Kinetic Polar Diagram

Mojtaba Nouri Bygi[1,2], Fatemeh Chitforoush[1],
Maryam Yazdandoost[1], and Mohammad Ghodsi⋆[1,2]

[1] Department of Computer Engineering, Sharif University of Technology,
P.O. Box 11365-9517, Tehran, Iran
[2] School of Computer Science, Institute for Studies in Theoretical Physics and Mathematics,
P.O. Box 19395-5746, Tehran, Iran

Abstract. Polar Diagram [4] is a new locus approach for problems processing angles. The solution to many important problems in Computational Geometry requires some kind of angle processing of the data input. Using the Polar Diagrma as preprocessing, exhaustive searches to find those sites with smallest angle become unnecessary.
In this paper, we use the notion of kinetic data structure [1][2] to model the dynamic case of polar diagram, i.e we 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.

Key words: Polar diagram, Kinetic data structures, Geometric events, Computational Geometry

1 Introduction

C. I. Grima et al. [4] introduced the Polar Diagram. The polar diagram of the scene E, consisting of n two-dimensional objects, $E = \{o_0, o_1, \ldots, o_{n-1}\}$, denoted as $\mathcal{P}(E)$, is a plane partition in polar regions. Each generator object o_i creates a polar region $\mathcal{P}_E(o_i)$ representing the locus of points with common angular characteristics in a starting direction. Any point in the plane can only belong to a polar region, which determines its angular situation with respect to the rest of generator objects in the scene. More specifically, if point p lies in the polar region of object o_i , $p \in \mathcal{P}_E(o_i)$, we know that o_i is the first object found after performing an angular scanning from the horizontal line crossing p in counterclockwise direction. The polar diagram can be computed efficiently using the Divide and Conquer or the Incremental methods, both working in $\Theta(n \log n)$. The strength of using this tessellation as preprocessing is avoiding any angular sweep by locating a point into a polar region in logarithmic time [4].

A KDS is a structure that maintains a certain attribute of a set of continuously moving objects. It consists of two parts: a combinatorial description of the attribute and a set of certificates with the property that as long as the outcomes of the certificates do not change, the attribute does not change. It is assumed that each object follows a known trajectory so that one can compute the failure time of each certificate. Whenever a certificate fails – we call this an event – the KDS must be updated. The KDS remains valid

⋆ This author's work was in part supported by a grant from IPM (N. CS2386-2-01).

until the next event. See the excellent survey by Guibas [3] for more background on KDSs and their analysis.

In this paper we use the notion of kinetic data structure to model the dynamic case of polar diagram, i.e we 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.

The rest of this paper is organized as follows: In section 2.1 we define our kinetic configuration for Polar Diagram, and in section 2.1 we see what happens when the objects move in the plane. In section 2.2 we extend our model for circular objects.

2 Kinetic Polar Diagram

In this section we present a model for kinetic behavior of 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 Kinetic Configuration

Proof Scheme For simplicity of disscussions, we assume that our objects are points in 2D. In Section 2.2 we will show that our model is also valid for circular objects.

We claim that if we have the sorted list of objects according to their y-coordinates, and the for each object, its *pivot*, the second object that lies on the polar edge passing the object, we will have a unique polar diagram.

Suppose there are n points in the scene. For our proof scheme, we maintain two kinds of information about the scene: we maintain the vertically sorted list of sites, and for each site 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: $n-1$ certificates will indicate the sorted list of sites. For instance, if the sorted list of sites is $s_{i_0}, s_{i_1}, \ldots, s_{i_{n-1}}$, we need the certificates $s_{i_0} < s_{i_1}, s_{i_1} < s_{i_2}, \ldots, s_{i_{n-2}} < s_{i_{n-1}}$.

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

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 those not *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 event:

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

In the former case, we must update the certificates relating to sorted sequence of two neighbor 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 site) is needed. As we will show, other certificates will remain still.

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

Proof: From the incremental method used for the construction of the polar digram of a set of points [4] we know that there is no need to know about the state of objects below a site to determine its pivot object. 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 site has no pivot). □
Pivot event:
First, we consider the simplest case, i.e. when the lowest object is moving. Figures 1 and 2 show these cases, where s_2 is moving. In Figure 1, s_0 is the pivot of s_2. While s_2 is moving left, the line segment s_0s_2 is coinside with the site s_1 (note that there may be other sites between s_0 and s_2, but we are only interested in s_1). At the moment that three sites s_0, s_1, and s_2 become collinear, the s_1 will occlude s_0 from s_2 and it no longer can be its pivot. From now on, s_1 becomes the new pivot of s_2. Similarly, in Figure 2, s_1 is the pivot of moving site s_2. When three sites s_0, s_1, and s_2 become collinear (again, there may be other sites between each pair of these sites, but we are not interested in them), s_2 needs to change its pivot which becomes s_2.

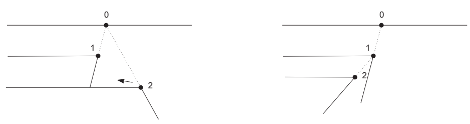

Fig. 1. A pivot event. As s_2 moves left, s_0, s_1 and s_2 become collinear.

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 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

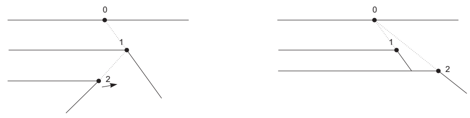

Fig. 2. A pivot event. As s_2 moves right, s_0, s_1 and s_2 become collinear.

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).

Now we see what happens to the second lowest site (see Figures 3 and 4, where s_2 is moving right). In Figure 3, s_1 is the pivot of s_2, and also the pivot of the lower site s_3. While moving, there will be a time that s_2 occlude the lower site s_3 from its pivot. In Figure 3 it is when the sites s_1, s_2 and s_3 become collinear. At this time, although there is no change in polar structure of moving site s_2, there is a change in the lower site s_3, and we must update the proof scheme accordingly. If s_2 continues its motion, there will be a pivot event (see Figure 4) that its polar structure is changing.

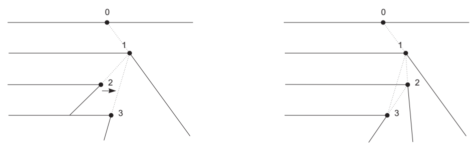

Fig. 3. While moving, s_2 can change the pivot of each of its below sites by occluding their initial pivots.

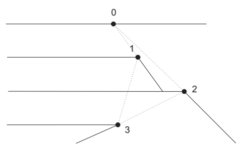

Fig. 4. For each moving site, there is one pivot event when its own pivot will change.

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

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

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

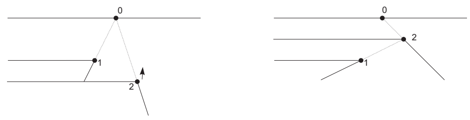

Fig. 5. When two sites s_1 and s_2 lay on a same horizontal level, a horizontal event is occurred and the polar structure will change.

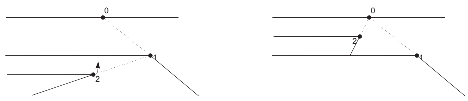

Fig. 6. In a horizontal event, only one of the sites will change its pivot.

Horizontal event:

In these events, one of the situations of Figures 5 and 6 will happen. As we can see, only one of the sites 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 site's pivot, and three or two for change in vertical order of sites.

Now we 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 sites? We can see that for a change in the pivot of a site, there must be an occlusion between the sites and its previous pivot, and it means that three sites must lay on a same line, i.e. we need a pivot event (see Figure 7).

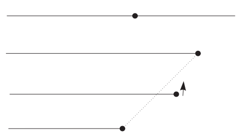

Fig. 7. Only upon occurring a pivot event the structure of other sites will change.

Theorem 1. *Each of the events in kinetic polar diagram of a set of points takes $O(\log n)$ time to process and causes has $O(1)$ changes in 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 thing is holds for pivot events, which we need to find and update $O(1)$ pivot certificates. □

Theorem 2. *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 violates by a simple $O(\log n)$ search, the proof is straightforward. □

2.2 Circular Objects

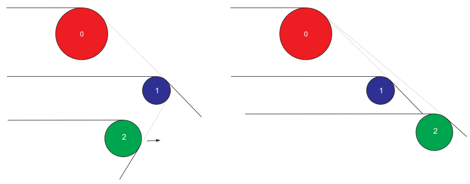

Fig. 8. As three objects s_0, s_1 and s_2 form a tri-tangent, a pivot event will occur.

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.

Pivot event:

These events are essentially the same as those for point objects. As we can see in Figure 8, 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:

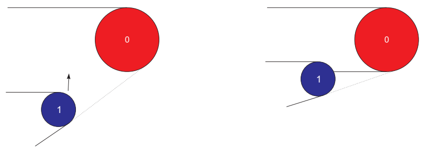

Fig. 9. A horizontal event. A polar edge from a South pole will appear.

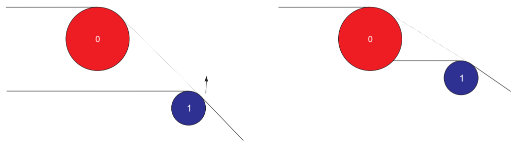

Fig. 10. A horizontal event. A polar edge will be occluded.

As there are $2n$ poles for n circular objects, the processing of horizontal events are a little different from those of point objects. Figures 9 and 10 shows the cases where two different pole types lay on a same horizontal level. As we can see, in the case of Figure 9, a new polar edge from a South pole appears, and in case of Figure 10, a previous present polar becomes 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) lay on a horizontal line (Figures 11). Apart from appearing or occluding of polar edges, there might be another change in polar structure. In these cases, an oblique edge can appear (Figure 11) or disappear. So we need to add or remove the corresponding certificates indicating the oblique polar edge.

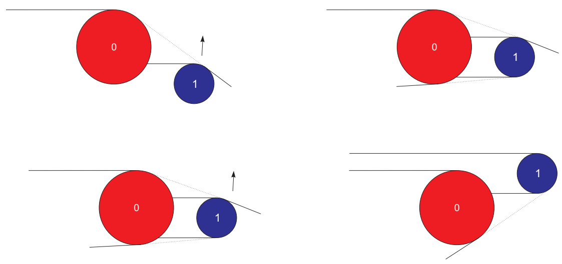

Fig. 11. Horizontal events. (a) A polar edge from a South pole and an oblique edge will appear. (b) A polar edge will no be occluded anymore.

From above discussions we can deduce the following proposition.

Proposition 1. *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 proof scheme.*

2.3 KDS Evaluation

Evaulation of a good kDS depends on some properties [2]. Here we consider thease properties in our kinetic model.

Compactness The size of the proof. 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 The time to process an event. $O(\log n)$ for processing an event as there are $O(1)$ certificates need to reschedule. Each reschedule takes $O(\log n)$ time.

Locality The number of certificates that a single object involves in. Each object is involved in at most three certificates.

Efficiency The number of events processed. All the 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.

3 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 a certain attribute of a set of continuously moving objects among moving objects. 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 a very 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*, 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.