Mohammad Ghodsi ‡ Mohammad Mahdian §

Abstract

We study two well-known planar visibility problems, namely visibility testing and visibility counting, in a model where there is uncertainty about the input data. The standard versions of these problems are defined as follows: we are given a set $\mathcal{S}$ of n segments in $\mathbb{R}^2$, and we would like to preprocess $\mathcal{S}$ so that we can quickly answer queries of the form: is the given query segment $s \in \mathcal{S}$ visible from the given query point $q \in \mathbb{R}^2$ (for visibility testing) and how many segments in $\mathcal{S}$ are visible from the given query point $q \in \mathbb{R}^2$ (for visibility counting).

In our model of uncertainty, each segment may or may not exist, and if it does, it is located in one of finitely many possible locations, given by a discrete probability distribution. In this setting, the probabilistic visibility testing problem (PVTP, for short) is to compute the probability that a given segment $s \in \mathcal{S}$ is visible from a given query point q and the probabilistic visibility counting problem (PVCP, for short) is to compute the expected number of segments in $\mathcal{S}$ that are visible from a query point q. We first show that PVTP is $\#P$-complete. In the special case where uncertainty is only about whether segments exist and not about their location, we show that the problem is solvable in $O(n \log n)$ time. Using this, together with a few old tricks, we can show that one can preprocess $\mathcal{S}$ in $O(n^5 \log n)$ time into a data structure of size $O(n^4)$, so that PVTP queries can be answered in $O(\log n)$ time. Our algorithm for PVTP combined with linearity of expectation gives an $O(n^2 \log n)$ time algorithm for PVCP. We also give a faster 2-approximation algorithm for this problem.

1 Introduction

Background. Visibility testing and visibility counting are basic problems in computational geometry. Visibility plays an important role in robotics and computer graphics. In robotics, for example, the efficient exploration of an unknown environment requires computing the visibility polygon of the robot or the number of visible objects from the robot or test whether the robot sees a specific object. In some computer graphics applications, also, it is important to identify the objects in a scene that are illuminated by a light source.

Two points $p, q \in \mathbb{R}^2$ are visible from each other with respect to $\mathcal{S}$, if there exists no segment $s \in \mathcal{S}$ intersecting line segment $\overline{pq}$. We say that a segment $\overline{st} \in \mathcal{S}$ is visible from a point p, if a point $q \in \overline{st}$ can be found from which p is visible. In this paper, we consider two planar visibility problems; visibility testing and visibility counting. For a set $\mathcal{S}$ of n segments in $\mathbb{R}^2$ and a point q, in visibility testing problem, we want to test whether q sees a given segment $s \in \mathcal{S}$. In visibility counting problem we want to count the number of segments in $\mathcal{S}$ that are visible from q. For simplicity we assume all the segments are contained in a bounding box.

Uncertain data. It is not surprising that in many real-world applications we face uncertainty about the data. For geometric problems like visibility, this means uncertainty about the location of the input set. There are multiple ways to model such uncertainty. For example, we can assume each object lies inside some region, but not exactly where in that region, and use this assumption to prove bounds on the quantity of interest. Such a model is used in [14]. Alternatively, we can use a discrete probability distribution to model uncertainty. This "stochastic" approach is used in [1, 11]. We choose the latter approach in this paper. In particular, our model of uncertainty is very similar to the model used in [11].

Related work. There is significant prior work on the non-stochastic version of the problems studied in this paper. There are some works dedicated not only to the exact computing [5, 12, 15] of the problem but also to approximate computing [3, 4, 9, 12]. In both, time-space trade-offs haven been considered.

In real application there are situations where we need to model the problems based on uncertain data (See [1, 14, 10]). In [6], they compute visibility between imprecise points among obstacles. This leads us to define the uncertain model of VTP and VCP and propose algorithms to solve them.

*Computer Engineering Department, Sharif University of Technology

†Institute for Research in Fundamental Sciences (IPM) School of Computer Science

‡Computer Engineering Department, Sharif University of Technology and Institute for Research in Fundamental Sciences (IPM) School of Computer Science

§Google research

Problem statement. Suppose we are given a set $\mathcal{S}$ of n uncertain segments. More precisely, we are given a discrete probability distribution for each $s_i \in \mathcal{S}$, that is, we have a set $\mathcal{D}_i = \{s_{i,1}, \cdots, s_{i,m_i}\} \cup \{s_{i,0} = \perp\}$ of possible locations with associated probabilities $p_{i,j}$ such that $\mathbf{Pr}(s_i = s_{i,j}) = p_{i,j}$ and $\sum_j p_{i,j} = 1$. The special segment $\perp$ indicates that the segment s_i does not exist in $\mathcal{S}$. In this setting, the set $\mathcal{S}$ can be seen as a random variable (or random set) as it consists of probabilistic segments. This random variable gets its value from a sample space of size $\Pi_i(m_i + 1)$ with the probability being equal to $\Pi_{s\in\mathcal{S}}\mathbf{Pr}(s)\Pi_{s\notin\mathcal{S}}(1-\mathbf{Pr}(s))$. To this end, assume $z = \max\{1 + m_i\}$, i.e., z denotes the maximum size of the given distributions. A special case that we will pay special attention to is when $z = 2$. This is the case where the uncertainty is only about the existence of the segments, and not about their location.

It is natural to define the probabilistic version of visibility testing and visibility counting problems in the above setting where $\mathcal{S}$ is a random set:

- Probabilistic Visibility Testing Problem (PVTP): compute the probability that a given segment $s \in \mathcal{S}$ is visible from a given query point q.
- Probabilistic Visibility Counting Problem (PVCP): compute the expected number of segments in $\mathcal{S}$ being visible from q.

Our results. We first show that PVTP is $\#P$-complete. We then turn our attention to the special case where $z = 2$. We present an algorithm running $O(n \log n)$ time that answers PVTP. Then, we present a simple way of putting n uncertain segments into a data structure of size $O(n^4)$ such that queries can be answered in $O(\log n)$ time. Finally, we focus our attention to PVCP whose complexity class is unknown to us. Here, we present a polynomial-time 2-approximation algorithm that approximately solves PVCP. We then show how to preprocess $\mathcal{S}$ into a data structure of size $O(n^4)$ in order to approximately answer each query in $O(\log n)$ time.

2 Probabilistic visibility testing

We start by a simple polynomial-time reduction from #perfect-matching problem to PVTP in order to show PVTP is $\#P$-complete. The #perfect-matching problem of computing the number of perfect matching in a given bipartite graph, is known to be $\#P$-complete [13] even for 3-regular bipartite graphs [8]. We next explain the details.

Suppose a bipartite graph $G = (U, V, E)$ is input to #perfect-matching problem where $U = \{u_1, \cdots, u_n\}$ and $V = \{v_1, \cdots, v_n\}$ are vertex parts of G and E is the edge set of G. For the given bipartite graph, we construct an instance of PVTP and introduce a query point q and a query segment s such that each perfect matching uniquely corresponds to one element of the sample space of uncertain segments in which s is not visible from q. Consider n intervals $[i, i + 1]$ on the x-axis where i changes from 0 to $n - 1$. Imagine the interval $[i, i + 1]$ corresponds to the vertex v_i; denoted by $I(v_i)$. For each vertex $u_i \in U$, we define an uncertain segment $\mathcal{D}_i = \{I(v_j) | \{u_i, v_j\} \in E\}$ with the uniform distribution—note that in this instance each uncertain segment always exist. We add one more uncertain segment s consisting of one segment with probability 1 whose endpoints are $(0, -1)$ and $(n, -1)$. To this end, consider the query point q is $(n/2, n)$ (See figure 1).

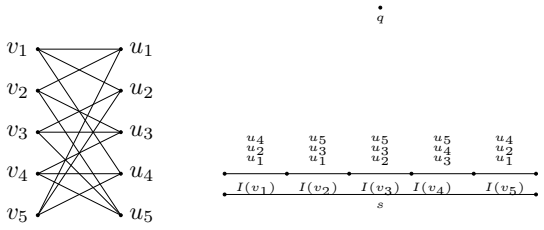

Figure 1: Each matching in the left side corresponds to a set of segments that cover s in the right side and each set of segments that cover s corresponds to a matching.

Segment s is not visible from q iff the interval $[0, n]$ is completely covered by the uncertain segments defined on the x-axis. There are n such uncertain segments and each covers exactly 1 unit of $[0, n]$. Therefore, each uncertain segment must cover exactly one of n unit intervals. So, the number of perfect matchings is equal to the number of ways that s is covered by the uncertain segments. This is the intuition behind one-to-one correspondence between perfect matching and the subset of the sample space in which s is not visible from q. Therefore, we conclude the following theorem.

Theorem 1 *PVTP is $\#P$-complete.*

From now on, we restrict ourself to the special case where $z = 2$, i.e., each uncertain segment either does not exist or exists in only one possible location. Suppose we are given n uncertain segments $s_1, \cdots, s_n$. Let $\mathbf{Pr}(s_i \in \mathcal{S}) = p_i$ which of course implies $\mathbf{Pr}(s_i \notin \mathcal{S}) = 1 - p_i$.

Next, we explain how to compute $\mathbf{Pr}(q \text{ sees } s)$ for the given segment s and point q. If $s \notin \mathcal{S}$, q of course can not see s. Therefore, $\mathbf{Pr}(q \text{ sees } s) = \mathbf{Pr}(q \text{ sees } s | s \in \mathcal{S})\mathbf{Pr}(s \in \mathcal{S})$. This reduces our task to computing of $\mathbf{Pr}(q \text{ sees } s | s \in \mathcal{S})$. Let Δ be a triangle with vertex q and side s. Every other uncertain segment that does not intersect Δ, can not prevent q to see s. Therefore, we can restrict ourself to uncertain segments intersecting Δ. We project these uncertain segments to s with respect to q. Now, as the main ingredient, we must solve

the following problem (See figure 2):

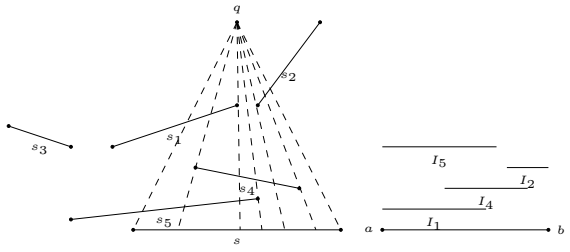

Figure 2: The projection of uncertain segments on s according to q defines four uncertain intervals.

- Suppose we are given n uncertain intervals $I = \{I_1, \cdots, I_n\}$ on the real line; each I_i exists with probability p_i. Compute the probability that the given interval $[a, b]$ is covered by the uncertain intervals, denoted by $\mathbf{Pr}([a, b]$ is covered).

Computing the desired probability seems needs $\Theta(2^n)$ time as the size of the sample space can be $\Theta(2^n)$ in the worst case. But, we next show how the dynamic paradigm helps us to perform the computation in $O(n \log n)$ time. For simplicity, we can assume the intervals have been sorted by their right endpoints and intersection of each I_i with $[a, b]$ is not empty. Let $r(I_i)$ ($l(I_i)$) be the right (left) endpoint of I_i. We present the following recursive formula.

For each point $a' \in [a, b]$, let $sol(a')$ be the probability that $[a', b]$ is covered. So, $sol(a)$ is the probability that $[a, b]$ is covered. Let $S(a') = \{I'_1, ..., I'_l\}$ be the set of intervals that cover a' and they are sorted according to their right endpoints (See figure 3).

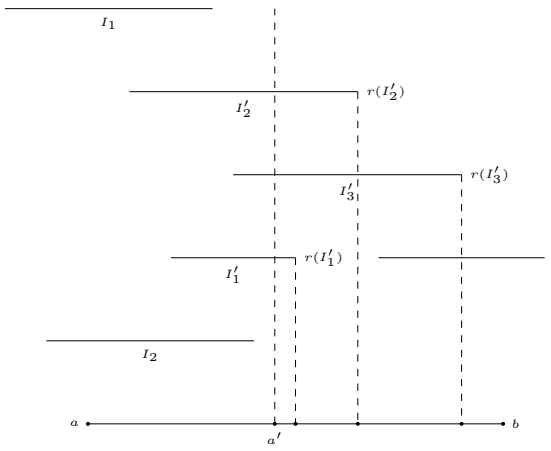

Figure 3: I'_1, I'_2 and I'_3 are the intervals that can cover $[a, b]$, so we have $sol(a') = p'_1 sol(r(I'_1)) + p'_2(1 - p'_1)sol(r(I'_2)) + p'_3(1 - p'_2)(1 - p'_1)sol(r(I'_3))$.

Lemma 2 *We define $sol(b) = 1$, then we have*

$$sol(a') = \sum_{j=1}^{l} p'_j(\prod_{i=1}^{j-1}(1 - p'_i))sol(r(I'_j)).$$

Proof. Suppose that $a' \in [a, b]$, so if $[a', b]$ is covered, then at least one of the segments in $S(a')$ should be chosen. There are l segments that cover a'. Since the segments in $S(a')$ are sorted according to their right endpoints then, the probability that I'_j is the first segment that covers a' is $p'_j \prod_{i'=1}^{j-1}(1-p'_i)$. Recursively $[a', b]$ is covered with the probability of $sol(r(I'_j))$. So, we have

$$sol(a') = \sum_{j=1}^{l} p'_j(\prod_{i'=1}^{j-1}(1 - p'_i))sol(r(I'_j)).$$

□

Each right endpoint of the intervals can be covered by $O(n)$ of the intervals. In the recursive formula, we call each right endpoint at most once. For each $sol(r(I'_j))$ we have to compute $\prod_{i'=1}^{j-1}(1 - p'_i)$, since the segments are sorted according to their right endpoint, for each $sol(r(I'_j))$ we multiply $\prod_{i'=1}^{j-2}(1 - p'_i)$(the value of previous step) by $1-p'_j$, which means we can compute $sol(a)$ in $O(n^2)$ time. Next we propose a faster algorithm.

To fill the array sol, we sweep the endpoints from right to left and keep the track of all intervals intersecting the sweep line in a binary search tree (BST, for short) over the right endpoint of intervals supporting insertion/deletion in $O(\log n)$ time. We augment each node of the BST with extra values in order to expedite our computation as we explain next.

Upon processing a right endpoint, say $r(I_i)$, we compute $sol(r(I_i))$, which is the sum of all the nodes of tree. This can be computed in $O(\log n)$ time. Then, we implicitly multiply all the nodes by $(1 - p_i)$ and then add $r(I_i)$ to the tree with the value of $p_i sol(r(I_i))$. For the left endpoint of an interval, $l(I_i)$, we delete I_i, from the tree and implicitly divide all the right endpoints greater than $r(I_i)$ by $(1-p_i)$. This also can be done in $O(\log n)$ time. There are $O(n)$ endpoints, so the running time is $O(n \log n)$.

Theorem 3 *Given a point and a segment, PVTP can be answered in $O(n \log n)$ time when $z = 2$.*

Now, we preprocess the segments such that for any given query point q, PVTP can be answered in $O(\log n)$ time. First, connect each pair of the endpoints by a line and extend it until it hits the bounding box. These lines will partition the bounding box into $O(n^4)$ regions. For a fixed segment $s \in \mathcal{S}$, the answer to PVTP for all the points in a given region is the same, because the combinatorial order of segments that cover s is the same for all the points inside that region. Therefore, in the preprocessing time we choose a point q_i from each region

r_i and compute $\mathbf{Pr}(q_i \text{ sees } s)$ in $O(n \log n)$ time. So, for a given set of segments $\mathcal{S}$ and a segment $s \in \mathcal{S}$, we preprocess the segments in $O(n^5 \log n)$ time and $O(n^4)$ space such that for any given query point q, we locate the region r_i containing q in $O(\log n)$ time and return $\mathbf{Pr}(q_i \text{ sees } s) = \mathbf{Pr}(q \text{ sees } s)$.

3 Probabilistic visibility counting

In this section we study the probabilistic visibility counting problem. We start with a few notations. For each subset $T \subset \mathcal{S}$, let $m_q(T)$ be the number of segments visible from q when the set of segments is T. So, the expected number of segments visible from q can be written as: $E(m_q) = \sum_{T \subseteq \mathcal{S}} \mathbf{Pr}(T) m_q(T)$, where $\mathbf{Pr}(T)$ denotes the probability that the set of realized segments is T.

Another way to compute $E(m_q)$ is using linearity of expectations: $E(m_q) = \sum_{i=1}^{n} \mathbf{Pr}(q \text{ sees } s_i)$.

For the case $z = 2$, we can use the above identity and the algorithm in the previous section to compute $E(m_q)$ in $O(n^2 \log n)$ time with no preprocessing. Also as in the previous section, we can use preprocessing to reduce query time: the answer of $PVCP$ is the same for all the points in each region in the space partition. So, we can compute this number for all the regions in $O(n^6 \log n)$ preprocessing time and $O(n^4)$ space, such that for any query point q, $E(m_q)$ can be answered in $O(\log n)$ time. Now, we show how to approximately solve this problem more efficiently.

3.1 Approximation of PVCP

In this section we propose a 2-approximation solution for PVCP. First, we present the following theorem

Theorem 4 [4] *Let S be a set of disjoint line segments in the plane and ve_q be the number of visible endpoints of the segments and m_q be the number of visible segments, then we have*

$$m_q \leq ve_q \leq 2m_q$$

Now, we use Theorem 4 to approximate PVCP. Let $m_q(T)$ and $ve_q(T)$ be the number of visible segments and visible endpoints, respectively in $T \subset \mathcal{S}$ w.r.t T, so we have $m_q(T) \leq ve_q(T) \leq 2m_q(T)$. So, we can conclude that,

$$\begin{aligned}\sum_{T \subset \mathcal{S}} \mathbf{Pr}(\mathcal{S} = T) m_q(T) &\leq \sum_{T \subset \mathcal{S}} \mathbf{Pr}(\mathcal{S} = T) ve_q(T) \\ &\leq \sum_{T \subset \mathcal{S}} \mathbf{Pr}(\mathcal{S} = T) 2m_q(T).\end{aligned}$$

Or in other words,

$$E(m_q) \leq E(ve_q) \leq 2E(m_q).$$

So, we compute

$$E(ve_q) = \sum_{i=1}^{n} \mathbf{Pr}(r(s_i) \text{ sees } q) + \mathbf{Pr}(l(s_i) \text{ sees } q).$$

We have

$$\mathbf{Pr}(r(s_i) \text{ sees } q)) = \textstyle\sum_{j=1}^{z} p_{i,j} \mathbf{Pr}(r(s_{i,j}) \text{ sees } q).$$

Let $s_{k,1'}, s_{k,2'}, ..., s_{k,l'}$ be the possible locations of s_k in $\mathcal{D}_k$ that cross $\overline{r(s_{i,j})q}$, the probability that s_k does not intersect $\overline{r(s_{i,j})q}$ is $p_k^{i,j} = (1 - p_{k,1'} - p_{k,2'} - ... - p_{k,l'})$.

$$\mathbf{Pr}(q \text{ sees } r(s_i)) = \textstyle\sum_{j=1}^{z} p_{i,j} p_1^{i,j} p_2^{i,j} .... p_n^{i,j}$$

We have $2nz$ possible locations for the endpoints and we can compute $P(q \text{ sees } r(s_i))$ in $O(zn)$, so $E(ve_q)$ is computed in $O(n^2 z^2)$.

For $z = 2$ we present a faster algorithm. Suppose that $a \in s_i$ is an endpoint of s_i. Let $s'_1, s'_2, ..., s'_k$ be the set of segments that intersect $\overline{aq}$, since the probability of selection of the segments are independent, we have

$$\mathbf{Pr}(q \text{ sees } a) = p_i(1 - p'_1)(1 - p'_2)...(1 - p'_k).$$

Which yields: $E(ve_p) = \sum_{a \in s_i} \mathbf{Pr}(q \text{ sees } a)$.

So, for each endpoint, we need the segments that intersect $\overline{aq}$. We use the following theorem:

Theorem 5 [2, 7] *Let S be a set of n segments in the plane and $n \leq k \leq n^2$, we can preprocess the segments in $O_\epsilon(k)$ such that for a given query segment s, the number of segments crossed by s can be computed in $O_\epsilon(n/\sqrt{k})$. Where $O_\epsilon(f(n)) = O(f(n)n^\epsilon)$ and $\epsilon > 0$ is a constant that can be made arbitrarily small.*

By Theorem 5 we can compute $\mathbf{Pr}(q \text{ sees } a)$ in $O(n/\sqrt{k})$. So, for $2n$ endpoints, $E(ve_p)$ is computed in $n \cdot O(n/\sqrt{k})$. If $k = n^{\frac{4}{3}}$, then we have:

Theorem 6 *Let, S be a set of given segments and q be a given point. If each segment is chosen with probability p_i, then, the expected number of visible endpoints from q can be computed in $O_\epsilon(n^{\frac{4}{3}})$ which is a 2-approximation of $E(m_q)$.*

4 Conclusion

We introduced a probabilistic variant of two well known visibility problems: visibility testing and counting. We proved that visibility testing problem in general case is $\#P$-complete. Then, we proposed a polynomial time for a special case of these problems and then gave an approximation algorithm for the probabilistic visibility counting problem. In future we want to study the complexity of these problems in some other special cases. Also, we want to study algorithms to approximate the answer of probabilistic visibility testing problem.

Acknowledgments

We thank Mahdi Safarnejad for his comments and helps.

References

[1] M. A. Abam, M. de Berg, and A. Khosravi. Piecewise-linear approximations of uncertain functions. In *Algorithms and Data Structures - 12th International Symposium, WADS 2011, New York, NY, USA, August 15-17, 2011. Proceedings*, pages 1–12, 2011.

[2] P. K. Agarwal and M. Sharir. Applications of a new space-partitioning technique. *Discrete & Computational Geometry*, 9:11–38, 1993.

[3] S. Alipour, M. Ghodsi, A. Zarei, and M. Pourreza. Visibility testing and counting. *Inf. Process. Lett.*, 115(9):649–654, 2015.

[4] S. Alipour and A. Zarei. Visibility testing and counting. In *Proceedings of the 5th Joint International Frontiers in Algorithmics, and 7th International Conference on Algorithmic Aspects in Information and Management*, FAW-AAIM'11, pages 343–351, Berlin, Heidelberg, 2011. Springer-Verlag.

[5] T. Asano. An efficient algorithm for finding the visibility polygon for a polygonal region with holes. *IEICE TRANSACTIONS (1976-1990)*, 68(9):557–589, 1985.

[6] K. Buchin, I. Kostitsyna, M. Löffler, and R. I. Silveira. Region-based approximation of probability distributions (for visibility between imprecise points among obstacles). *CoRR*, abs/1402.5681, 2014.

[7] S. Cheng and R. Janardan. Algorithms for ray-shooting and intersection searching. *J. Algorithms*, 13(4):670–692, 1992.

[8] P. Dagum, M. Luby, M. Mihail, and U. V. Vazirani. Polytopes, permanents and graphs with large factors. In *29th Annual Symposium on Foundations of Computer Science, White Plains, New York, USA, 24-26 October 1988*, pages 412–421, 1988.

[9] J. Gudmundsson and P. Morin. Planar visibility: testing and counting. In *Proceedings of the 26th ACM Symposium on Computational Geometry, Snowbird, Utah, USA, June 13-16, 2010*, pages 77–86, 2010.

[10] M. Löffler and M. J. van Kreveld. Largest bounding box, smallest diameter, and related problems on imprecise points. *Comput. Geom.*, 43(4):419–433, 2010.

[11] A. Munteanu, C. Sohler, and D. Feldman. Smallest enclosing ball for probabilistic data. In *30th Annual Symposium on Computational Geometry, SOCG'14, Kyoto, Japan, June 08 - 11, 2014*, page 214, 2014.

[12] S. Suri and J. O'Rourke. Worst-case optimal algorithms for constructing visibility polygons with holes. In *Proceedings of the Second Annual Symposium on Computational Geometry*, SCG '86, pages 14–23, New York, NY, USA, 1986. ACM.

[13] L. G. Valiant. The complexity of computing the permanent. *Theor. Comput. Sci.*, 8:189–201, 1979.

[14] M. J. van Kreveld and M. Löffler. Approximating largest convex hulls for imprecise points. *J. Discrete Algorithms*, 6(4):583–594, 2008.

[15] G. Vegter. The visibility diagram: a data structure for visibility problems and motion planning. In J. R. Gilbert and R. G. Karlsson, editors, *SWAT*, volume 447 of *Lecture Notes in Computer Science*, pages 97–110. Springer, 1990.