Collision Detection using Polar Diagrams

L. Ortega and F. Feito
University of Jaén
E.P.S Jaén
Avd. Madrid n^{o} 35
23071, Jaén, Spain
{lidia|ffeito}@ujaen.es

C. Grima and A. Márquez
University of Seville
Escuela de Informática
Avd. Reina Mercedes s/n
41012, Seville, Spain
{grima|marquez}@us.es

ABSTRACT

In Computer Graphics, Collision Detection is considered a key problem with important applications in related areas. Several solutions have been proposed, but independently of the chosen strategy, different stages of the solution are considered. It is not only relevant to apply a good static collision test, but reducing the pairs of objects susceptible of intersecting is important in order to obtain applicable computation times. Some methods achieve this aim computing a tessellation as preprocessing. The new approach we propose in this paper computes a partition of the plane called Polar Diagram, in which every object in the scene is owner of a polar region as the locus of points with some common angle properties. Polar diagrams used as preprocessing can be applied to Collision Detection and many other geometric problems where the solution is given by angle processing.

Keywords
Collision Detection, Plane tessellation, Visibility problems.

1. INTRODUCTION

Collision Detection has been extensively studied in fields like Computational Geometry and Computer Graphics, and can be considered as one of the Robotics fundamental pillars. In Animation or Robotics, whenever some objects representing a scene are provided of movement, it is necessary to determine if these mobile objects collides with others, sometimes to avoid possible contacts or just to know the collision response. The results have been applied to Virtual Reality, Computer-Aided Design and Physical Simulation, in general. The state of the art is described in some works like [Jim01a, Lin98a, Osu01a].

Collision detection can be classified attending to different approaches: spatio-temporal intersection, swept volume interference, multiple interference detection or trajectory parameterization. However, computation times are usually crucial for real-time applications and the efficiency can not be guaranteed if collision tests are widely applied. A previous phase in these techniques should be able to restrict the number of collision tests by carrying out objects pairs selection, in order to compute intersection tests only in those pairs of objects sufficiently close to intersect. These non-discarded pairs are the input to a second phase in which some more concrete interaction aspects are studied. Then, if the contact has been finally detected, static collision tests provide information about the intersection area, possible objects deformations or changes in the trajectory of the mobile objects.

WSCG POSTERS proceedings
WSCG'2004, February 2-6, 2004, Plzen, Czech Republic.
Copyright UNION Agency-Science Press

This pruning phase mentioned above can be computed according to different time and space bounding strategies. It is well known how Voronoi diagrams [Oka92a] can be useful for techniques based on distance computation. Collision detection takes advantage of Voronoi regions characteristics because determining the minimum Euclidean distance between two objects is considered a spatio-temporal technique. The new approach we present in this paper uses a strategy based on tessellation, similar to the Voronoi diagram. Our aim is to provide an efficient pruning method for $2D$ simplified scenes, not based on the minimum Euclidean distance, but using the smallest polar angle criterion. The result is a new partition of the plane called *polar diagram.*

2. THE POLAR DIAGRAM

We first introduce the polar angle of the point $p = (x,y)$ with respect to s_i, denoted as $ang_{s_i}(p) < \pi$, as the angle formed by the positive horizontal line of p and the straight line joining p and s_i (see Figure 1).

Given a set S of n objects in the plane, the locus of points with smaller positive polar angle with respect

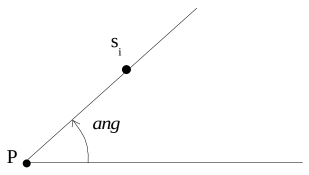

Figure 1: Polar angle.

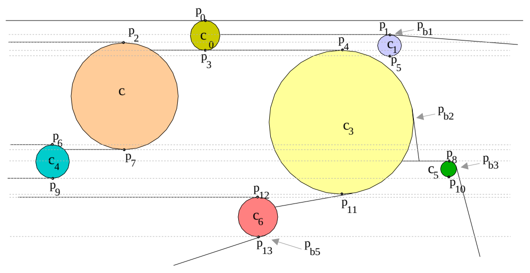

Figure 2: Example of circles polar diagram.

to $s_i \in S$ than with respect to any other $s_j \in S$ is called *polar region* of s_i, denoted $\mathcal{P}_S(s_i)$. Thus, $\mathcal{P}_S(s_i) = \{p = (x,y) \in E^2 \mid ang_{s_i}(p) < ang_{s_j}(p), \ \forall j \neq i\}$, being $ang_{s_i}(p)$ the angle described in Figure 1. The plane is divided in different regions in such a way that if the point $p \in E^2$ lies whithin $\mathcal{P}_S(s_i)$, it is known that s_i is the first object found after performing an angular scanning starting from p in the way that polar angle is defined. The union of all these n regions defines a tessellation we have called *polar diagram* of S, denoted as $\mathcal{P}(S)$ [Gri98a, Gri99b].

Polar diagram of circles

The polar diagram of a set of circles has similar features to any other geometric objects. We focus our attention on circles due to its election for the collision problem.

For the polar diagram construction of a set of circles, we create a set of $2n-1$ horizontal strips by throwing horizontal infinite lines in all North and South poles. The set of $2n$ poles are sorted obtaining the sequence of points $\{p_0, \ldots, p_{2n-1}\}$ (observe Figure 2). Every strip f_i is the locus of points lying within the horizontal band given by $[p_i, p_{i+1})$. A strip contains only one pole, the North or the South pole of a circle. As much, the number of polar edges processed in each strip is two, a horizontal and an oblique edge. There is always a horizontal one starting from North poles and another one starting from the South if an obstacle is found to the right. Oblique edges exist only in the right portion of the right most circle in a band, fact that reduces a lot the edges search. The strip computation time is $O(\log n)$ time what implies that the polar diagram of a set of n circles in the plane can be computed in $\Theta(n \log n)$ time ([Gri99b]).

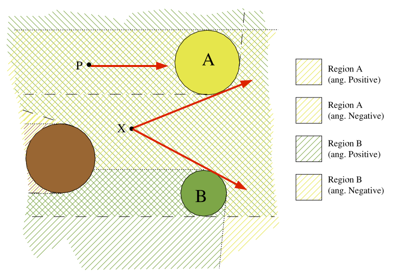

Figure 3: Point x belongs to A and B polar regions.

3. Visibility information

Visibility problems are some of the most important topics in Computational Geometry. One of these problems consists on finding the maximum visibility angle in an orthogonal direction that is easily computed in linear time by performing angular sweeps using clockwise and counter-clockwise criteria (observe Figure 3). However when this calculation is repetitive, it should be desirable to avoid all these exhaustive searches in order to obtain improved computation times. In fact, polar diagrams are known to be able to avoid these angular sweeps by locating a point into a polar region. It is easy to see that a positive angular scanning can be avoided in order to find object A, because point x lies whithin A polar region. In addition, it suffices to change the polar diagram criterion of construction to compute a new plane tessellation with different angle characteristics. In the example, it is possible to find object B using the negative polar angle criterion.

In Figure 3 it has been superimposed the two East polar diagrams according to the positive and negative angle criteria. The visibility problem solution is given by a simple result:

- when a point lies whithin regions associated to different objects, it always implies that there is an open visibility angle in the chosen polar diagram direction.
- when a point lies whithin regions belonging to the same object, the visibility angle is null.

Thus, as point x lies whithin polar regions belonging to objects A and B, it is known that there is an open visibility angle to the right. Nevertheless, point p only belongs to polar regions of the object A, what implies that the visibility angle is null. Anyway, this information can be enormously important because we do know

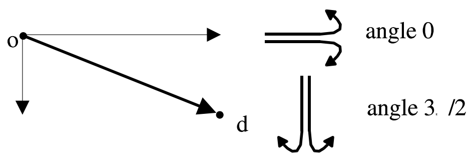

Figure 4: Orthogonal vector decomposition.

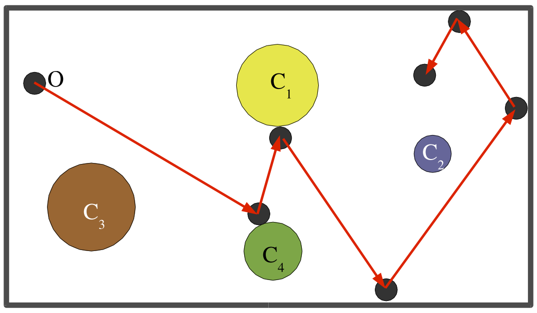

Figure 5: Example of trajectory.

the only object obstructing a trajectory in the specified direction.

The question now is whether polar diagrams are able to provide information about non-orthogonal directions. In real problems, a mobile object uses to move according to natural trajectories, what implies a generalized visibility problem resolution. In Figure 4 it is represented a $\overrightarrow{od}$ vector showing a possible trajectory from a start point o towards a goal position d. The decomposition of this vector gives two orthogonal vectors in East and South directions. As we study next, there is visibility information enough using these pairs of polar diagrams about any southeasterly direction.

4. COLLISION DETECTION

Visibility problems resolution can be considered the key to solve some other classical problems in Computer Graphics. Once we are able to check for the presence of obstacles using the visibility information, we can avoid any intersection with them as Motion or Path Planning techniques do, or just analysing the collision response if the contact is produced. In Collision Detection, polar diagrams are going to be able to recognize obstacle-free regions. This capability becomes useful to anticipate the object where the collision is going to take place with. Polar diagrams can help in the pruning phase, providing exactly the pair of object that are going to come into contact.

We consider a $2D$ scene consisting of a set of n circular objects as static obstacles, $C = \{c_0, c_2 \ldots, c_{n-1}\}$, where the mobile object O is considered a circular object as well. Circles have been chosen as dynamic

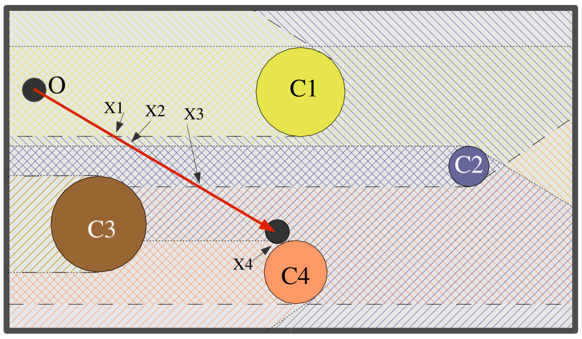

Figure 6: East polar diagrams.

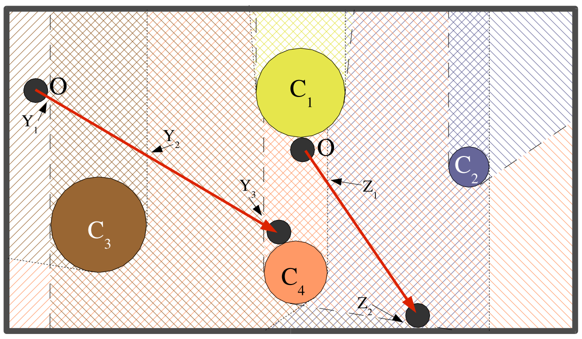

Figure 7: South polar diagrams.

and static objects due to the simplicity of the collision response. Furthermore in $3D$ scenes, complex objects are replaced with boxes or spheres in a pruning phase. The goal is always to speed up this previous phase of collision detection, therefore many simplification methods consist in doing projections to a $2D$ scene where we finally find circles and polygonal objects [Ben79a].

We observe now Figure 5, where an example of collided trajectory is illustrated. We suppose that the mobile object O is moving in a $2D$ scene inside a bounding box where O can rebound. All these suppositions are not crucial for the pruning strategy proposed in this work. The straight lines in the figure ends in circular obstacles or in the boundary of the scene, representing the trajectory vectors. Each of these lines symbolizes a visibility problem that is solved using polar diagrams.

Figures 6 and 7 illustrate the polar diagrams in East and South directions for this example. Once polar diagrams are calculated in optimal time $O(n \log n)$, the start point O is located into the East or the South pairs of diagrams in optimal logarithmic time. Any of them can be used, the election depends on a heuristic criterion.

The first solution is going to be given by using the East pairs of polar diagrams, although the same result

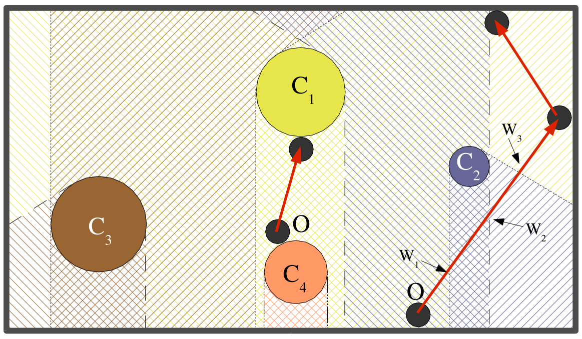

Figure 8: North polar diagrams.

is obtained with the South ones. Point O is located into C_1 polar regions in optimal time (see Figure 6). The described method proposed in Section 3 for Visibility problems resolution, shows that whether point O lies whithin polar regions of the same object, it implies that object C_1 is the first obstacle found if we move towards the East. As a consequence, it is known that the rectangle formed by the mobile object, the circle C_1 and the boundary line where x_1 lies, is obstacle-free. Consequently, point x_1 is known to be reached with no collision found.

One strength of polar diagrams is that polar edges can maintain information about adjacent regions with no additional computation time. Thus, new location operations are not necessary to be processed when the mobile point reaches a polar region boundary. Every time the trajectory vector crosses a frontier, one of the two polar regions changes, providing the mobile object with new visibility information. The process iterates from region to region until an obstacle or the bounding box is reached. In the example, point O belongs to C_1 and C_2 polar regions after crossing point x_1, what implies that no object can be found into the horizontal $[x_1, x_2]$ horizontal band.

The following portion of trajectory goes from x_3 towards x_4, being C_2 the only obstacle that could block the object O, however it is easily discarded because of the distance between them. In the new piece of trajectory, obstacle C_4 is not reached until point x_4 is crossed. Once there, it is known that O lies whithin polar regions of C_4, and according to how close they are, an intersection is known to be occurred.

The trajectory described above has been calculated using only the pair of East polar diagrams. The same result is obtained using the South pairs as it is illustrated in Figure 7. The start position where object O lies is a special case, differing of any other studied at the moment because it belongs to only one polar region. This circumstance automatically involves that object O is outside the minimum bounding box of the set of obstacles. Any movement towards point y_1 or any other point on this boundary line, is collision free. The rest of trajectory portions are obtained in a similar way to those using the East pairs of polar diagrams. In fact, no other possible circumstances can be found, making this method to gain in simplicity. The efficiency of this approach depends on the number of polar regions crossed in every straight trajectory. Consequently, the heuristic method to decide whether the horizontal or vertical polar diagrams is more efficient, can become crucial.

The location of an object into a region needs a logarithmic time, however when successive trajectory vectors have the same orthogonal component, new locations can be avoided (see Figure 8). This helps to avoid location operations and increase the collision detection method.

REFERENCES

[Ben79a] J.L Bentley, T.A. Ottmann. Algorithms for reporting and counting geometric intersections. *IEEE Transactions on Computing*, 28(9), 1979.

[Gri98a] C.I. Grima, A. Márquez and L. Ortega. A locus approach to angle problems in computational geometry. *14th European Workshop in Computational Geometry*, Barcelona, Spain. 1998.

[Gri99b] C.I. Grima, A. Márquez and L. Ortega. Polar diagrams of geometric objects. *15th European Workshop in Computational Geometry*, Antibes France, 1999.

[Jim01a] P. Jiménez, F. Thomas, C. Torras. 3D collision detection: a survey, *Computer & Graphics* 25, 2001.

[Lin98a] M.C. Lin, S. Gottschalk. Collision detection between geometric models: a survey, *IMA Conference on Mathematics of Surfaces*, 1998.

[Oka92a] A. Okabe, B. Boots y K. Sugihara. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams.* John Wiley and Sons, 1992.

[Osu01a] C. O'Sullivan, J. Dingliana, F. Ganovelli, G. Bradshaw, *T6:* Collision Handling for Virtual Environments. *Eurographics 2001 Turorial Proceedings*, 2001.