Polar Diagram of Moving Objects*

Mojtaba Nouri Bygi† Mohammad Ghodsi‡

Abstract

Many important problems in computational geometry needs to perform some kinds of angle processing. The *polar diagram* [4] is a locus approach for problems that process angles. Using this structure in the preprocessing phase, one can eliminate exhaustive search to find objects in the scene with smallest angle according to a point. For example, using this new partition, Jarvis's march method in finding convex hull of n objects will reduce to optimal $O(n \log n)$ time by replacing an $O(n)$ search problem in an optimal $O(\log n)$ location operation.

In this paper, we use the notion of the kinetic data structure (or KDS) to maintain the the polar diagram of a set of continuously moving objects in the scene. KDS is one of the design and analysis tools used in modeling of moving geometric objects, and can be used to maintain certain attributes of a set of objects moving in a continuous manner. We show that our proposed structure meets the main criteria of a good KDS.

1 Introduction

Most of the solutions for geometric problems are optimum in the worst case. However, if the size of the result is small or we need to get answers for several problem instances, these solutions may not be suitable. In these case, algorithms that preprocess the scene and then provide answers to each query with a better performance are widely used in this field.

C. I. Grima et al. [4, 5] introduced the concept of the *Polar Diagram*. The polar diagram of the scene consisting of n objects is a partition of plane to *polar regions*. Each object creates a polar region representing the locus of points with common angular characteristics in a starting direction. If point p lies in the polar region of object o, we know that o is the first object found after performing an angular scanning from the horizontal line crossing p in counterclockwise direction. The computation of the polar diagram can be done using the divide and conquer or the incremental methods, both working in $\Theta(n \log n)$, which is optimum. By using this tessellation as the preprocessing phase, we can avoid other angular sweeps by locating a point into a polar region in logarithmic time [4].

Kinetic Data Structure (KDS) is a framework for maintaining certain attributes of a set of objects moving in a continuous manner. For example, KDS has been used for maintaining the convex hull of moving objects, or maintaining the closest distance among moving objects. A KDS consists of mainly two parts: a description of the needed attributes with some certificates such that as a certificate remains unchanged as long its attribute does not change. It is assumed that we can efficiently compute the failure time of each of these certificates. In such events that a certificate fails, the KDS must be updated. Until the next event, the current KDS remains valid. See the survey by Guibas [3] for more background on KDS and its analysis.

In this paper, we first propose an improved algorithm for computing the polar diagram of a set of line-segments or polygons. Then we use the notion of KDS to model maintain the polar diagram of a set of continuously moving objects in the scene. We show that our proposed structure meets the main criteria of a good KDS.

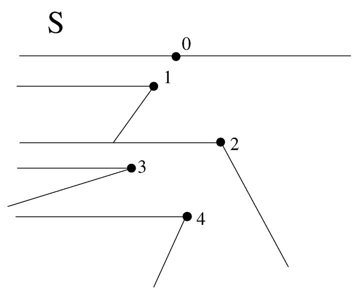

Figure 1: The polar diagram of a set of points in plane.

The rest of this paper is organized as follows: In Section 2, we define our kinetic configuration for the polar diagram, and in Section 2.2, we see what happens when the objects move in the plane. In Section 3.1.1, we give a one-step algorithm to compute the polar diagram of line-segments and polygons, which we will use in Section

*This research was in part supported by a grant from IPM (No. CS2386-2-01.)

†Department of Computer Engineering, Sharif University of Technology, P.O. Box 11365-9517, Tehran, Iran, nouribaygi@ce.sharif.edu

‡Department of Computer Engineering, Sharif University of Technology, P.O. Box 11365-9517, Tehran, Iran, and IPM School of Computer Science, Niavaran, Tehran, Iran. ghodsi@sharif.edu

3.1 for KDS of line-segments and polygons. Finally, in Section 3.2 we see the case for circular objects.

2 Kinetic Configuration

In this section we present a model for kinetic behavior of the polar diagram for different situations. Given a set of points moving continuously, we are interested in knowing at all times the polar diagram of the scene.

2.1 Proof Scheme

For simplicity of discussions, we assume that our objects are points in 2D. We state that each edge of the polar diagram is called a *polar edge*. We also define a *pivot* of an object to be the second object that lies on the polar edge passing through it, e.g., in Figure 1, the pivot of s_4 is s_2 and the pivot of s_2 is s_0.

We claim that if we have the sorted list of objects according to their y-coordinates, and the pivot of each object, we will have a unique polar diagram.

Suppose that there are n points on the scene. For our proof scheme, we maintain two kinds of information about the scene: we maintain the vertically sorted list of objects, and for each object its current pivot. As we will show shortly, these data is sufficient for the uniqueness of our polar data, i.e. only if one of these conditions change, the polar structure of the scene will change.

So, we will have two kinds of certificates: First we need $n-1$ certificates for storing the sorted list of objects. For instance, if the sorted list of objects is $s_{i_0}, s_{i_1}, \ldots, s_{i_{n-1}}$, we need the following certificates.

$$\begin{gathered} s_{i_0} < s_{i_1} \\ s_{i_1} < s_{i_2} \\ \cdots \\ s_{i_{n-2}} < s_{i_{n-1}} \end{gathered}$$

For stating the pivot of each object, we need n more certificates, each indicating a object and its pivot in the polar diagram. In total, our proof scheme consists of $2n-1$ certificates.

2.2 Events and Event Handling

Once we have a proof system, we can animate it over time as follows. As stated before, each condition in the proof is called a certificate. A certificate fails if the corresponding function flips its sign. It is also called an event happens if a certificate fails. All the events are placed in a priority queue, sorted by the time they occur. When an event happens, we examine the proof and update it. An event may or may not change the structure. Those events that cause a change to the structure are called *exterior events* and other events are called *interior events*. When the motion of an object changes, we need to reevaluate the failure time of the certificates that involve that object (this is also called *rescheduling.*).

As there are two kinds of certificates in our proof scheme, it is obvious that there must be two kinds of events:

- **pivot event**, when three objects, one of which is the pivot of another one, become collinear.
- **horizontal event**, when two objects have the same y-coordinate (have a same horizontal level)

In the former case, we must update the certificates relating to the sorted sequence of two neighboring points, which is at most three certificates (two, if one of the points is a boundary point, i.e. top-most or button-most points). In the latter case, one certificate becomes invalid and another certificate (indicating the new pivot of the object) is needed. As we will show, other certificates will remain still.

Lemma 1 *When an event is raised, the objects above the other object(s) which raised the event do not change their polar structures.*

Proof: From the incremental method used for the construction of the diagram of a set of points [4], we know that there is no need to know about the state of objects below an object to determine its pivot, so when an object changes its state, it will not affect its above objects.

We can also say that an angular sweep that starts from the horizontal direction would never intersect any objects below this initial horizontal line (by definition, the top-most object has no pivot). □

Pivot event:

First, we consider the simplest case when the lowest object is moving. Figures 2 and 3 show these cases, where s_2 is moving. In Figur 2, s_0 is the pivot of s_2. While s_2 is moving left, the line segment $s_0 s_2$ coincides with the object s_1 (note that there may be other objects between s_0 and s_2, but we are only interested in s_1). At the moment that three objects s_0, s_1, and s_2 become collinear, s_1 will occlude s_0 from s_2 and it no longer can be its pivot. From this event on, s_1 becomes the new pivot of s_2. Similarly, in Figure 3, s_1 is the pivot of the moving object s_2. When three objects s_0, s_1, and s_2 become collinear (again, there may be other objects between each pair of these objects, but we are not interested in them), s_2 needs to change its pivot which becomes s_0.

As we assumed that no other object other than s_2 is moving, form lemma 1 we know that there will be no change in other objects, so at this event, only one

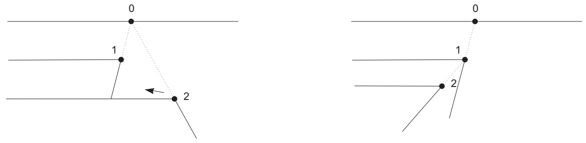

Figure 2: A pivot event. As s_2 moves left, s_0, s_1 and s_2 become collinear.

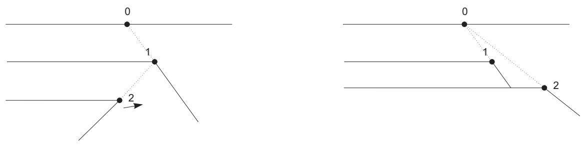

Figure 3: A pivot event. As s_2 moves right, s_0, s_1 and s_2 become collinear.

certificate becomes invalid and it must be replaced by another certificate indicating the new pivot of the moving object. It is clear that upon occurring this event, the processing of the event and changing of proof scheme can be done in $O(1)$ and $O(\log n)$, respectively (we need to find the corresponding certificate in the certificates list).

We now see what happens to the second lowest object (see Figures 4 and 5, where s_2 is moving right). In Figure 4, s_1 is the pivot of s_2, and also the pivot of the lower object s_3. While moving, there will be a time that s_2 occludes the lower object s_3 from its pivot. In Figure 4 it is when the objects s_1, s_2 and s_3 become collinear. At this time, although there is no change in the polar structure of the moving object s_2, there is a change in the lower object s_3, and we must update the proof scheme accordingly. If s_2 continues its motion, there will be a pivot event (see Figure 5) that its polar structure is changing.

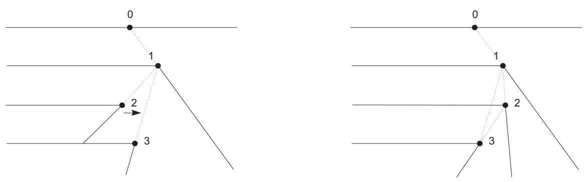

Figure 4: While moving, s_2 can change the pivot of each of its below objects by occluding their initial pivots.

Lemma 2 *The changes in the structure of an object caused by moving an above object, would not cause any other change in other objects.*

Proof: The structure of each object is determined by the first object that encountered by an angular sweep. As we assume that no other objects is moved, this encountered object would not change. □

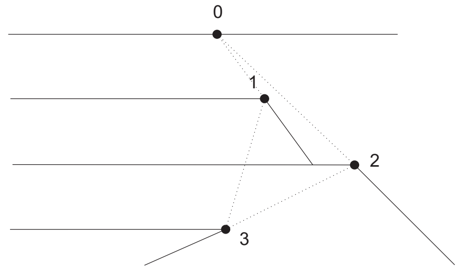

Figure 5: For each moving object, there is one pivot event when its own pivot will change.

From above discussions, we can deduce that if an object is moving in the scene and there are k other objects below it, there can be up to k pivot events changing the structure of the below objects, and one pivot event changing its own structure. Each of these events can be processed in $O(1)$ time and the change in the proof scheme can be done in $O(\log n)$.

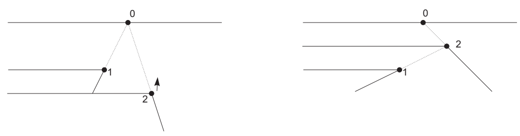

Figure 6: When two objects s_1 and s_2 lay on a same horizontal level, a horizontal event is occurred and the polar structure will change.

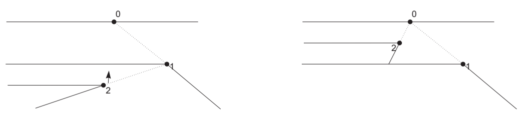

Figure 7: In a horizontal event, only one of the objects will change its pivot.

Horizontal event:

In these events, one of the situations of Figures 6 and 7 will happen. As we can see, only one of the objects will change its pivot (set it to the third object). This change of configuration is equal to changing three or four certificates in proof scheme: one for a change in one of the object's pivot, and three or two for change in vertical order of objects.

We will now show that no more changes is needed. Assume that in a small interval before and after the horizontal event, no other pivot events would occur. From lemma 1, we know that there would be no change in the above objects. What about the below objects? We can see that for a change in the pivot of an object, there must be an occlusion between the objects and its previous pivot, and this means that three objects must lie on the same line, i.e. we need a pivot event (see Figure 8).

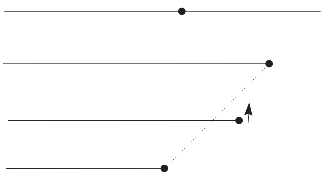

Figure 8: Only upon occurring a pivot event the structure of other objects will change.

Theorem 3 *Each of the events in the kinetic polar diagram of a set of points takes $O(\log n)$ time to process and causes $O(1)$ changes in the proof scheme.*

Proof: For horizontal events, we need to update at most three certificates, we just need to find these certificates in the proof scheme and replace them with the new ones, which takes $O(\log n)$ time. We also need to update one pivot certificate with the same cost. The same holds for pivot events, which we need to find and update $O(1)$ pivot certificates. □

Theorem 4 *The initial event list can be built in $O(n \log n)$ time, using a suitable event queue.*

Proof: As there are $O(n)$ certificates in our proof scheme, and for each moving object. we can find the first certificate that it will violate by a simple $O(\log n)$ search. The proof is straightforward. □

3 Other Geometric Objects

Grima et al. proposed the polar diagram of geometric objects as a new plane partition with similar characteristics to the polar diagram of points [4]. In this section we extend our kinetic model to include these types of objects as well.

3.1 Line Segments and Polygonal Objects

3.1.1 Polar Diagram of Line Segments and Polygons

The two-step algorithm for calculating the polar diagram of a set of line segments and polygons described in [4] is not suitable for our purpose and we modify it so that it can be used in constructing the KDS of polar diagram.

In our proposed modification, the final set of polar edges is calculated at the same time as the incremental method is processed. In each step, we decide whether or not the edges, horizontal or oblique, will be added.

Every polar edge associated with the polar diagram of a set of line segments or polygons, is contained into the polar diagram of the set of points made up of their line segments or polygon vertices [4]. As the incremental method is processed, for each point of this set, we have the following rules:

For horizontal polar edges, we can say:

1. If there is any obstacle to the right of an endpoint, the horizontal polar edge will be added.
2. If we decided to add a horizontal edge according to the previous rule, and it reaches any obstacle, its left portion will be discarded.

For oblique polar edges, the oblique edge is added if none of the following rules applies:

3. If there is any obstacle to the right of an endpoint.
4. If the oblique edge lies to the left of the belonging line segment or inside the polygon.
5. If the pivot of the endpoint is the other endpoint of the same line segment.

Figure 9 shows an example of applying this algorithm (with discarded edges as dotted lines) in the polar diagram of a set of line segments.

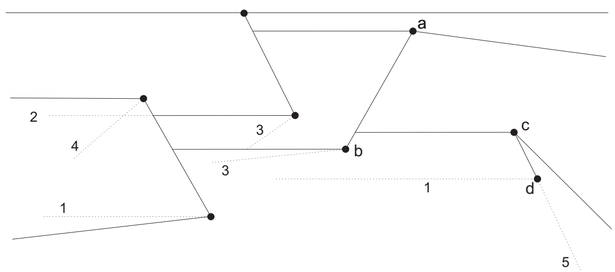

Figure 9: Discarded edges in a polar diagram of line segments.

3.1.2 KDS of the Polar Diagram of Line Segments and Polygons

As stated in Section 3.1.1, the given algorithm for calculating the polar diagram of a set of line segments or polygons, is based on the incremental process of finding the polar diagram of a set of points; the only difference is in applying the specified rules to decide about the appearance of each polar edge in the polar diagram at each step. Therefore, the events which degenerate the certificates of this kinetic configuration is a subset of the events of the KDS of a set of points. The following theorem shows that the events of the line segments and polygons case are exactly the same as the events of the point objects.

Theorem 5 *The events of the kinetic polar diagram of a set of line segments and polygons are exactly the same as those of the set of points made up of line segment endpoints or polygon vertices.*

Proof: As discussed before, according to the proposed algorithm for finding the polar diagram of a set of line segments or polygons, the events of this case include the events of point objects case. We only need to prove that according to specified rules of the proposed algorithm, the decision on the appearance of polar edges could only change at the events for the set of related points:

1. If an oblique polar edge is discarded because of an obstacle to its right, the only way that this position changes and the obstacle leaves its right (which, then we must draw the oblique edge) is in a horizontal event. For example in Figure 9 the segment ab must move in such a way that b passes segment cd and then we must draw b's oblique edge. During this movement a horizontal event will occur when b passes d and this situation can be handled with horizontal event.

2. If an oblique edge is discarded because of lying to the left of belonging line segment or inside the polygon, the only movement that can change this situation, as an event is the movement of the bottom endpoint of the segment and swapping its location with the upper one. During this movement a horizontal occurs.

3. If an oblique edge is discarded because one of its endpoint is pivot of its other endpoint, the only way that this situation can change is the movement of the segment in such a way that the two endpoints of the segment lies in a same horizontal level, which is a horizontal event.

4. If a horizontal line is partially added because of reaching an obstacle, the motion that can change this situation and causes an event is the movement of the endpoint to pass the obstacle, which is a horizontal event.

5. If a horizontal line is added because of an obstacle to its right, when the obstacle moves and clear the right of the point a horizontal event will occur one of the endpoints of the obstacle passes the point.

This completes the proof. □

3.2 Circular Objects

For circular objects, we use a similar approach to that of previous section about the line segments. For our proof scheme, we maintain a sorted list of all $2n$ north and south poles. It can be done by $2n-1$ certificates. Also, for each oblique polar edge, we add a certificate, denoting its main object and its pivot. As there may be up to $3(n+1)-6$ such edges [4], we may have up to $3n-3$ such certificates. Like the point objects case, we have two kinds of events upon moving of objects: horizontal events and pivot events. As we will see, while handling these events, there might be one other type of change in polar structure which we are not interested in, i.e. as we used a lazy structure for our proof scheme, we do not consider this type of change. This is when a polar edge is occluded by another object in its way.

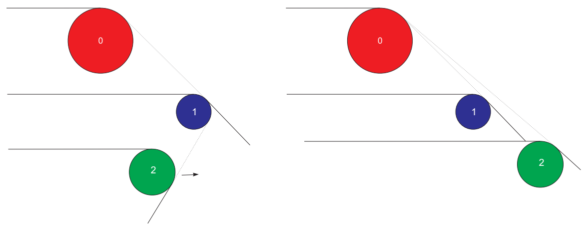

Figure 10: As three objects s_0, s_1 and s_2 form a tri-tangent, a pivot event will occur.

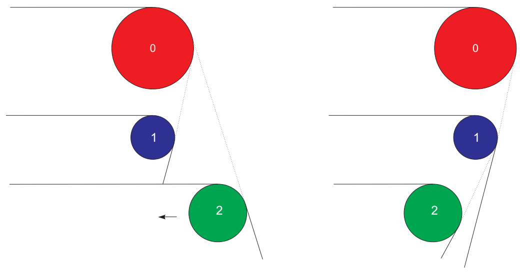

Figure 11: A pivot event.

Pivot event:

These events are essentially the same as those for point objects. As we can see in Figures 10 and 11, when three objects become tri-tangent, there is a potential pivot event: when one of them is pivot of another one, we have a pivot event. In these events, the object that has its pivot in trio will change its pivot and we need to replace the corresponding certificate in proof scheme with a another one.

Horizontal event:

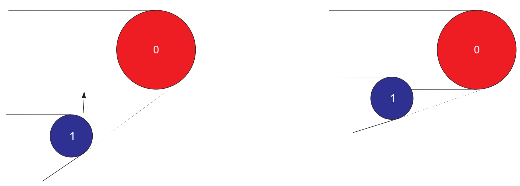

Figure 12: A horizontal event. A polar edge from a South pole will appear.

As there are $2n$ poles for n circular objects, the processing of horizontal events are a little different from those of point objects. Figures 12 and 13 shows the cases where two different pole types lay on a same horizontal level. As we can see, in the case of Figure 12, a new polar edge from a South pole appears, and in case of Figure 13, a previous present polar becomes occluded.

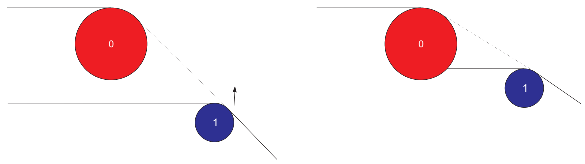

Figure 13: A horizontal event. A polar edge will be occluded.

As we said before, we take non of these changes in polar structure in our proof scheme, and we only need to update certificates corresponding to the vertical order of poles.

Another type of horizontal event occurs when two pole of the same kind (north or south) lie on a horizontal line (Figures 14 and 15). Apart from appearing or occluding of polar edges, there might be another change in the polar structure. In these cases, an oblique edge can appear (Figure 14) or disappear (Figure 15). So we need to add or remove the corresponding certificates indicating the oblique polar edge.

Figure 14: A horizontal event. A polar edge from a South pole and an oblique edge will appear.

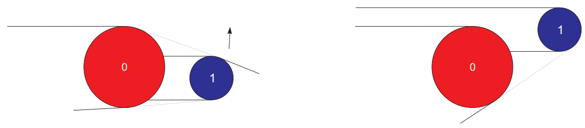

Figure 15: A horizontal event. A polar edge will no be occluded anymore.

From above discussions we can deduce the following proposition.

Proposition 6 *Each of the events in kinetic polar diagram of a set of circles takes $O(\log n)$ time to process and it has $O(1)$ changes in the proof scheme.*

4 KDS Evaluation

In this section we evaluate our kinetic model according to the criteria of a *good* KDS. Similar to other algorithms, a good KDS should take small space, small initialization cost, and efficient update time. In KDS, an update may happen in two cases. One is when a certificate fails and an event happens. The other is when the motion of an object changes. In the first case, we need to update the certificate set, and in the second case we must recompute the failure times for all the certificates that involve that object. These requirements induce the following quality measurements for KDSs [2].

Compactness: the size of the proof.

Responsiveness: the time to process an event.

Locality: the number of certificates that a single object involves in.

Another crucial efficiency factor of a KDS is the number of events processed. This factor determines the number of times we need to stop and check our proof and structure. This factor is expressed by efficiency:

Efficiency: the number of events processed.

Now, we consider each of the above criteria in our kinetic model.

Compactness. The structure clearly takes linear space. As we stated in Section 2.1, for a set of n point objects, the proof scheme consists of $n-1$ certificates for sorted vertical order of objects and n certificates for maintaining the pivots of each object, so in total, our proof scheme have $2n-1$ certificates.

Responsiveness. It is $O(\log n)$ for processing an event as there are $O(1)$ certificates need to reschedule. Each reschedule takes $O(\log n)$ time.

Locality. Each object is involved in at most three certificates.

Efficiency. All events are exterior—the ordering changes once a horizontal event happens, or the pivot of an object changes once a pivot event happens. The number of events is bounded by $O(n^2)$ as any two points can exchange their ordering only constant number of times for constant degree algebraic motions, and any point is a potential candidate for being the pivot of another point.

5 Conclusion and Future Work

In this paper we studied the concept of polar diagram, which is a new locus approach for problems processing angles, and KDS, which is a structure that maintains certain attributes of a set of continuously moving objects in the scene. We used KDS to model the behavior of a polar diagram when our scene is dynamic, i.e. we maintain the polar diagram of a set of continuously moving objects. We showed that our proposed structure meets the main criteria of a good KDS.

Following our defined model for kinetic polar diagram, we can use it in direct applications of polar diagram to maintain the computed attributes. For example, we can use kinetic polar diagram for maintaining the convex hull of a set of moving objects with low cost.

References

[1] J. Basch, L. J. Guibas, and J. Hershberger. *Data structures for mobile data.* In Proc. 8th ACM-SIAM Sympos. Discrete Algorithms, pages 747–756, 1997.

[2] J. Basch. *Kinetic data structures*. PhD thesis, Stanford University, Palo Alto, California, 1999.

[3] L. Guibas. *Kinetic data structure: a state of art report.* In Proc. 3rd Workshop Algorithmic Found. Robot., pages 191–209, 1998.

[4] C.I. Grima, A. Márquez, and L. Ortega. *A new 2D tessellation for angle problems: The polar diagram.* In Computational Geometry, Theory and Application, 34(2): 58-74, 2006.

[5] C.I. Grima, A. Márquez, and L. Ortega. *A locus approach to angle problems in computational geometry.* In Proc. of 14th European Workshop in Computational Geometry, Barcelona, 1998.