Computational Geometry
Lectures 1 and 2

Swati Gupta [*]

August 31, 2010

1 Navigating in a geometric environment

Example 1.1 *Consider you are in a room. You have the map of the room, which looks like figure(1). Now, you need to travel between the points A and B as shown in the figure. Which path must you follow so as not to hit the obstacles?*

This problem is called motion planning problem and is of crucial importance in the field of robotics. The number of solutions of this problem cannot be bounded by 2^n, since every slight perturbation of a feasible path may give a new path. This problem can also not be solved using djisktra's algorithm, for we have a map and not a graph. One can discretize the problem by imposing a grid on the map and the using manhattan distance. This gives an approximate solution only. We will try to stretch a band around the obstacles and try to obtain the shortest path.

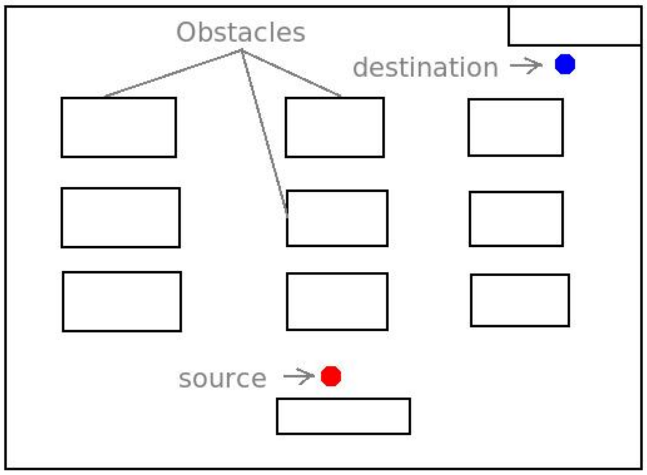

Figure 1: Map of a room

For simplicity, let us assume that the obstacles we are dealing with have straight boundaries(and

*Department of Computer Science and Engineering, IIT Delhi, New Delhi 1100116, India. E-mail:cs5060451@cse.iitd.ernet.in

not curves) and that they are in the form of rectangles. Also, we will assume that our problem is 2-dimensional, though 3-dimensional version can also be solved similarly. We can now make the following observations -

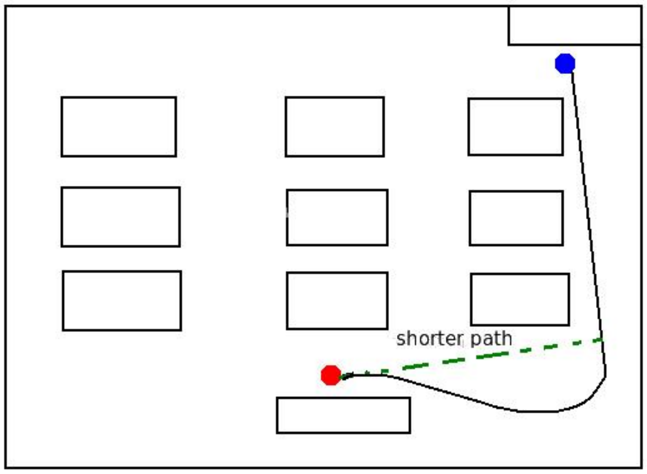

Figure 2: Piecewise linearity

Observation 1.1 *The shortest path taken by the object to go from point A to B must be piecewise linear. This is because we can otherwise always short cut the path, as shown in figure(2), and reduce the length of the path.*

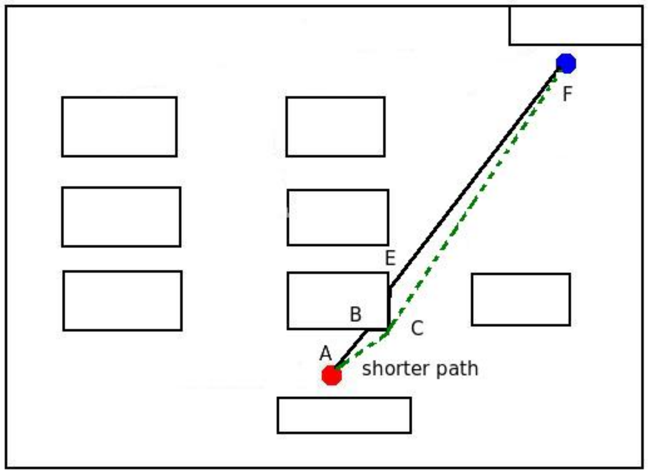

Figure 3: Bending at the Boundaries

Observation 1.2 *Our path will bend at the obstacles only. If it bends elsewhere other than obstacles, then the path can be shortened again. Suppose it bends at the boundary points (and not the corner points) then we can reduce the path length, as shown in figure (3). In the figure, if we join the corner point (which is the point C) with the last (bend) point on the path(which is the source itself, point A), then we get a shorter path. Thus, the shortest path must bend only at the corners of the obstacles.*

So, our algorithm to compute the shortest distance becomes -

1. Consider the corner points of each obstacle and add two more points for the required source and destination locations. Taking these points as the vertices, we will build a graph.
2. Out of the $\binom{4n}{2}$ potential edges between these vertices, include the edges in the graph which do not pass through the obstacles. The number of edges will be $O(n^2)$ and the weight on the edges will be the euclidean distance between the corresponding points on the given map.
3. Run djikstra's shortest path algorithm on the obtained graph. This graph is often called the visiblity graph.

Thus reducing the navigation problem to a graph instance gives us the shortest path between the given source and destination points.
In case the obstacles have curved boundaries, the shortest path may not bend only at the corner points. We may have to consider geodesic shortest paths. These problems have to be handled using parametric equations and tangential paths.

2 Art-Gallery Problem

The Art-Gallery problem is one of the most famous visiblity problems in computational geometry. Consider an art gallery represented by a simple polygon. A simple polygon is defined as a closed area on a plane that has a boundary composed of non-intersecting line segments. Assume there are no holes in the gallery, i.e. the interior of the gallery and its exterior are connected components. The walls of the art gallery are made up of n straight line segments which are not necessarily orthogonal. The art gallery problem is to place the minimum number of guards in the gallery, such that every point on the boundary of the gallery is visible to atleast one guard. We will assume that guards have a viewport of 360 degrees. Also, they can see everything in their field of view, that is, unless obstructed by a boundary. Guards will be represented by (stationary) points in the region enclosed by the polygon.

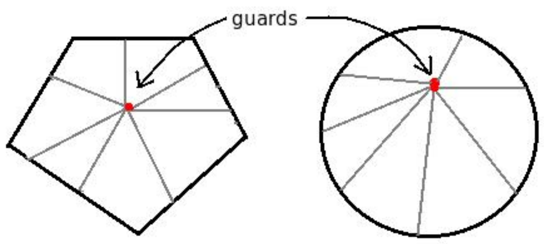

Figure 4: Example 1

Example 2.1 *Consider a circular art gallery, or even a pentagonal art gallery. These galleries require only one guard. In fact, every convex shaped art gallery requires only one guard. Line joining any two points in a convex polygon lies completely inside the polygon only and does not cross the boundary. Refer to figure (4).*

Example 2.2 *Now, consider a gallery as shown in figure(5). The blue-shaded region is the region that contains all points from where point P will be visible. The red-shaded region contains all points from where the point Q will be visible. As these regions do not interesect, we can argue that atleast 2 guards will be required for this problem. Placement of guards given by points R and S gives a possible solution using two guards. Thus, we have argued that two guards are necessary and sufficient.*

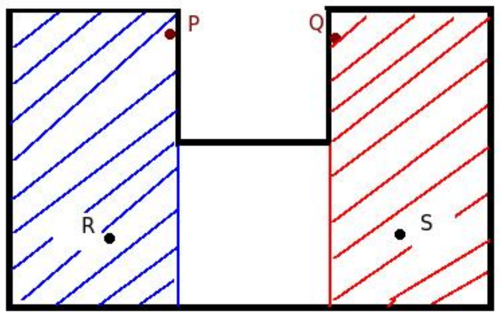

Figure 5: Example 2

Example 2.3 *Let us now consider an art gallery of the form as in figure(6). The gallery can be viewed as being formed using two triangles. The interesection of the two triangles here is called the kernel of the polygon. Kernel is defined as............ From the kernel, every point on the polygon is visible. A star-shaped polygon is a polygon which has a non-trivial kernel. One guard suffices for every star-shaped polygon.*

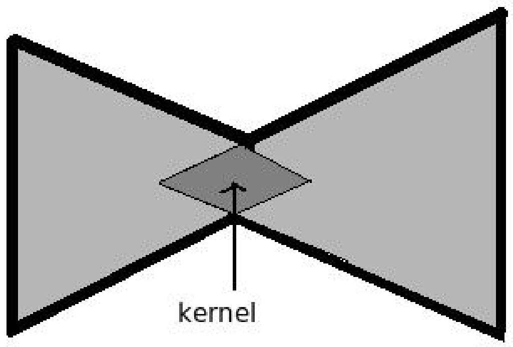

Figure 6: Example 3

Goal: Given an arbitrary simple polygon, find (algorithmically) the minimum number of guards.

This problem is an NP-hard problem. There is no known polynomial time algorithm that finds the minimum number of guards. For a simple polygon with n-vertices, what can we say about the -

1. Minimum number of guards required in the worst case.
2. Maximum number of guards required in the worst case.

The answer for the first question is 1. For the second, we have the following theorem -

Art-Gallery Theorem (Chavatal): If there are n vertices of a polygon (or equivalently edges), then $\lfloor n/3 \rfloor$ guards are necessary (for some instances) and sufficient(for all instances).

Proof of Art-Gallery Theorem

Observation 2.1 *For any gallery with n vertices, n guards will suffice.*

The above obervation holds because placing a guard in each triangle will cover the entire polygon.

To prove the Art-Gallery theorem, we will first triangulate the representative polygon, and then color each vertex of the triangles with either Red, Blue or Green. We will prove that a valid coloring of the vertices (such that no edge is incident on two vertices with the same color) exists, and placing a guard on vertices with a particular color gives us the required result.

Lemma 2.1 *One can always find a diagonal in a simple polygon.*

Proof To find a diagonal, consider the left most point of the polygon say L, and the two points immediately adjacent to it, say A and B (refer to figure (7)). If the line joining AB does not intersect the boundary then we are done. Otherwise translate a line parallel to AB toward L and record the last vertex encountered on the line. Join this vertex with L to obtain a diagonal.

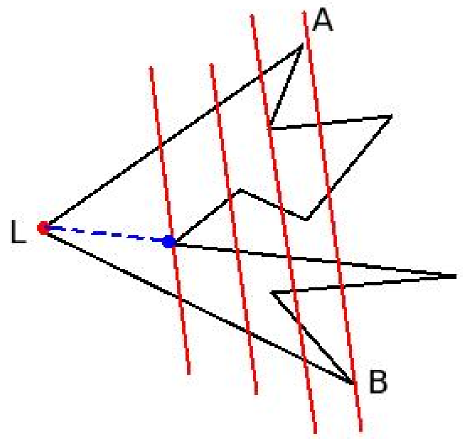

Figure 7: Finding a diagonal

Theorem 2.1 *Every simple polygon can be triangulated.*

Proof From lemma (2.1), we can always find a diagonal in a polygon such that it does not intersect any boundary. Adding the diagonal to the polygon, we get two smaller polygons. Repeating the procedure on the two smaller polygons, we can triangulate the entire polygon.

Lemma 2.2 *There exists a valid coloring for the vertices of a triangulation, such that no two adjacent vertices have the same color.*

Proof To find a 3-coloring, it is helpful to observe that the dual graph to the triangulation (the undirected graph having one vertex per triangle and one edge per pair of adjacent triangles) is a tree, for any cycle in the dual graph would form the boundary of a hole in the polygon, contrary to the assumption that it has no holes. Whenever there is more than one triangle, the dual graph (like any tree) must have a vertex with one neighbor, corresponding to a triangle that is adjacent to other triangles along only one of its sides. The simpler polygon formed by removing this triangle has a 3-coloring by mathematical induction, and this coloring is easily extended to the one additional vertex of the removed triangle. (Reference: Wikipedia)

Theorem 2.2 *$\lfloor n/3 \rfloor$ guards are sufficient for all simple polygons.*

Proof To prove the above, we can triangulate the polygon and color the vertices of the triangles using three colors, say Red, Green and Blue. Let the number of vertices colored Red be denoted by r, number of Blue be denoted by b, and number of vertices colored green by g. Then,

$$r + g + b = n$$

By pigeon hole principle, at least one color(say Red) will occur $\leq \lfloor n/3 \rfloor$ times. Placing guards at vertices colored Red will ensure that every point on the boundary is visible to atleast one guard, as all triangles will have exactly one vertex colored Red.

Theorem 2.3 *$\lfloor n/3 \rfloor$ guards are necessary for some cases of simple polygons.*

Example 2.4 *As an example, a comb structure as shown in the figure(8) requires $\lfloor n/3 \rfloor$ guards. Here, we have 12 sides and require a guard in each of the shaded regions, as in example(2.2).*

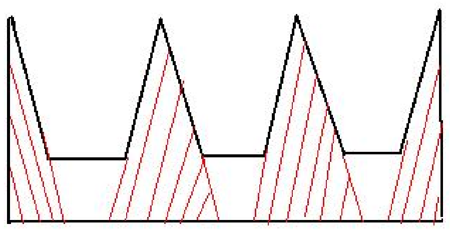

Figure 8: Example 4