Journal of Computational Science 3 (2012) 127–131

Contents lists available at SciVerse ScienceDirect

Journal of Computational Science

journal homepage: www.elsevier.com/locate/jocs

Near-pole polar diagram of objects and duality

Bahram Sadeghi Bigham[a,*], Marzieh Eskandari[b], Maryam Tahmasbi[c]

[a] Department of Computer Science and Information Technology, Institute for Advanced Studies in Basic Sciences, Zanjan, Iran
[b] Department of Mathematics, Alzahra University, Tehran, Iran
[c] Department of Computer Science, Mathematical Science Faculty, Beheshti University, Tehran, Iran

ARTICLE INFO

Article history:
Received 24 June 2010
Received in revised form 12 August 2011
Accepted 29 August 2011
Available online 7 September 2011

Keywords:
Polar diagram
Voronoi diagram
Duality
Computational geometry
Radar

ABSTRACT

The polar diagram of a set of points in a plane and its extracted dual *EDPD* were recently introduced for static and dynamic cases. In this paper, the near-pole polar diagram *NPPD* for a set of points is presented. This new diagram can be considered as a generalization of the polar diagram and has applications in several communication systems and robotics problems. After reviewing the *NPPD* of points, we solve the problem for a set of line segments and simple polygons in optimal time $\Theta(n \log n)$, where n is the number of line segments or polygon vertices. We introduce duality for the *NPPD* of points and identify some applications.

1. Introduction

Solutions to visibility and many other important problems in computational geometry require some type of angle processing of the data input. In this paper we introduce a relation between some visibility problems and a new plane partitioning process we call a near-pole polar diagram (*NPPD*). The problems in question occur in communication and radio systems, antenna arrangement and target detection. The approach proposed can also be applied to illumination problems, robot vision, hidden surface removal, collision detection and decorative patterning. *NPPD* is a generalization of the Voronoi diagram and is described in Section 3.

One of the most fundamental concepts in computational geometry is the Voronoi diagram, and its algorithms and applications have been studied extensively [6,8]. This concept has also been generalized in a variety of fields by replacing the standard Euclidean distance with other metrics such as the L_p distance [11], weighted distances [4,7], the power distance [5], and a skew distance [2]. Using the Euclidean distance, the β-skeleton and proximity graphs have been defined and are used to solve other problems [1,3]. As stated later, these concepts can be redefined using a polar diagram for application in angle-related problems. As the solution to many problems in computational geometry requires some type of angle processing of the input, some other generalizations of the Voronoi diagram based on angles have been studied [9,10,12,13]. Grima et al. proposed a new locus approach for problems with angle processing, called the polar diagram [9]. For any position q in a plane (represented by a point), the region in which q lies belongs to the site with the smallest polar angle. Use of the polar diagram principle can help in solving some important problems that require angle processing in computational geometry. Grima et al. proved a preprocessing step using a polar diagram can speed up the process for many problems in computational geometry [9]. Applications include the convex hull, visibility problems and path-planning problems.

Sadeghi et al. introduced a dual of the polar diagram, the near-pole polar diagram for a set of points, and described some properties and applications [12,13]. Sadeghi et al. also solved a dynamic polar diagram in which new sites can be added to or removed from the point set[14,15]. In the following section we review some concepts and properties needed in subsequent sections. In Section 3, the *NPPD* of points is briefly introduced [13]. In Section 4, the *NPPD* of line segments and simple polygons is obtained. We define the duality of the *NPPD* for points in Section 5 and present an algorithm to find it. Section 6 concludes and identifies some open problems.

2. Polar diagram and its extracted dual (*EDPD*)

The polar angle of point p with respect to s_i, denoted by $ang_{s_i}(p)$, is the angle formed by the positive horizontal line from p and the straight line linking p and s_i (Fig. 1a).

* Corresponding author. Tel.: +98 2414152183; fax: +98 2414152182

1877-7503/$ – see front matter.
doi:10.1016/j.jocs.2011.08.005

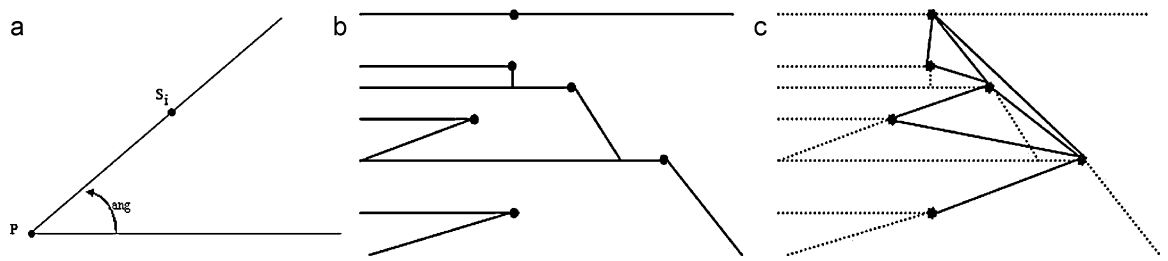

Fig. 1. (a) Polar angle, (b) polar diagram of six points and (c) corresponding extracted dual of the polar diagram (*EDPD*).

Given a set S of n sites (fixed points) in the plane, the locus of points with a smaller positive polar angle with respect to $s_i \in S$ is called the polar region of s_i. Thus,

$$\mathcal{P}_S(s_i) = \{(x, y) \in E^2 | ang_{S_i}(x, y) < ang_{S_j}(x, y); \forall j \neq i\}.$$

The plane is divided into different regions in such a way that if the point $(x, y) \in E^2$ lies in $\mathcal{P}_S(s_i)$, s_i is the first site found when performing an angular scan starting from (x, y). The resulting diagram is called a polar diagram. It is possible to draw an analogy between this angular sweep and the behavior of a radar [10]. Fig. 1b depicts the polar diagram of a set of points in a plane. Grima et al. proved that for a given set S of n points in a plane, the polar diagram of S can be computed in time $\Theta(n\log n)$ using an incremental method [9].

Sadeghi et al. defined a dual of the polar diagram for a set of points [12]. They also defined another graph named the extracted dual of the polar diagram (*EDPD*) (Fig. 1c) and presented an optimal algorithm for finding *EDPD* in time $\Theta(n)$ for sorted sites and $\Theta(n \log n)$ for unsorted sites.

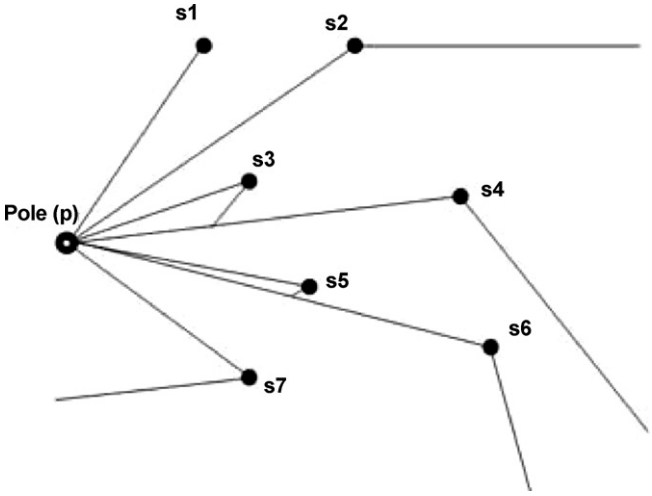

Fig. 2. Near-pole polar diagram of points.

3. Near-pole polar diagram (*NPPD*) for a set of points

In the classic definition of the polar diagram, the pole lies on the left-hand side of the plane at $-\infty$ [12,9]. In *NPPD*, it is assumed that the pole is located on the left-hand side of the sites and close to them. Thus, it can be concluded that this problem is a generalization of the polar diagram. This means that the problem is applicable to more situations; for example, the pole can be considered as the center of vision (eye) of a robot.

In addition to the given point sites, a point p in the plane is also given as a pole and partitioning of the plane will depend on the position of p. Site s_i and the arbitrary point x are also given in the plane. Throughout this paper, we consider the angle formed by the lines ps_i and xs_i as the polar angle. Without loss of generality, assume that pole p is located on the left-hand side of the sites. Fig. 2 shows an example of *NPPD* for seven points with respect to pole p.

In short, a near-pole polar diagram can be described as follows. Initially there is a radar at each of the point sites looking at the pole. These simultaneously start rotating counterclockwise and scan the periphery. The region in the plane observed by radar p_i before other radars is called the region of p_i.

To draw *NPPD* for a set of points, an assumption is made to simplify the problem. It is assumed that the line of view for each site blocks that for any other site that intercepts it later. Fig. 3 depicts two sites s_1 and s_2, the pole p and a point x in the plane as the input. In partitioning of the plane, since $p\hat{s_1}x < p\hat{s_2}x$, x belongs to the region of s_1. In this figure, the line segment ps_2 blocks the line of vision for s_1.

Assumption 1. The line of vision for each site blocks that for any other site that intercepts it later.

Sadeghi et al. presented is an algorithm for drawing the *NPPD* of points in optimal $\Theta(n \log n)$ time [13]. The slope of the line segments $s_ip : i = 1, \ldots, n$ is first computed and the values are sorted. Then a ray starting at pole p rotates clockwise around pole p and sweeps the plane. This algorithm runs in $\Theta(n \log n)$ time.

4. *NPPD* of geometric objects

NPPD of a set of points is reviewed in the previous section. In this section, two algorithms to find *NPPD* of a set of line segments and simple polygons in a plane are presented.

Given a set of line segments or a set of polygons, first the *NPPD* of endpoints and vertices is found using Algorithm 1 from [13], then some pieces of the edges in the diagram are removed. As for

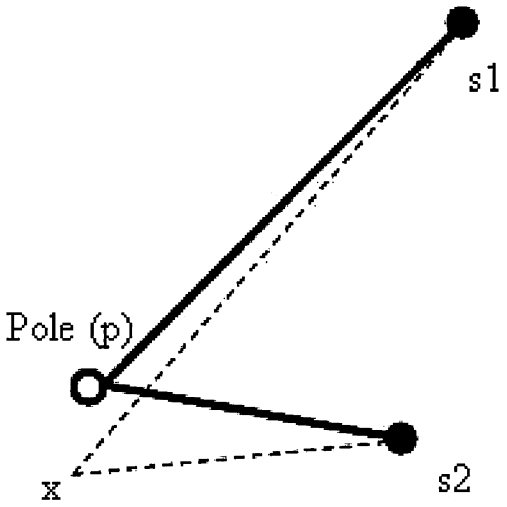

Fig. 3. Assumption: the line segment ps_2 blocks the line of vision for s_1.

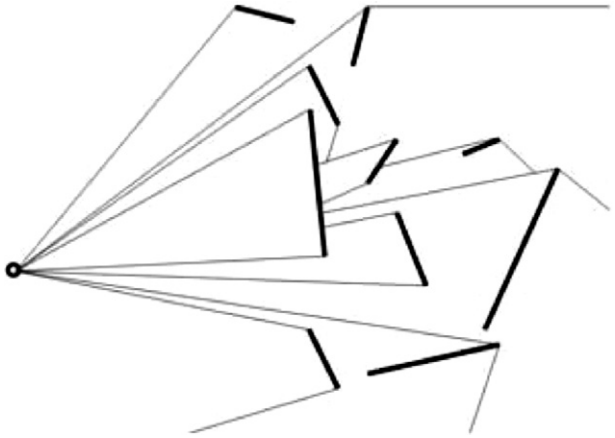

Fig. 4. Near-pole polar diagram of line segments.

the line segment case, the angular sweep starting at some point (x, y) always finds an endpoint. It may be possible to find two endpoints at the same time, but it is not possible to find any middle point for a line segment. Thus, the following lemma can be derived.

Lemma 1. *Every polar edge associated with the NPPD of a set of line segments or a set of polygons in the plane is contained in the NPPD associated with the set of endpoints of the line segments or the polygon vertices, respectively.*

Proof. The sweep line starts from pole p and sweeps the plane in a clockwise manner. In detecting a line segment (or polygon), it detects one endpoint and continues sweeping to reach another endpoint. In other words, the polar regions of middle points are included in the polar regions of the endpoints. This technique is similar to finding the convex hull of a set of segments or polygons in the plane, for which it is sufficient to compute the convex hull of the endpoints or vertices of the polygon. □

Similar to the *NPPD* of a set of points, it is assumed that the line segments are all on the right-hand side of the pole (Fig. 4). To draw *NPPD*, first the *NPPD* of endpoints or vertices is drawn using Algorithm 1 presented by Sadeghi et al. [13]. Then different points for several edges are discarded using a new algorithm. Before presenting the new method, we define the polar right and left.

Definition 4.1. Let p be a pole in the plane. An object O_1 is on the p-right (polar right) of an object O_2 with respect to pole p if the convex hull made up of the pole p and O_1 includes O_2.

Algorithm 1.

Input: n line segments in a plane and a pole p.
Output: *NPPD* of line segments.
01: Extract the endpoints of n line segments.
02: Find the *NPPD* of 2^*n endpoints using Algorithm 1 of Sadeghi et al. [13].
03: If there is any obstacle to the p-right of an endpoint, then discard any edge for which the line equation does not apply to the coordinates of p (Restriction 1).
04: Eliminate all the polar edges that split two sectors of the same polar region (Restriction 2).
05: Discard the portions of polar edges that lie in another region (Restriction 3).

The p-right relation is well defined if and only if object O_2 is a point. In this paper, O_2 is always considered as an endpoint. According to Definition 4.1, the line segment d with endpoints d_1 and d_2 is on the p-right of a point s with respect to pole p if the triangle made up of p, d_1 and d_2 includes s (Fig. 5).

First, we compute the *NPPD* of a set of endpoints of line segments or polygon vertices using Algorithm 1 presented by Sadeghi et al.

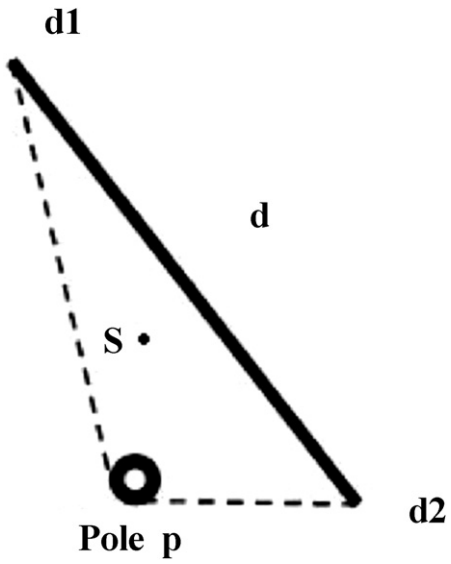

Fig. 5. Segment d is on the p-right of point s with respect to pole p.

[13]. Restrictions are then added to discard some edges or portions thereof. The first three restrictions relate to both line segments and polygons, and the fourth relates to polygons only. Algorithm 1 is used to determine the *NPPD* of line segments.

Fig. 6a shows an example of excluded polar edges (dotted lines) in an *NPPD* of a set of two line segments (Fig. 6b). The rules in Algorithm 1 are numbered according to the four restrictions and are shown in Figs. 6 and 7.

1. If there is any obstacle to the p-right of an endpoint, any edge for which the line equation does not apply to the coordinates of p is discarded (Fig. 6a).
2. A polar edge is eliminated if it splits two sectors of the same polar region (Fig. 6a).
3. If a portion of a polar edge lies in another region, it must be discarded (Fig. 6a).
4. If an edge lies in the object it belongs to, it is discarded (Fig. 7).

Algorithm 2.

Input: n polygons in a plane and a pole p.
Output: *NPPD* of polygons.
01: Extract the vertices of n polygons.
02: Find the *NPPD* of vertices (only for reflex right, start and end vertices) using Algorithm 1 of Sadeghi et al. [13].
03: If there is any obstacle to the p-right of a vertex, then discard any edge for which the line equation does not apply to the coordinates of p (Restriction 1).
04: Eliminate all polar edges that split two sectors of the same polar region (Restriction 2).
05: Discard the portions of polar edges that lie in another region (Restriction 3).
06: Discard all edges lying within the object they belong to (Restriction 4).

Nevertheless, the calculation process for polygons can be improved since not all vertices generate a polar edge. First, we sort the vertices of a polygon in a clockwise manner and present them as $\{v_0, v_1, \ldots, v_{n-1}\}$. If $sl(v_ip)$ denotes the slope of v_ip and $|v_ip|$ indicates the Euclidean distance between two points p and v_i, then the vertices of the polygon are classified as follows.

- Start: $sl(v_{i-1}p), sl(v_{i+1}p) < sl(v_ip)$
- End: $sl(v_{i-1}p), sl(v_{i+1}p) > sl(v_ip)$
- Ordinary left: $sl(v_{i-1}p) < sl(v_ip) < sl(v_{i+1}p), |v_ip| < |v_{i+1}p|, |v_{i-1}p|$
- Reflex reft: $sl(v_{i-1}p) < sl(v_ip) < sl(v_{i+1}p), |v_ip| > |v_{i+1}p|, |v_{i-1}p|$
- Ordinary right: $sl(v_{i-1}p) > sl(v_ip) > sl(v_{i+1}p), |v_ip| > |v_{i+1}p|, |v_{i-1}p|$

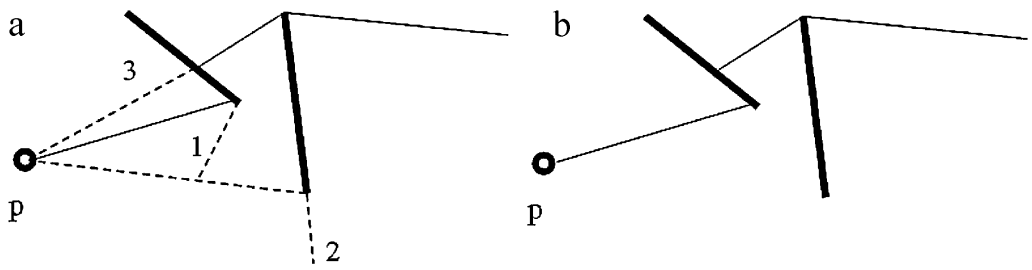

Fig. 6. (a) *NPPD* of endpoints and discarded edges (dotted lines)in an *NPPD* of line segments. (b) *NPPD* of two line segments.

• Reflex right: $sl(v_{i-1}p) > sl(v_ip) > sl(v_{i+1}p)$, $|v_ip| < |v_{i+1}p|$, $|v_{i-1}p|$

In the ordinary left and reflex left cases, the resulting polar edges are completely contained within polar regions that are generated by end and start vertices. As these points belong to the same polygon, then all the polar edges of the left vertices are eventually omitted. Furthermore, the reflex right vertices are omitted because of the fourth restriction. Algorithm 2 is used to determine the *NPPD* of polygons.

Theorem 4.2. *The NPPD of a set of line segments or simple polygons in a plane can be computed in time* $\Theta(n \log n)$*, where n is the number of line segments or polygon vertices.*

Proof. The *NPPD* of a set of sites made up of $2n$ endpoints can be computed in time $O(n\log n)$ using Algorithm 1 of Sadeghi et al. [13]. Every discarding operation needs an additional $O(\log n)$ time to search neighbors to the p-left and p-right of every endpoint. Moreover, it should be noted that there exist at most $O(n)$ discarding operations. In the case of polygons (Fig. 8) with a total of n vertices, the time complexity is the same. Thus, the *NPPD* of n line segments or polygons with $O(n)$ vertices can be found in optimal time $O(n \log n)$, since it has been proved that the problem cannot be solved for a set of points with time complexity less than $O(n \log n)$ [13]. Therefore, the new algorithm is also optimal and consequently its time complexity is $\Theta(n \log n)$.□

In the next section, the extracted dual of *NPPD* for a set of points is introduced and solved. However, the problem is left open for a set of segments and simple polygons. The results can be used in hidden surface removal, in finding the best exit path for a robot from a space containing obstacles, in collision detection and in other similar problems. As mentioned above, the problem is solved for a set of points and the method presented is not applicable in the case of objects.

5. Extracted dual of *NPPD*

In addition to useful applications of *NPPD* of a set of points, the *EDPD* has properties that make it more useful. In fact, the dual of a Voronoi diagram is the Delaunay triangulation, which has many applications. In this section the duality of a polar diagram is

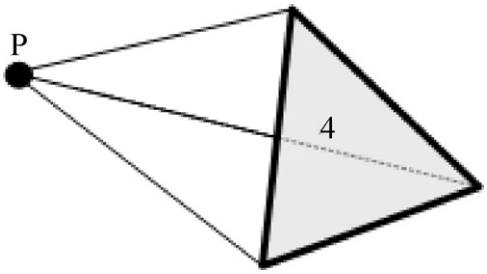

Fig. 7. The edge lying in the triangle is discarded.

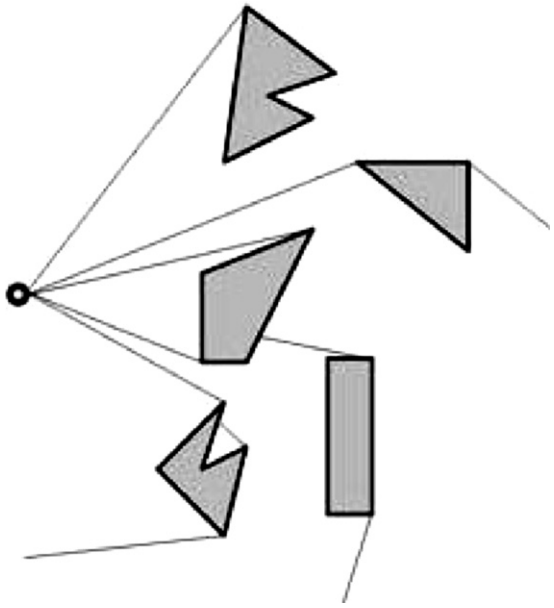

Fig. 8. *NPPD* of polygons.

described; this is not a triangulation, but it can be applied to visibility problems. The *EDPD* for a set of points is defined and an optimal algorithm to find it is presented. This algorithm can be changed to find the *EDPD* for other objects in a plane or space, which is left as an open problem.

Two sites are joined by edge e^* in the dual of *NPPD* if and only if their corresponding faces are separated by edge e in *NPPD*. Therefore, some parallel edges or loops in the dual of *NPPD* may be created. If the loops are omitted and the parallel edges are replaced

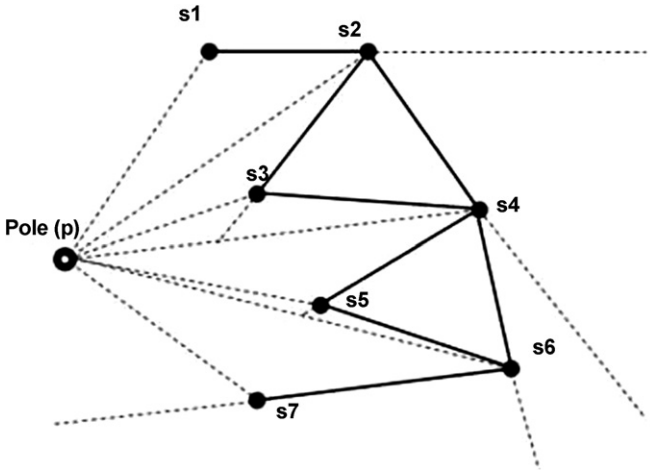

Fig. 9. Extracted dual of *NPPD*.

by one edge, then another graph named the *EDPD* will result (Fig. 9). We now describe an algorithm to find the *EDPD* for a set of points.

5.1. Algorithm

Assume that n sorted point sites in a plane are given and *NPPD* is calculated. The aim is to present an algorithm to draw the *EDPD* and evaluate its validity and complexity.

The regions of two consecutive sites at pole p are adjacent, and thus all lines drawn in Step 1 belong to the *EDPD*. When t_i is on the p-left of line $t_{i+1}t_{i-1}$ (the p-right angle $t_{i+1}t_it_{i-1}$ is less than π), then site t_{i+1} is in the region of t_{i-1}. Thus, the regions of t_{i-1} and t_{i+1} are adjacent and these points connect to each other in Step 6.

Now it must be proved that the algorithm draws all the lines in the *EDPD*. Assume that two sites s_i and s_j are not consecutive. Thus, there exists at least one site between them, named s_k. s_k can be on the p-left or p-right of line s_is_j. If it is on the p-right, then the region of s_k separates the regions of s_i and s_j by the half-line drawn from s_k to the pole p. Thus, s_i and s_j cannot be connected to each other in the *EDPD*. If s_k is on the p-left of line s_is_j, then the angle $s_is_ks_j$ will be less than π and s_i and s_j are connected to each other. Therefore, the algorithm yields a valid solution.

Using the *NPPD* of given sites and sweeping the plane around pole p, we can sort the slope of lines s_ip in time $\Theta(n)$. Step 5 is the most time-consuming and runs in time $\Theta(n)$. In each step, one site is added to or omitted from the stack and no site is added twice. Therefore, the results can be summarized as follows.

Theorem 5.1. *For n given points in a plane and their NPPD, the EDPD can be found in time* $\Theta(n)$.

Calculation of the *EDPD* is more difficult for objects than for points and is still unsolved. The *EDPD* of objects can be used as a preprocessing step for some visibility problems. The problems mentioned and some other open problems are outlined in the following section.

6. Conclusion and future work

After reviewing the near-pole polar diagram problem for a set of points, the same problem was solved for a set of line segments and simple polygons. In both cases, the method discussed is optimal and runs in time $\Theta(n \log n)$. The *EDPD* of a set of points was also defined and an optimal algorithm was presented.

Algorithm 3.

Input: n sites in a plane and *NPPD* of the sites.
Output: *EDPD*.
01: Sort the sites with respect to the slope of lines s_ip; $i - 0, \ldots, n - 1$.
02: Draw a straight line between each two consecutive sites s_i and s_{i+1} for $i = 1, \ldots, n - 1$.
03: Insert $s[1]$ and $s[2]$ into an empty stack T.
04: For $i = 3, \ldots, n$, repeat.
05: When $T \neq$ and $(T[top - 1]T[top]s[i])$ is less than π, repeat.
06: Draw a straight line between $s[i]$ and $T[top - 1]$.
07: Remove the top of the stack $T[top]$.
08: Insert $s[i]$ in the stack.

Calculation of the *EDPD* of objects is an open problem with several visibility applications. For instance, let S be a set of n disjoint polygons in a plane, and let p be a point that is not on any of the polygons of S. Using the *EDPD* of the polygons with respect to pole p, all the polygons of S for which p has weak visibility could be found. The various types of this problem can be seen in communication and radio connection systems, antenna arrangement and target detection. Another set of related open problems arise when Assumption 1 is not considered. Without this assumption, the problem is more difficult and complex. Curved boundaries and disconnected regions result and solving the *NPPD* problem for a set of points (or any objects) is open. Duality and other applications may be further developed and the angle criterion can be applied in β-skeleton and proximity graphs in the future to solve a number of machine vision problems.

References

[1] A. Adamatzky, On excitable β-skeletons, J. Comput. Sci. 1 (2010) 175–186.
[2] O. Aichholzer, F. Aurenhammer, D.Z. Chen, D.T. Lee, Skew Voronoi diagrams, Int. J. Comput. Geom. Appl. 9 (1999) 235–247.
[3] R. Alonso-Sanz, A. Adamatzky, On β-skeletons with memory, J. Comput. Sci. 2 (2011) 57–66.
[4] P.F. Ash, E.D. Bolker, Generalized Dirichlet tessellations, Geom. Dedicata 20 (1986) 209–243.
[5] F. Aurenhammer, Power diagrams—properties, algorithms and applications, SIAM J. Comput. 16 (1987) 78–96.
[6] F. Aurenhammer, Voronoi diagrams a survey of a fundamental geometric data structure, ACM Comput. Surv. 23 (1991) 345–405.
[7] F. Aurenhammer, H. Edelsbrunner, An optimal algorithm for constructing the weighted Voronoi diagram in the plane, Patt. Recog. 17 (1984) 251–257.
[8] S. Fortune, Voronoi diagrams and Delaunay triangulations, in: D.-Z. Du, F.K. Hwang (Eds.), Computing in Euclidean Geometry, World Scientific Publishing, Singapore, 1992, pp. 193–233.
[9] C.I. Grima, A. Márquez, L. Ortega, A new 2D tessellation for angle problems: the polar diagram, Comput. Geom. 34 (2006) 58–74.
[10] C.I. Grima, A. Márquez, L. Ortega, A locus approach to angle problems in computational geometry, in: Proceedings of 14th European Workshop in Computational Geometry, Barcelona, Spain, 1998.
[11] D.-T. Lee, Two-dimensional Voronoi diagrams in the Lp-metric, J. Assoc. Comput. Mach. 27 (1980) 604–618.
[12] B. Sadeghi Bigham, A. Mohades, The dual of polar diagrams and its extraction, in: Proceedings of the International Conference on Computational Methods in Sciences and Engineering (ICCMSE), Greece, 2006.
[13] B. Sadeghi Bigham, A. Mohades, Polar diagram with respect to a near pole, in: Proceedings of the 23rd European Workshop on Computational Geometry EWCG07, Austria, 2007, pp. 206–209.
[14] B. Sadeghi Bigham, A. Mohades, L.M. Ortega, Dynamic polar diagram, Inf. Process. Lett. 109 (2008) 142–146.
[15] M. Nouri Bygi, M. Ghodsi, Polar diagram of moving objects, in: Proceedings of the 20th Canadian Conference on Computational Geometry (CCCG2008) August 13–15, McGill University, Montreal, Canada, 2008, pp. 51–54.

Bahram Sadeghi Bigham, currently working as Dean of Department of IT. He obtained B.Sc., Applied Mathematics, 1993–1996 from Birjand University, Birjand, Iran. He obtained M.S., Applied Mathematics, Thesis title: "Solving the problem Shortest Path using parallel algorithms" 1998–2000 from Amirkabir University of Technology, Tehran, Iran. He obtained Ph.D., Computer Sciences, "Computational Geometry", Thesis title: Angle sensitive Voronoi Diagrams, 2004–2008 from Amirkabir University of Technology, Tehran, Iran.