24 SHORTEST PATHS AND NETWORKS

Joseph S. B. Mitchell

24.1 INTRODUCTION

Computing an optimal path in a geometric domain is a fundamental problem in computational geometry, with applications in robotics, geographic information systems (GIS), wire routing, etc.

A taxonomy of shortest-path problems arises from several parameters that define the problem:

1. Objective function (metric for "length").
2. Constraints on the path ("get from s to t," visit a set of points/regions, etc).
3. Input geometry (the "map"); e.g., types of "obstacles" allowed.
4. Type of moving object (single point vs. more complex robot geometry).
5. Dimension of the problem (2, 3, or higher).
6. Single shot vs. repetitive mode queries.
7. Static vs. dynamic environments (e.g., allowing insertions/deletions of obstacles, or allow moving obstacles).
8. Exact vs. approximate algorithms.
9. Known vs. unknown map; e.g., discovery of obstacles on-line, using a line-of-sight sensor.

We survey various forms of the problem, primarily in two dimensions, for motion of a single point, since most results have focussed on these cases. We discuss shortest paths in a simple polygon (Section 24.1), shortest paths among obstacles (Section 24.2), other metrics for length (Section 24.3), other path/network optimization problems (Section 24.4), and higher dimensions (Section 24.5).

GLOSSARY

Path: A continuous image of an interval.

Polygonal s-t path: A path from point s to point t consisting of a finite number of line segments (***edges***, or ***links***) joining a sequence of points (***vertices***).

Start s, goal t: Two points for which one wants to compute an interconnecting path.

Length of a path: A nonnegative number associated with a path, measuring its total cost according to some prescribed metric. Unless otherwise specified, the length will be the Euclidean length of the path.

Shortest/optimal/geodesic path: A path of minimum length among all paths that are feasible (satisfying all imposed constraints).

Shortest-path distance: The metric induced by a shortest-path problem. The shortest-path distance between s and t is the length of a shortest s-t path; in many geometric contexts, it is also referred to as ***geodesic distance***.

Locally shortest/optimal path: A path that cannot be improved by making a small change to it that preserves its ***combinatorial structure*** (e.g., the ordered sequence of triangles visited, for some triangulation of a polygonal domain P); also known as a ***taut-string*** path in the case of a shortest obstacle-avoiding path.

Simple polygon P of n vertices: A closed, simply-connected region whose boundary is a union of n (straight) line segments (edges), whose endpoints are the vertices of P.

Polygonal domain P of n vertices and h holes: A closed, multiply-connected region whose boundary is a union of n line segments, forming $h+1$ closed (polygonal) cycles. A simple polygon is a polygonal domain with $h = 0$.

Triangulation of a simple polygon P: A decomposition of P into triangles such that any two triangles either intersect in a common vertex, a common edge, or not at all. A triangulation of P can be computed in $O(n)$ time [33].

Obstacle: A region of space whose interior is forbidden to paths. The complement of the set of obstacles is the ***free space***. If the free space is a polygonal domain P, the obstacles are the $h+1$ connected components (h ***holes***, plus the ***face at infinity***) of the complement of P.

Visibility graph VG(P): A graph whose nodes are the vertices of P and whose edges join pairs of nodes for which the corresponding segment lies inside P. An example is shown in Figure 24.2.

Shortest path tree, SPT(s,P): The union of shortest paths from s to every vertex of P. (it is unique in simple polygon)

Single-source query: A query that specifies a goal point t, and requests the length of a shortest path from a *fixed* source point s to t. The query may also require the retrieval of an actual instance of a shortest s-t path; in general, this can be reported in additional time $O(k)$, where k is the complexity of the output (e.g., number of edges).

Shortest Path Map, SPM(s): A decomposition of free space into regions (***cells***) according to the "combinatorial structure" of shortest paths from a fixed source point s to points in the regions. Specifically, for shortest paths in a polygonal domain, SPM(s) is a decomposition of P into cells such that for all points t interior to a cell, the sequence of obstacle vertices along an s-t path is fixed. In particular, the *last* obstacle vertex along a shortest s-t path is the ***root*** of the cell containing t. Each cell is ***star-shaped*** with respect to its root, which lies on the boundary of the cell. See Figure 24.1, where the root of the cell containing t is labeled r.

If SPM(s) is preprocessed for point location, then single-source queries can be answered efficiently by locating the query point t within the decomposition.

Two-point query: A query that specifies two points, s and t, and requests the length of a shortest path between them. In all cases discussed here, an actual instance of a shortest path can be reported in additional time $O(k)$, where k is the complexity of the output (e.g., number of edges).

Geodesic Voronoi diagram (VD): A Voronoi diagram for a set of ***sites***, in which the underlying metric is the geodesic distance.

Geodesic center of P: A point within P that minimizes the maximum of the shortest-path lengths to any other point in P.

Geodesic diameter of P: The length of a longest shortest path between a pair of vertices of P.

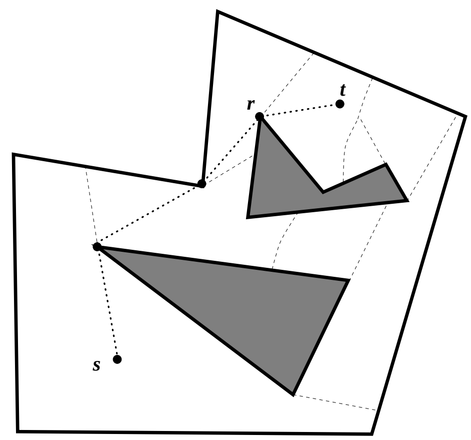

Figure 24.1
A shortest path map with respect to source point s within a polygonal domain. The heavy dotted path indicates the shortest s-t path, which reaches t via the root r of its cell.

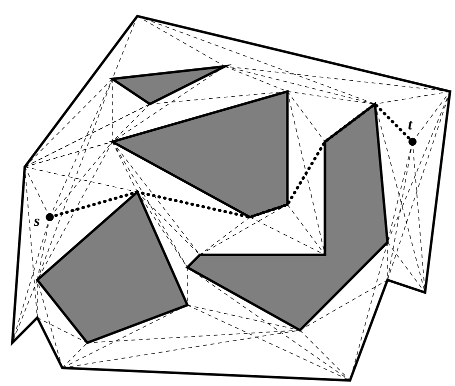

Figure 24.2
The visibility graph VG(P): Edges of VG(P) are of two types – (1) the heavy dark boundary edges of P, and (2) the edges that intersect the interior of P, shown with thin dashed segments. A shortest s-t path is highlighted.

24.2 PATHS IN A SIMPLE POLYGON

The most basic geometric shortest-path problem is to find a shortest path, inside a ***simple*** polygon, P (having no "holes"), connecting two points, s and t. The complement of P serves as an "obstacle" through which the path is not allowed to travel.

In this case, there is a unique taut-string path from s to t, since there is only one way to "thread" a string through a simply-connected region.

Algorithms for computing a shortest s-t path begin with a triangulation of P ($O(n)$ time; [33]), whose dual graph is a tree. The "sleeve" is comprised of the triangles that correspond to the (unique) path in the dual that joins the triangle containing s to that containing t. By considering the effect of adding the triangles in order along the sleeve, it is not hard to obtain an $O(n)$ time algorithm for collapsing the sleeve into a shortest path [32, 94]. At a generic step of the algorithm, the sleeve has been collapsed to a structure called a "funnel" (with "base" ab and "root" r) consisting of the shortest path from s to a vertex r, and two (concave) shortest paths joining r to the endpoints of the segment ab that bounds the triangle abc that is about to be considered (see Figure 24.3). In adding triangle abc, we "split" the funnel in two according to the taut-string path from r to c, which will, in general, include a segment, uc, joining c to some (vertex) point of tangency, u, along one of the two concave chains of the funnel. After the split, we keep that funnel (with base ac or bc) that contains the s-t taut-string path. The work needed to search for u can easily be charged off to those vertices that are discarded from further consideration. The end result is that a shortest s-t path is found in time $O(n)$, which is worst-case optimal.

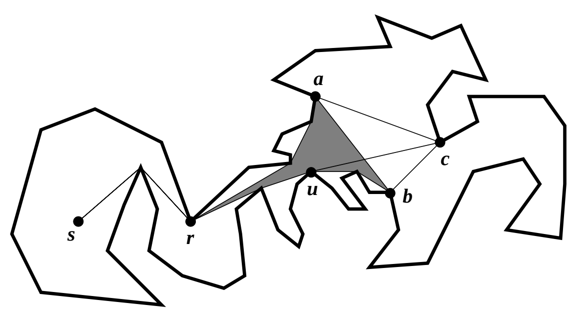

Figure 24.3
Splitting a funnel.

SHORTEST PATH MAPS

The shortest path map SPM(s) for a simple polygon has a particularly simple structure, as the boundaries between cells in the map are simply (line segment) chords of P obtained by extending appropriate edges of the visibility graph $VG(P)$. It can be computed in time $O(n)$ by using somewhat more sophisticated data structures to do funnel splitting efficiently (since, in this case, we cannot discard one side of each split funnel). Single-source queries can be answered in $O(\log n)$ time, after storing the SPM(s) in an appropriate $O(n)$-size point location data structure.

TWO-POINT QUERIES

A simple polygon can be preprocessed in time $O(n)$, into a data structure of size $O(n)$, to support shortest-path queries between any two points $s, t \in P$. In time $O(\log n)$ the length of the shortest path can be reported, and in additional time $O(k)$, the shortest path can be reported, where k is the number of vertices in the output path.

DYNAMIC VERSION

In the dynamic version of the problem, one allows the polygon P to change, with the addition/deletion of edges and vertices. If the changes are always made in such a way that the set of all edges yields a ***connected planar subdivision*** of the plane into simple polygons (i.e., no "islands" are created), then one can maintain a data structure of size $O(n)$ that supports two-point query time of $O(\log^2 n)$ (plus $O(k)$ if the path is to be reported), and update time of $O(\log^2 n)$ for each addition/deletion of an edge/vertex [69]. (The result of [69] improves the first results on the dynamic problem, obtained by Chiang, Preparata, and Tamassia [40, 41], who gave a data structure achieving $O(\log^3 n)$ query and update bounds, using $O(n \log n)$ space. The same data structure also gives the best known dynamic point location solution for connected maps, with optimal $O(\log n)$ query time.)

OTHER RESULTS

Several other problems have been studied with respect to geodesic distances induced by a simple polygon and are summarized in Table 24.1.

Shortest paths within simple polygons give a wealth of structural information about the polygon. In particular, they have been used to give an output-sensitive algorithm for constructing the visibility graph of a simple polygon ([75]) and can be used for constructing a ***geodesic triangulation*** of a simple polygon, which allows for efficient ray-shooting (see [34, 82]). They also form a crucial step in solving ***link distance*** problems, as we will discuss later.

OPEN PROBLEMS

1. Can one devise a simple $O(n)$ time algorithm for computing the shortest path between two points in a simple polygon, *without* resorting to a (complicated) linear-time triangulation algorithm?
2. Can the geodesic Voronoi diagram for k sites within P be computed in time $O(n + k \log k)$?

TABLE 24.1 Shortest paths and geodesic distance in simple polygons.

PROBLEM VERSION	COMPLEXITY	NOTES	SOURCE
Shortest s-t path	$O(n)$		[32, 71, 79, 94]
Single-source query; SPM(s)	$O(\log n)$ query $O(n)$ preproc./space	builds SPM(s) in $O(n)$ time	[71, 79]
Two-point query	$O(\log n)$ query $O(n)$ preproc./space		[70, 76]
Two-polygon query	$O(\log k + \log n)$ query $O(n)$ space	between convex k-gons in simple n-gon	[42]
Dynamic two-point query	$O(\log^2 n)$ update/query $O(n)$ space		[69]
Dynamic two-polygon query	$O(\log k + \log^2 n)$ query $O(\log^2 n)$ update $O(n)$ space	between convex k-gons in simple n-gon	[42]
Parallel algorithm (CREW PRAM)	$O(\log n)$ time $O(n/\log n)$ processors	in triangulated polygon also builds SPM(s)	[77]
Geodesic Voronoi diagram (VD)	$O((n+k)\log(n+k)))$	k point sites	[122]
All nearest neighbors	$O(n)$	for set of vertices	[81]
Geodesic furthest-site VD	$O((n+k)\log(n+k))$ time $O(n+k)$ space	k point sites	[15]
All furthest neighbors	$O(n)$	for set of vertices	[81]
Geodesic diameter	$O(n)$		[81]
Geodesic center	$O(n\log^2 n)$		[124]

3. Can the geodesic center of a simple polygon be computed in $O(n)$ time? What can be said in the ***bit complexity*** (vs. real RAM) model of computation?

24.3 PATHS IN A POLYGONAL DOMAIN

While in a simple polygon there is a unique taut-string path between two points, in a general polygonal domain P, there can be an exponential number of taut-string simple paths between two points.

The homotopy type of a path can be expressed as a sequence (with repetitions) of triangles visited, for some triangulation of P. For any given homotopy type, expressed with N triangles, a shortest (taut) path of that type can be computed in $O(N)$ time [79].

SEARCHING THE VISIBILITY GRAPH

Without loss of generality, we can assume that s and t are vertices of P (since we can make "point" holes in P at s and t). It is easy to show that any locally optimal s-t path must lie on the visibility graph $VG(P)$ (Figure 24.2). We can construct $VG(P)$ in output-sensitive time $O(E_{VG}+n\log n)$, where E_{VG} denotes the number of edges of $VG(P)$ [68], even if we allow only $O(n)$ working space [123, 127]. Given the graph $VG(P)$, whose edges are weighted by their Euclidean lengths, we can use Dijsktra's algorithm to construct a tree of shortest paths from s to all vertices of P, in time $O(E_{VG}+n\log n)$ [64, 60]. Thus, Euclidean shortest paths among obstacles in the plane can be computed in time $O(E_{VG}+n\log n)$. This bound is worst-case quadratic in n, since $E_{VG} \leq \binom{n}{2}$; note too that domains exist with $E_{VG} = \Omega(n^2)$.

Given the tree of shortest paths from s, we can compute SPM(s) in time $O(n \log n)$ [100].

CONTINUOUS DIJKSTRA METHOD

Instead of searching the visibility graph (which may have quadratic size), an alternative paradigm for shortest-path problems is to construct the (linear-size) shortest path map directly. The ***continuous Dijkstra*** method [97, 106, 107, 100, 101, 98, 99] was developed for this purpose.

Building on the success of the method in solving (in nearly linear time) the shortest-path problem for the L_1 metric, Mitchell [102, 104] developed a version of the continuous Dijkstra method applicable to the Euclidean shortest-path problem, obtaining the first subquadratic ($O(n^{1.5+\epsilon})$) time bound. Subsequently, this result was improved by Hershberger and Suri [83, 84], who achieve a nearly optimal algorithm based also on the continuous Dijkstra method. They give an $O(n \log n)$ time and $O(n \log n)$ space algorithm, coming close to the lower bounds of $\Omega(n + h \log h)$ time and $O(n)$ space.

The continuous Dijkstra paradigm involves simulating the effect of a "wavefront" propagating out from the source point, s. The ***wavefront*** at distance δ from s is the set of all points of P that are at geodesic distance δ from s. It consists of a set of curve pieces, called ***wavelets***, which are arcs of circles, centered at obstacle vertices that have already been reached. At certain critical "events," the structure of the wavefront changes due to one of the following possibilities:

(1) a wavelet disappears (due to the "closure" of a cell of the SPM); or

(2) a wavelet collides with an obstacle vertex; or

(3) a wavelet collides with another wavelet; or

(4) a wavelet collides with an obstacle edge at a point interior to that edge.

It is not difficult to see from the fact that SPM(s) has linear size that the total number of such events is $O(n)$. The challenge in applying this propagation scheme is in devising an efficient method to know *what* events are going to occur and in being able to *process* each event as it occurs (updating the combinatorial structure of the wavefront).

One approach, used in [102, 104], is to track a "pseudo-wavefront," which is allowed to run over itself, and "clip" only when a wavelet collides with a vertex that has already been labeled due to an earlier event. Detection of when a wavelet collides with a vertex is accomplished with range searching techniques. An alternative approach, used in [83, 84], simplifies the problem by first decomposing the domain P using a "conforming subdivision," which allows one to propagate an "approximate wavefront" on a *cell-by-cell* basis. A key property of a conforming subdivision is that for any edge (of length L) of the subdivision, there are only a *constant* number of (constant-sized) cells within geodesic distance L of it.

TABLE 24.2 Shortest paths among planar obstacles, in a polygonal domain.

PROBLEM	COMPLEXITY	NOTES	SOURCE
Shortest s-t path	$O(n \log n)$	$O(n \log n)$ space	[83, 84]
	$O(n + h^2 \log n)$	$O(n)$ space	[91, 103]
	$O(n^{1.5+\epsilon})$	$O(n)$ space	[102, 104]
Approx shortest s-t path	$O((n \log n)/\epsilon)$	$O(n/\epsilon)$ space	[51, 36]
	$O((n \log n)/\sqrt{\epsilon})$	$O(n/\sqrt{\epsilon})$ space	[101, 98]
SPM(s)/geodesic VD	$O(n \log n)$	$O(n \log n)$ space	[83, 84]
	$O(n^{1.5+\epsilon})$	$O(n)$ space	[102, 104]
Approx two-point query	$O(n \log n)$ query $O(n)$ space $O(n \log n)$ preproc.	$(1+\epsilon)$-approx	[51]
Approx two-point query	$O(\log n)$ query $O(n^2)$ space $O(n^2 \log n)$ preproc.	$(1+\epsilon)$-approx	[51, 36]
Approx two-point query	$O(\log n)$ query $O(n \log n)$ space $O(n^{3/2}/ \log^{1/2} n)$ preproc.	$(3\sqrt{2}+\epsilon)$-approx	[?]
Approx Two-point query	$O(\log n + \sqrt{r})$ query $O(n^2/\sqrt{r})$ space $O(n^2/\sqrt{r})$ preproc.	$(\sqrt{2}+\epsilon)$-approx $1 \leq r \leq n$	[?]
Approx Two-point query	$O(n)$ query $O(n)$ space $O(n \log n)$ preproc.	$(\sqrt{2}+\epsilon)$-approx	[?]
Approx Two-point query	$O(\log n)$ query $O(n^{3/2})$ space $O(n^{3/2})$ preproc.	$(2\sqrt{2}+\epsilon)$-approx	[?]

OTHER RESULTS

Table 24.2 summarizes the results discussed so far, and gives some approximation results, especially for the two-point query problem.

OPEN PROBLEMS

1. Can one solve the Euclidean shortest-path problem in $O(n+h \log h)$ time and $O(n)$ space?
2. How efficiently, and using what size data structure, can one preprocess a polygonal domain for exact two-point queries?
3. How efficiently can one compute a geodesic center/diameter for a polygonal domain?
4. How efficiently can one solve Euclidean problems (shortest paths, geodesic center/diameter) in the *bit complexity*, instead of the real RAM, model of computation?

24.4 OTHER METRICS FOR LENGTH

In the problems considered so far, the Euclidean metric has been used to measure the length of a path. We consider now several other possible objective functions for measuring path length. Tables 24.3 and 24.4 summarize results.

GLOSSARY

L_p Metric: The L_p distance between $q = (q_x, q_y)$ and $r = (r_x, r_y)$ is given by $d_p(q, r) = [|q_x - r_x|^p + |q_y - r_y|^p]^{1/p}$. The L_p length of a polygonal path is the sum of the L_p lengths of each edge of the path. Special cases of the L_p metric include the L_1 metric (***Manhattan metric***) and the L_∞ metric ($d_\infty(q, r) = \max\{|q_x - r_x|, |q_y - r_y|\}$).

Rectilinear path: A polygonal path with each edge parallel to a coordinate axis; also known as an ***isothetic*** path.

C-Oriented path: A polygonal path with each edge parallel to one of a set C of $c = |C|$ ***fixed orientations***.

Link distance: The minimum number of edges from s to t within a polygonal domain P. If the paths are restricted to be rectilinear or C-oriented, then we obtain the ***rectilinear link distance*** or ***C-oriented link distance***.

Min-link s-t path: A polygonal path from s to t that achieves the link distance.

Weighted region problem: Given a piecewise-constant function, $f : R^2 \rightarrow R$, that is defined by assigning a nonnegative ***weight*** to each face of a given triangulation in the plane. The ***weighted length*** of an s-t path π is the path integral, $\int_\pi f(x, y) d\sigma$, of the weight function along π. The ***weighted region metric*** associated with f defines the distance $d_f(s, t)$ to be the infimum over all s-t paths π of the weighted length of π. The ***weighted region problem*** (WRP) asks for an s-t path of minimum weighted length.

Sailor's problem: Compute a minimum-cost path, where the cost of motion is *direction-dependent*, and there is a cost L per turn (in a polygonal path).

Bounded curvature shortest-path problem: Compute a shortest obstacle-avoiding smooth (C^1) path joining point s, with prescribed velocity orientation, to point t, with prescribed velocity orientation, such that at each point of the path the radius of curvature is at least 1.

Maximum concealment path: A path within polygonal domain P that minimizes the length during which the robot is exposed to a given set of "enemy" observers. This problem is a special case of the weighted region problem, in which weights are 0 (for travel in concealed free space), 1 (for travel in exposed free space), or ∞ (for travel through obstacles).

Total turn for an s-t path: The sum of the absolute values of all turn angles for a polygonal s-t path.

Minimum-time path problem: Find a path to minimize the total time required to move from an initial position, at an initial velocity, to a goal position and velocity, subject to bounds on the allowed acceleration and velocity along the path. This problem is also known as the "kinodynamic motion planning problem."

LINK DISTANCE

In the min-link path problem, our goal is to minimize the number of links (and hence the number of turns) in a path connecting s and t. In many problems, the link distance provides a more natural measure of path complexity than the Euclidean length, as well as having applications to curve simplification. [72, 88, 111].

In a simple polygon P, a min-link path can be computed in time $O(n)$ [133, 135]. In fact, in time $O(n)$ a ***window partition*** of P with respect to a point s can be computed, after which a min-link path from s to t can be reported in time proportional to the link distance. The window partition of P with respect to s is a shortest path map, SPM(s), in the link distance metric: it partitions the interior of P into cells such that the link distance from s to points of a cell is constant. The points $V(s) \subseteq P$ that are visible to s constitute the cell for link distance 1. The cells at link distance 2 consist of the regions of $P \setminus V(s)$ that are visible from the "windows" of $V(s)$ (a ***window*** of $V(s)$ is an edge that forms a boundary between $V(s)$ and $P \setminus V(s)$). Continuing in this way, doing "staged illumination," we obtain the window partition of P into (polygonal) cells, each of which is fully illuminated by a window of an adjacent cell whose link distance is one less. Essentially, this procedure is a form of the continuous Dijkstra method, under the link distance metric.

In a polygonal domain with holes, min-link paths can also be computed using a staged illumination method, but the algorithm is not simple and it relies on efficient methods for computing a single face in an arrangement of line segments [7]. A min-link s-t path can be computed in time $O(E_{VG}\alpha^2(n)\log n)$, where $\alpha(n)$ is the extremely slowly growing inverse Ackermann function [109]. If we consider C-oriented and rectilinear link distance, then some better time/space bounds are possible, and some of these apply also to "combined metrics," in which there is a cost for length as well as links.

The vast body of results on link distance computation has assumed a real RAM model of computation. But, as has recently been pointed out by Kahan and Snoeyink [90], this assumption may be particularly misleading when calculating link distances. The problem is that the multiple stages of illumination that define the window partition of a polygon P leads to algorithms requiring the multiplication of $O(n)$ transformation matrices, causing the required precision to escalate. In particular, Kahan and Snoeyink show that even representing an optimal min-link path in a polygon whose n vertices occupy $\Theta(n \log n)$ bits can require $\Theta(n^2 \log n)$ bits to express the rational coordinates of the path vertices. Kahan and Snoeyink also give some positive results by showing that one can get close to the minimum number of links, under the assumption that the precision of path vertices is fixed. If path vertices are restricted to be "first derived points" (crossing points where an extension of a visibility graph edge intersects a polygon edge), then one can always find a path that uses at most 2 times the optimal number of links. If path vertices are restricted to be grid points (on an N-by-N grid), then the number of links can increase by a factor of $\Theta(\log N)$ times optimal.

Refer to table Table 24.3 for many related results on link distance, including rectilinear link distance, two-point queries, and link diameter and link center problems.

TABLE 24.3 Link distance shortest-path problems.

PROBLEM	COMPLEXITY	NOTES	SOURCE
Min-link path	$O(E_{VG}\alpha^2(n)\log n)$	polygonal domain	[109, 7]
Min-link path	$O(n)$	simple polygon	[133, 135]
Rectilinear link path	$O(n\log n)$ time, $O(n)$ space	rectilinear obstacles	[54]
Rectilinear link path	$O(n)$	rectilinear simple polygon	[55, 79]
C-Oriented link path	$O(c^2 n\log n)$ time $O(c^2 n\log n)$ space, preproc.	C-oriented obstacles builds SPM(s)	[1]
Two-point link query (exact distance)	$O(\log n)$ query $O(n^3)$ space, preproc.	simple polygon	[14]
Two-Point Link Query (approx distance)	$O(\log n)$ query $O(n^2)$ space	polygonal domain additive error (+4)	[105]
Two-Point Link Query (approx distance)	$O(\log n)$ query $O(n^2)$ space, preproc.	simple polygon additive error (+1)	[14] [14]
Two-Point Link Query (approx distance)	$O(\log n)$ query $O(n)$ space, preproc.	simple polygon additive error (±2)	[135]
Dynamic Two-Point Link Query	$O(L\log^2 n)$ query $O(\log^2 n)$ update $O(n)$ space	in connected polygonal map L = link dist	[42]
Two-point rectilinear link query	$O(\log n)$ query $O(n\log n)$ space, preproc.	rectilinear simple polygon also is L_1 opt	[55, 79]
Two-Polygon Link Query	$O(\log k+\log n)$ query $O(n^3)$ space	between convex k-gons in simple n-gon	[42]
Dynamic Two-Polygon Link Query	$O(\log k+L\log^2 n)$ query $O(\log^2 n)$ update $O(n)$ space	between convex k-gons in simple n-gon L = link dist	[42]
Link-diameter	$O(n\log n)$	simple polygon	[134, 92]
Approx Link-Diameter	$O(n)$	simple polygon, ±2 error	[135]
Rectilinear link diameter	$O(n)$	rectilinear simple polygon	[117]
Link-center	$O(n\log n)$	simple polygon	[59, 92]
Rectilinear link-center	$O(n)$	rectilinear simple polygon	[116]
Shortest k-link path	$O(n^3 k^3 \log(Nk/\epsilon^{1/k}))$	simple polygon	[108]

L_1 METRIC

Instead of measuring path length according to the L_2 (Euclidean) metric, consider the problem of computing shortest paths in a polygonal domain P that are short according to the L_1 metric.

A method based on visibility graph principles allows one to construct a sparse graph (with $O(n\log n)$ nodes and edges) that is ***path preserving*** in that it is guaranteed to contain a shortest path between any two vertices [52]. Applying Dijkstra's algorithm then gives an $O(n\log^{1.5} n)$ time ($O(n\log n)$ space) algorithm for L_1 shortest paths.

A method based on the continuous Dijkstra paradigm allows the SPM(s) to be constructed in time $O(n\log n)$, using $O(n)$ space [101, 98]. The special property of the L_1 metric that is exploited in this algorithm is the fact that the wavefront in this case is piecewise-linear, with "wavelets" that are line segments of slope ±1, so that the first vertex hit by a wavelet can be determined by rectangular range searching techniques.

Methods for finding L_1 shortest paths generalize to the case of C-oriented paths, in which $c = |C|$ fixed directions are given. Shortest C-oriented paths can be computed in time $O(cn\log n)$. Since the Euclidean metric is approximated to within accuracy $O(1/c^2)$ if we use c equally spaced orientations, this results in an

TABLE 24.4 Shortest paths in other metrics.

PROBLEM	COMPLEXITY	NOTES	SOURCE
L_1 Shortest path, SPM(s)	$O(n \log n)$	polygonal domain	[52, 101, 98]
L_1 Two-point query	$O(\log^2 n)$ query $O(n^2 \log n)$ space $O(n^2 \log^2 n)$ preproc.	polygonal domain	[35, ?]
L_1 Two-point query	$O(\log n)$ query $O(n^2)$ space, preproc.	rectangle obstacles	[20, 19, 62]
L_1 Two-point query	$O(\sqrt{n})$ query $O(n^{1.5})$ space, preproc.	rectangle obstacles	[62]
L_1 Approx two-point query	$O(\log n)$ query $O(n \log n)$ space $O(n \log^2 n)$ preproc.	3-approx rectangle obstacles	[37]
L_1 Approx two-point query	$O(\log n + \sqrt{r})$ query $O(n^2/\sqrt{r})$ space $O(n^2/\sqrt{r})$ preproc.	$(1+\epsilon)$-approx $1 \leq r \leq n$ polygonal domain	[?]
L_1 Approx two-point query	$O(n)$ query $O(n)$ space $O(n \log n)$ preproc.	$(1+\epsilon)$-approx polygonal domain	[?]
L_1 Approx two-point query	$O(\log n)$ query $O(n^{3/2})$ space $O(n^{3/2})$ preproc.	$(2+\epsilon)$-approx polygonal domain	[?]
L_1 Approx two-point query	$O(\log n)$ query $O(n \log n)$ space $O(n^{3/2}/\log^{1/2} n)$ preproc.	$(3+\epsilon)$-approx polygonal domain	[?]
Weighted region problem	$O(n^8 L)$ $L = O(\log \frac{nNW}{\epsilon})$	$(1+\epsilon)$-approx	[107]
Weighted region problem	$O(n^2)$	region weights 0, 1, ∞	[66]
L_1 Weighted region problem	$O(n \log^{3/2} n)$ $O(n \log n)$ space, $O(\log n)$ query	rectilinear regions single-source queries	[35]
L_1 WRP, two-point query	$O(\log^2 n)$ query $O(n^2 \log^2 n)$ space, preproc.	rectilinear regions	[35]
Bounded curvature shortest path	$O(n^4 \log n)$	moderate obstacles	[26]
Sailor's problem ($L = 0$)	$O(n^2)$	polygonal domain	[129]
Sailor's problem ($L > 0$)	poly(n, ϵ)	ϵ-approx	[129]
Max concealment, simple polygon	$O(v^2(v+n)^2)$	v viewpoints	[66]
Max concealment, polygonal domain	$O(v^4 n^4)$	v viewpoints	[66]
Min total turn	$O(E_{VG} \log n)$	polygonal domain	[13]

algorithm to compute, in time $O((n/\sqrt{\epsilon}) \log n)$, a path guaranteed to have length within a factor $(1+\epsilon)$ of the Euclidean shortest path length [101, 98].

WEIGHTED REGION METRIC

The ***weighted region problem*** (WRP) is to find an optimal s-t path according to the weighted region metric, d_f, induced by a given piecewise-constant weight function, f. This problem is a natural generalization of the shortest-path problem in a polygonal domain: Consider a weight function that assigns weight 1 to P and weight ∞ (or a sufficiently large constant) to the obstacles (the complement of P).

The weighted region problem models the minimum-time path problem for a point robot moving in a terrain of varied types (e.g., grassland, brushland, blacktop, bodies of water, etc), where each type of terrain has an assigned weight equal to the reciprocal of the maximum speed of traversal for the robot.

Assume that f is specified by a triangulation having n vertices, with each face assigned an integer weight $\alpha \in \{0, 1, \ldots, W, +\infty\}$. (We can allow edges of the triangulation to have a weight that is possibly distinct from that of the triangular facets on either side of it; in this way, "linear features" such as "roads" can be modeled.) Using an algorithm based on the continuous Dijkstra method, one can find a path whose weighted length is guaranteed to be within a factor of $(1+\epsilon)$ of optimal, where $\epsilon > 0$ is any user-specified degree of precision [107]. The time complexity of the algorithm is $O(E \cdot S)$, where E is the number of "events" in the continuous Dijkstra algorithm, and S is the complexity of performing a numerical search to solve the following subproblem: Find a $(1+\epsilon)$-shortest path from s to t that goes through a given sequence of k edges of the triangulation. It is known that $E = O(n^4)$ and that there are examples where E can actually achieve this upper bound. The numerical search can be done using a form of binary search that exploits the local optimality condition: An optimal path bends according to "Snell's Law of Refraction" when crossing a region boundary. This leads to a bound of $S = O(k^2 \log(nNW/\epsilon))$ on the time needed to perform a search on a k-edge sequence, where N is the largest integer coordinate of any vertex of the triangulation. Since one can show that $k = O(n^2)$, this yields an overall time bound of $O(n^8 L)$, where $L = \log(nNW/\epsilon)$ can be thought of as the bit complexity of the problem instance.

Various special cases of the weighted region problem admit faster and simpler algorithms. For example, if the weighted subdivision is rectilinear, and path length is measured according to weighted L_1 length, then efficient algorithms for single-source and two-point queries can be based upon searching a path-preserving graph [35]. Similarly, if the region weights are restricted to $\{0, 1, \infty\}$ (while edges may have arbitrary (nonnegative) weights), then an $O(n^2)$ algorithm can be based on constructing a path-preserving graph similar to a visibility graph [66]. This also leads to an efficient method for performing ***lexicographic*** optimization, in which one prioritizes various types of regions according to which is most important for path length minimization.

MINIMUM-TIME PATHS

The minimum-time path problem is a difficult optimal control problem; optimal paths will be complicated curves given by solutions to differential equations.

As a first step towards understanding the algorithmic complexity of computing minimum-time paths under dynamic constraints, there is a polynomial-time $(1+\epsilon)$-approximation algorithm [27]. The approach is to discretize the four-dimensional phase space that represents position and velocity, with special care to ensure that the size of the grid is bounded by a polynomial in $1/\epsilon$ and n and that the shortest paths obtained from the resulting graph are guaranteed to be close to optimal.

If there is an upper bound on the L_∞ norm of the velocity and acceleration vectors, one can obtain an *exact*, exponential-time, polynomial-space algorithm, based on characterizing a set of "canonical solutions" (related to "bang-bang" controls) that are guaranteed to include an optimal solution path [28]. This leads to an expression in the first-order theory of the reals, which can be solved exactly. However, it remains an open question whether or not a polynomial-time algorithm exists.

A closely related shortest-path problem is the ***bounded curvature shortest-path problem***, in which we require that no point of the path have a radius of curvature less than 1. For this problem, $(1+\epsilon)$-approximation algorithms are known, with polynomial ($O(\frac{n^2}{\epsilon^2}\log n)$) running time [137], but the complexity of solving the problem exactly remains open. For the special case in which the obstacles are "moderate" (have differentiable boundary curves, with radius of curvature at least 1), both an approximation algorithm[3], and, most recently, an exact $O(n^4 \log n)$ algorithm have been found[26].

OPTIMAL ROBOT MOTION

So far, we have considered only the problem of optimally moving a *point* robot. If the robot is modeled as a circle, or as a nonrotating polygon, then many of the results carry over by simply applying the standard ***configuration space*** approach in motion planning: "shrink" the robot to a (reference) point, and "grow" the obstacles (using a Minkowski sum) so that the complement of the grown obstacles model the region of the plane for which there is no collision with an obstacle if the robot has its reference point placed there [39, 78].

Optimal motion of *rotating* non-circular robots is a much harder problem. Even the simplest case of moving a (unit) line segment (a ***ladder***) in the plane is highly nontrivial. One notion of "optimal" motion requires that we minimize the average distance traveled by a set of k fixed points, evenly distributed along the ladder. This "d_k-distance" in fact defines a metric (for $k \geq 2$). The special case of $k = 2$ is the well-known ***Ulam's problem***, for which optimal motions are fully characterized in the absence of obstacles [87]. The case of $k = \infty$ is an especially interesting case, requiring that we compute a minimum *work* motion of a ladder; however, no results are known for this problem. (The work measures the integral (over $\lambda \in [0, 1]$) of the path length, $L(\lambda)$, for each infinitesimal subsegment of length $d\lambda$.) O'Rourke [118] has studied a restricted case of the d_∞-optimal motion problem. While d_1 does not define a metric, several cases of d_1-motion, and its generalization of measuring the distance traveled by any fixed "focus" F on the ladder, have been studied. In particular, if F is restricted to move on the visibility graph of a polygonal environment, polynomial-time algorithms are known [121, 131]. Without restrictions, minimizing the d_1-distance (for any F not at an endpoint of the ladder) is NP-hard, but there exists an approximation algorithm [18].

MULTIPLE CRITERIA OPTIMAL PATHS

The standard shortest-path problem asks for paths that minimize some *one* objective (length) function. Frequently, however, an application requires us to find paths to minimize *two or more* objectives; the resulting problem is a ***bicriteria*** (or ***multi-criteria***) shortest-path problem. A path is called ***efficient*** or ***Pareto optimal*** if no other path has a better value for one criterion without having a worse value for the other criterion.

Multi-criteria optimization problems tend to be hard. Even the bicriteria path problem in a graph is NP-hard [65]: Does there exist a path from s to t whose length is less than L and whose weight is less than W? Pseudo-polynomial time algorithms are known, and many heuristics have been devised (e.g., see [73, 74]).

In geometric problems, various optimality criteria are of interest, including any pair from the following list: Euclidean (L_2) length, rectilinear (L_1) length, other L_p metrics, link distance, total turn, etc.

NP-hardness lower bounds are known for several versions, including: [13] (1) Find a path in a polygonal domain whose L_2 length is at most L, and whose "total turn" is at most T; (2) Find a path in a polygonal domain whose L_p length is at most λ_p and whose L_q length is at most λ_q ($p \neq q$); and (3) Given a subdivision of the plane into red and blue polygonal regions, find a path whose length within blue regions is at most B and whose length within red regions is at most R.

One problem of particular interest is to compute a Euclidean shortest path within a polygonal domain, constrained to have at most k links. No exact solution is currently known for this problem. Part of the difficulty is that a minimum-link path will not, in general, lie on the visibility graph (or any simple discrete graph). Furthermore, the computation of the turn points of such an optimal path appear to require the solution to high-degree polynomials.

For a given k ($k \geq d_L$, where d_L is the s-t link distance), one can compute a path in a *simple* polygon P whose length is guaranteed to be within a factor $(1+\epsilon)$ of the length of a shortest k-link path, for any tolerance $\epsilon > 0$. The algorithm runs in time $O(n^3k^3 \log(Nk/\epsilon^{1/k}))$, polynomial in n and k, and logarithmic in $1/\epsilon$ and the largest integer coordinate N of any vertex of P [108]. Within the same time bound, one can compute an ϵ-optimal path under any (single) *combined* objective, $f(L,G)$, where L and G denote link distance and Euclidean length, and f is an increasing function in G for each L.

Does there exist an s-t path that is *simultaneously* close to Euclidean shortest and minimum-link? In a *simple* polygon, one can always find an s-t path whose link length is within a factor of 2 of the link distance from s to t, while also having Euclidean length within a factor of $\sqrt{2}$ of the Euclidean shortest-path length [14]. A corresponding result is not possible for polygons with holes. However, in $O(kE_{VG}^2)$ time, one can compute a path in a polygonal domain having at most $2k$ links and length at most that of a shortest k-link path [108].

In a rectilinear polygonal domain, efficient algorithms are known for the bicriteria path problem that combines *rectilinear* link distance and L_1 length. For example, de Berg et al. [57, 56] give efficient algorithms in two or more dimensions for computing optimal paths according to a "combined metric," defined to be a linear combination of rectilinear link distance and L_1 path length. (Note that this is not the same as computing the Pareto-optimal solutions.) Yang, Lee, and Wong [139, 138] give an $O(n \log^2 n)$ algorithm for computing a shortest k-bend path, a minimum-bend shortest path, or any combined objective that uses a monotonic function of rectilinear link length and L_1 length in a rectilinear polygonal domain. In all of these rectilinear problems, there is an underlying grid graph which can serve as a "path preserving graph."

OPEN PROBLEMS

1. Can a minimum-link path in a polygonal domain be computed in subquadratic time? The only lower bound known is $\Omega(n \log n)$ [109].
2. What is the smallest size data structure for a simple polygon P that allows logarithmic-time two-point link distance queries?
3. For a polygonal domain (with holes) what is the complexity of computing a shortest k-link path between two given points?
4. What is the complexity of the ladder problem for a polygonal domain, in which the cost of motion is the total work involved in translation/rotation?
5. Does minimizing the d_1 distance of a ladder endpoint remain NP-hard?
6. What is the complexity of the bounded curvature shortest-path problem?

24.5 OTHER NETWORK OPTIMIZATION PROBLEMS

All of the problems considered so far involved computing a shortest path from one point to another (or from one point to all other points). We consider now some other network optimization problems, in which the objective is to compute a shortest path, cycle, tree, or other graph, subject to some constraints. A summary of results is given in Table 24.5.

GLOSSARY

Minimum spanning tree (MST) of S: A tree of minimum total length whose nodes are a given set S of n points, and whose edges are line segments joining pairs of points.

Minimum Steiner spanning tree (Steiner tree) of S: A tree of minimum total length whose nodes are a superset of a given set S of n points, and whose edges are line segments joining pairs of points. Those nodes that are not points of S are called ***Steiner points***.

k-Minimum spanning tree (k-MST): A minimum-length tree that spans some subset of $k \leq n$ points of S.

Traveling salesman problem (TSP): Find a shortest cycle that visits every point of a set S of n points.

MAX TSP: Find a *longest* cycle that visits every point of a set S of n points.

Min/max-area TSP: Find a cycle on a given set S of points such that the cycle defines a simple polygon of minimum/maximum area.

TSP with neighborhoods: Find a shortest cycle that visits at least one point in each of a set of n neighborhoods (e.g., polygons).

Watchman route (path) problem: Find a shortest possible cycle (path) within a polygonal domain P, such that every point of P is seen by some point of the cycle.

Lawnmowing problem: Find a shortest cycle (path) for the motion of a disk (a "lawnmower," or "cutter") such that every point of a given (possibly disconnected) region is covered by the disk at some position along the cycle (path).

Milling problem: Exactly like the Lawnmowing Problem, but with the constraint that the cutter must at all times remain inside the given region ("pocket" to be milled).

Zookeeper's problem: Find a shortest cycle in a simple polygon P (the ***zoo***), through a given vertex v, such that the cycle visits every one of a set of k disjoint convex polygons (***cages***), each sharing an edge with P.

Aquarium keeper's Problem: Find a shortest cycle in a simple polygon P (the ***aquarium***), such that the cycle touches every edge of P.

Red-blue separation problem: Find a minimum-length simple polygon that separates a set of "red" points from a set of "blue" points.

Relative convex hull of point set S within simple polygon P: The shortest cycle within P that surrounds S. The relative convex hull is necessarily a simple polygon, with vertices among the points of S and the vertices of P.

Monotone path problem: Find a shortest monotone path (if any) from s to t in a polygonal domain P. (A polygonal path is monotone if there exists a direction vector d such that every directed edge of the path has a nonnegative inner product with d.)

MINIMUM SPANNING TREES

The (Euclidean) minimum spanning tree problem can be solved to optimality in the plane in time $O(n \log n)$, by appealing to the fact that the MST is a subgraph of the Delaunay triangulation.

The Steiner tree and k-MST problems, however, are NP-hard. Polynomial-time approximation schemes have recently been obtained, allowing one, for any fixed $\epsilon > 0$, to get within a factor $(1+\epsilon)$ of optimal in time $O(n^{O(1/\epsilon)})$ [17, 113].

TRAVELING SALESMAN PROBLEM

The traveling salesman problem is a classical problem in combinatorial optimization, and has been studied extensively in its geometric forms [93, 24, 126]. The problem is NP-hard, but has a simple 2-approximation algorithm based on "doubling" the minimum spanning tree. The somewhat more involved Christofides heuristic yields a 1.5-approximation factor, which, until very recently, was the best factor known. There is now a polynomial-time approximation scheme for geometric versions of the planar TSP, allowing one, for any fixed $\epsilon > 0$, to get within a factor $(1+\epsilon)$ of optimal in time $O(n^{O(1/\epsilon)})$ [17, 113].

The ***TSP with neighborhoods*** problem arises when we require that the tour/path visit a set of regions, rather than a set of points. Constant-factor approximation algorithms are known for some special cases [11], and an $O(\log n)$-approximation algorithm is known for the general case in the plane [96].

TABLE 24.5 Other optimal path/cycle/network problems.

PROBLEM	COMPLEXITY	NOTES	SOURCE
Minimum spanning tree (MST)	$O(n \log n)$	uses Delaunay triangulation	[125]
Steiner tree	NP-hard		[25]
Steiner tree	$O(n \log n)$	$\frac{2}{\sqrt{3}}$-approx, using MST	[25, 61]
Steiner tree	$O(n^{O(1/\epsilon)})$	$(1+\epsilon)$-approx	[17, 113]
k-MST	NP-hard		[112]
k-MST	$O(n^{O(1/\epsilon)})$	$(1+\epsilon)$-approx	[17, 113]
Traveling salesman problem (TSP)	NP-hard		[119]
Traveling salesman problem (TSP)	$O(n^{O(1/\epsilon)})$	$(1+\epsilon)$-approx	[17, 113]
MAX TSP	open	$(1+\epsilon)$-approx	[23]
Min-area TSP	NP-complete		[63]
Max-area TSP	NP-complete	$(1/2)$-approx	[63]
TSP with neighborhoods (restricted)	NP-hard	$O(1)$-approx	[11]
TSP with neighborhoods (general)	NP-hard	$O(\log n)$-approx	[96]
Watchman route in simple polygon	$O(n^4)$		[31, 115]
Watchman Route in Simple Polygon	$O(n^2)$	through a *given* point	[136]
Watchman route in rectilinear simple polygon	$O(n)$		[44]
Watchman path in simple polygon	$O(n^{12})$		[30]
Min-link watchman in simple polygon	NP-hard	$O(1)$-approx	[4, 5]
Watchman route in polygonal domain	NP-hard	$O(\log k)$-approx	[43, 96]
Min-link watchman in polygonal domain	NP-hard	$O(\log n)$-approx	[12]
Lawnmowing problem	NP-hard	$O(1)$-approx	[8, 9]
Milling problem in simple polygon	open	$O(1)$-approx	[8, 9]
Milling Problem in polygonal domain	NP-hard	$O(1)$-approx	[8, 9]
Shortest postman path	$O(n^3)$	simple polygon	[30]
Simple s-t Hamiltonian path	$O(n^2 m(n+m))$	m points in simple n-gon	[50]
	NP-Complete	in polygonal domain	[50]
Aquarium-keeper's problem	$O(n)$	simple polygon	[53]
Aquarium-keeper's path problem	$O(n^4)$	simple polygon	[30]
Zookeeper's problem	$O(n \log^2 n)$	simple polygon	[80]
Red-blue separation	$O(n^5)$	$O(\log n)$-approx	[96]
Relative convex hull	$\Theta(n + k \log kn)$	k points in simple n-gon	[70]
Parallel Relative Convex Hull (CREW PRAM)	$O(\log kn)$ time, $O(k)$ processors	k points in a triangulated simple n-gon	[77]
Monotone path problem	$O(n^3 \log n)$		[10]

A closely related problem is that of computing an optimal path for a lawnmower, modeled as, say, a circular cutter that must sweep out a region that covers a given domain of "grass." This problem is NP-hard in general, but constant-factor approximation algorithms are known [8, 9, 89].

WATCHMAN ROUTE PROBLEM

Another problem closely related to the TSP is the watchman route problem, which can be thought of as a shortest-path/tour problem in which we have the constraint that the path/tour must visit the visibility region associated with each point of the domain.

In the case of a simple polygonal domain, the watchman route problem has an $O(n^4)$ time algorithm to compute an exact solution [31, 115]; $O(n^2)$ is possible if we are given a point through which the tour must pass [136]. (In a recent work of Nilsson et al. [?], an error was discovered in the earlier papers on watchman routes: It turns out that the "adjustment" algorithm that modifies a route in order to achieve the reflection property at essential cuts may in fact take an exponential number of iterations. However, the new adjustment algorithm in [?] reestablishes the previous claims.) In the case of a polygonal domain with holes, the problem is easily seen to be NP-hard (from Euclidean TSP), and the best approximation algorithm is one with error factor $O(\log n)$ [96].

OPEN PROBLEMS

1. Is the MAX TSP NP-hard?
2. Is the milling problem NP-hard for the case of a simple polygon?
3. Does the TSP with Neighborhoods problem have an efficient approximation algorithm, if the neighborhoods are not connected sets (e.g., if the neighborhoods are pairs of points)?
4. Does the watchman route problem in a polygonal domain have an $O(1)$-approximation algorithm?

24.6 HIGHER DIMENSIONS

GLOSSARY

Polyhedral domain: A connected subset, P, of R^3 whose boundary consists of a union of a finite number of triangles. (The definition is readily extended to d dimensions, where the boundary must consist of a union of "simplices.") The complement of P consists of connected (polyhedral) components, which are the ***obstacles***.

Orthohedral domain: A polyhedral domain having each boundary facet orthogonal to one of the coordinate axes.

Polyhedral surface: A connected union of triangles, with any two triangles intersecting in a common edge, a common vertex, or not at all.

Edge sequence: The ordered list of obstacle edges that are intersected by a path.

COMPLEXITY

In three or more dimensions, most shortest-path problems become very difficult. In particular, there are two sources of complexity, even in the most basic Euclidean shortest-path problem in a polyhedral domain P. The problem is difficult even if the obstacles are convex, or the domain P is simply connected.

One difficulty arises from algebraic considerations. In general, the structure of a shortest path in a polyhedral domain need not lie on any kind of discrete graph. Shortest paths in a polyhedral domain will be polygonal, with bend points that generally lie *interior* to obstacle edges, obeying a simple "unfolding" property: The path must enter and leave at the same angle to the edge. It follows that any locally optimal subpath joining two consecutive obstacle vertices can be "unfolded" at each edge along its edge sequence, thereby obtaining a straight segment. Given an edge sequence, this local optimality property uniquely identifies a shortest path through that edge sequence. However, to compare the lengths of two paths, each one shortest with respect to two (different) edge sequences, requires exponentially many bits, since the algebraic numbers that describe the optimal path lengths may have exponential degree [21, 22].

A second difficulty arises from combinatorial considerations. The number of combinatorially distinct (i.e., having distinct edge sequences) shortest paths between two points may be exponential. This fact leads to a proof of the NP-hardness of the shortest-path problem [29], even if the obstacles are simply a set of parallel triangles.

Thus, it is natural to consider approximation algorithms for the general case, or to consider special cases in which we can obtain polynomial bounds.

SPECIAL CASES

If the polyhedral domain P has only a small number, k, of convex obstacles, a shortest path can be found in $n^{O(k)}$ time [130]. If the obstacles are known to be vertical buildings having only k different heights, then shortest paths can be found in time $O(n^{6k-1})$ [67], but it is not known if this version of the problem is NP-hard if k is allowed to be large.

If we require paths to stay on a polyhedral surface (i.e., the domain P is essentially 2-dimensional), then the unfolding property of optimal paths can be exploited to yield polynomial-time algorithms. The current best methods are based on the continuous Dijkstra paradigm [38, 106], yielding an $O(n^2)$ time (and $O(n)$ space) algorithm to construct a shortest path map (or a geodesic Voronoi diagram), where n is the number of vertices of the surface.

Several facts are known about the set of edge sequences corresponding to shortest paths on the surface of a *convex* polytope P in R^3. In particular, the worst-case number of distinct edge sequences that correspond to a shortest path between some pair of points is $\Theta(n^4)$ [114], and the exact set of such sequences can be computed in time $O(n^6\beta(n)\log n)$, where $\beta(n) = o(\log^* n)$ [2]. (A simpler $O(n^6)$ algorithm can compute a small superset of the sequences [2].) The number of ***maximal*** edge sequences for shortest paths is $\Theta(n^3)$ [128]. Some of these results depend on a careful study of the "star unfolding" with respect to a point p on the boundary, ∂P, of P. The ***star unfolding*** is the (nonoverlapping [16]) cell complex obtained by subtracting from ∂P the shortest paths from p to vertices of P, and then "flattening" the resulting boundary [16, 2].

In the case of terrain surfaces (polyhedral surfaces having at most one intersection point with any line parallel to the z-axis), de Berg and van Kreveld [58] have studied optimal path problems, including some bicriteria versions, with constraints imposed on the maximum allowed altitude, steepest slope, etc.

Results on exact algorithms for special cases are summarized in Table 24.6.

APPROXIMATION ALGORITHMS

Papadimitriou [120] was the first to study the general problem from the point of view of approximations. He gave a fully polynomial approximation scheme that produces a path guaranteed to be no longer than $(1+\epsilon)$ times the length of a shortest path. His algorithm requires time $O(n^3(L+\log(n/\epsilon))^2/\epsilon)$, where L is the number of bits necessary to represent the value of an integer coordinate of a vertex of P. Clarkson [51] also gives a fully polynomial approximation scheme, which improves upon that of Papadimitriou [120] in the case that $n\epsilon^3$ is large.

Recently, Choi, Sellen, and Yap [45, 47] have re-examined closely the analysis of Papadimitriou and have addressed some inconsistencies found in the original algorithm. To this end, it is important to distinguish between the *bit* framework and the *algebraic* framework of studying the complexity of the problem. Almost all shortest path algorithms (and most computational geometry algorithms) assume an algebraic model of computation, in which the time complexity is measured in terms of the number of algebraic operations performed on real numbers. It is assumed that these operations are performed *exactly*. In the bit framework, though, time complexity is measured in terms of the number of boolean operations on bits, assuming the input is encoded with binary strings. Given the nature of current computer hardware, it is likely that the bit framework more accurately models actual computation times.

Choi, Sellen, and Yap [45] give upper bounds on the bit complexity of the approximate shortest-path problem. They have also introduced the important notion of "precision-sensitivity" in algorithms, where the goal is to write the complexity in terms of an implicit parameter, δ, that measures the implicit precision of the input instance [47]. For example, in the shortest-path problem, they define $\delta=(d_2-d^*)/d^*$ to be the relative difference between the length d^* of an optimal path, and the length, d_2, of the second-shortest, locally optimal path; i.e., $d_2>d^*$ is the length of a shortest path that uses an edge sequence distinct from any optimal edge sequence, but is closest in length to d^* among all such locally optimal paths. Provided that the optimal edge sequence is in some sense nondegenerate, one obtains an approximation algorithm that is polynomial in $1/\delta$ and the other parameters of the input, with only linear dependence on $1/\epsilon$.

Results on approximation algorithms, both in the general case and in special cases, are summarized in Table 24.7.

OTHER METRICS

Link distance in a polyhedral domain in R^d can be approximated (within factor 2) in polynomial time, by searching a weak visibility graph whose nodes correspond to simplices in a simplicial decomposition of the domain. The complexity of computing the exact link distance is open.

For the case of orthohedral domains, and rectilinear (L_1) shortest paths, the shortest-path problem in R^d becomes relatively easy to solve in polynomial time, since the "grid graph" induced by the facets of the domain serves as a path preserving graph that we can search for an optimal path. In R^3, we can do better than

TABLE 24.6 Shortest paths in 3-space: Exact algorithms.

OBSTACLES/DOMAIN	COMPLEXITY	NOTES	SOURCE
Polyhedral domain	NP-hard	even for convex obstacles	[29]
k Convex polytopes	$n^{O(k)}$	fixed k	[130]
Vertical buildings	$O(n^{6k-1})$	k different heights	[67]
Axis-parallel boxes	$O(n^2 \log^3 n)$	L_1 metric	[52]
Axis-parallel disjoint boxes	$O(n^2 \log n)$	L_1 metric	[49]
		monotonicity of paths in R^d	[48]
Axis-parallel boxes in R^d	$O(n^d \log n)$ preproc.	combined L_1, link metric	[57]
	$O(\log^{d-1} n)$ query	single-source queries	
	$O((n \log n)^{d-1})$ space		
Polyhedral surface	$O(n^2)$ time, $O(n)$ space	builds SPM(s), geodesic Voronoi	[38, 106]
Two-point query	$O((\sqrt{n}/m^{1/4}) \log n)$ query	convex polytope	[2]
	$O(n^6 m^{1+\delta})$ space, preproc.	$1 \le m \le n^2$, $\delta > 0$	
Geodesic diameter	$O(n^8 \log n)$	convex polytope	[2]

TABLE 24.7 Shortest paths in 3-space: Approximation algorithms.

OBSTACLES/DOMAIN	COMPLEXITY	APPROX FACTOR	SOURCE
Polyhedral domain	$O(n^4(L + \log(\frac{n}{\epsilon}))^2/\epsilon^2)$	$(1+\epsilon)$	[120, 45, 47]
	$O(n^2 \text{polylog } n/\epsilon^4)$	$(1+\epsilon)$	[51]
k Convex polytopes	$O(n)$	$2k$	[85]
Convex polyhedral surface	$O(n)$	2	[85]
Convex polyhedral surface	$O(n \log \frac{1}{\epsilon} + \frac{1}{\epsilon^3})$	$(1+\epsilon)$	[?]
Vertical buildings	$O(n^2)$	1.1	[67]
Min-link, polyhedral domain	poly(n)	2	

to use the $O(n^3)$ grid graph induced by $O(n)$ facets; an $O(n^2 \log^2 n)$ size subgraph suffices, which allows a shortest path to be found using Dijkstra's algorithm in time $O(n^2 \log^3 n)$ [52]. More generally, in R^d, one can compute a data structure of size $O((n \log n)^{d-1})$, in $O(n^d \log n)$ preprocessing time, that supports fixed-source link distance queries in $O(\log^{d-1} n)$ time [56]. In fact, this last result can be extended, within the same complexities, to the case of a ***combined metric***, in which path cost is measured as a linear combination of L_1 length and the rectilinear link distance.

For the special case of disjoint rectilinear box obstacles, and rectilinear (L_1) shortest paths, a recent structural result may help in devising very efficient algorithms: There always exists a coordinate direction such that *every* shortest path from s to t is monotone in this direction [46, 48]. In fact, this result has lead to an $O(n^2 \log n)$ algorithm for the case of $d = 3$ [49, 46].

OPEN PROBLEMS

1. Can one compute shortest paths on a convex polytope in R^3 in subquadratic time?

2. Can one compute a shortest path map for a polyhedral domain in output-sensitive time?
3. What is the complexity of the minimum-link path problem in 3-space?
4. Can one give very simple and efficient approximation algorithms for shortest paths in polyhedral domains or on nonconvex surfaces?
5. Can an approximate shortest path on a nonconvex polyhedral surface be computed in near-linear time?
6. Can one preprocess a convex polytope in near-linear time in order to permit logarithmic-time two-point queries for approximate shortest path distances? Can one compute an approximate SPM in near-linear time?
7. What is the complexity of the shortest-path problem in 3-space for special cases of obstacles — e.g., disjoint aligned boxes, unit spheres, etc?

24.7 SOURCES AND RELATED MATERIAL

SURVEYS

Several other surveys offer a wealth of additional and related material:

- [6]: A survey of shortest paths and visibility graphs.
- [25]: A survey of approximation algorithms for geometric optimization problems.
- [86]: A book on motion planning algorithms.
- [95]: A recent survey article on rectilinear path problems.
- [110]: A survey of computational geometry, with a large section on shortest paths.
- [132]: A survey of topological network design problems.

ACKNOWLEDGEMENTS

I want thank several people who have contributed significantly to improving this survey: Yi-Jen Chiang, Peter Johansson, Bengt Nilsson, Juergen Sellen, Chee Yap. A special thanks goes to Joe O'Rourke, who not only encouraged the writing of this survey, but helped in its preparation through many useful and detailed suggestions that greatly improved the final product.

REFERENCES

References

[1] John Adegeest, Mark Overmars, and Jack Snoeyink. Minimum-link c-oriented paths: Single-source queries. *Internat. J. Comput. Geom. Appl.*, 4(1):39–51, 94.

[2] P. K. Agarwal, B. Aronov, J. O'Rourke, and C. Schevon. Star unfolding of a polytope with applications. *SIAM J. Comput.*, 1996. To appear.

[3] Pankaj K. Agarwal, P. Raghavan, and H. Tamaki. Motion planning for a steering-constrained robot through moderate obstacles. In *Proc. 27th Annu. ACM Sympos. Theory Comput.*, pages 343–352, 1995.

[4] M. H. Alsuwaiyel and D. T. Lee. Minimal link visibility paths inside a simple polygon. *Comput. Geom. Theory Appl.*, 3(1):1–25, 1993.

[5] M. H. Alsuwaiyel and D. T. Lee. Finding an approximate minimum-link visibility path inside a simple polygon. *Inform. Process. Lett.*, 55(2):75–79, 1995.

[6] H. Alt and E. Welzl. Visibility graphs and obstacle-avoiding shortest paths. *Zeitschrift für Operations Research*, 32:145–164, 1988.

[7] N. M. Amato, M. T. Goodrich, and E. A. Ramos. Computing faces in segment and simplex arrangements. In *Proc. 27th Annu. ACM Sympos. Theory Comput.*, pages 672–682, 1995.

[8] E. Arkin, S. Fekete, and J. Mitchell. The lawnmower problem. In *Proc. 5th Canad. Conf. Comput. Geom.*, pages 461–466, Waterloo, Canada, 1993.

[9] E. Arkin, S. Fekete, and J. Mitchell. Approximation algorithms for lawnmowing and milling. Report, University at Stony Brook, 1995.

[10] E. M. Arkin, R. Connelly, and J. S. B. Mitchell. On monotone paths among obstacles, with applications to planning assemblies. In *Proc. 5th Annu. ACM Sympos. Comput. Geom.*, pages 334–343, 1989.

[11] E. M. Arkin and R. Hassin. Approximation algorithms for the geometric covering salesman problem. *Discrete Appl. Math.*, 55:197–218, 1994.

[12] E. M. Arkin, J. S. B. Mitchell, and C. Piatko. Minimum-link watchman tours. Report, University at Stony Brook, 1994.

[13] E. M. Arkin, J. S. B. Mitchell, and C. D. Piatko. Bicriteria shortest path problems in the plane. In *Proc. 3rd Canad. Conf. Comput. Geom.*, pages 153–156, 1991.

[14] E. M. Arkin, J. S. B. Mitchell, and S. Suri. Logarithmic-time link path queries in a simple polygon. *Internat. J. Comput. Geom. Appl.*, 5(4):369–395, 1995.

[15] B. Aronov, S. J. Fortune, and G. Wilfong. Furthest-site geodesic Voronoi diagram. *Discrete Comput. Geom.*, 9:217–255, 1993.

[16] B. Aronov and J. O'Rourke. Nonoverlap of the star unfolding. *Discrete Comput. Geom.*, 8:219–250, 1992.

[17] S. Arora. Polynomial time approximation schemes for Euclidean TSP and other geometric problems. In *Proc. 37th Annu. IEEE Sympos. Found. Comput. Sci. (FOCS 96)*, pages 2–12, 1996.

[18] Tetsuo Asano, David Kirkpatrick, and Chee K. Yap. d_1-Optimal motion for a rod. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, page to appear, 1996.

[19] M. Atallah and D. Chen. On parallel rectilinear obstacle-avoiding paths. *Comput. Geom. Theory Appl.*, 3:307–313, 1993.

[20] M. J. Atallah and D. Z. Chen. Parallel rectilinear shortest paths with rectangular obstacles. *Comput. Geom. Theory Appl.*, 1:79–113, 1991.

[21] C. Bajaj. The algebraic complexity of shortest paths in polyhedral spaces. In *Proc. 23rd Allerton Conf. Commun. Control Comput.*, pages 510–517, 1985.

[22] C. Bajaj. The algebraic degree of geometric optimization problems. *Discrete Comput. Geom.*, 3:177–191, 1988.

[23] A. Barvinok. Two algorithmic results for the TSP. *Math. Oper. Res.*, 21:65–84, 1996.

[24] J. L. Bentley. Fast algorithms for geometric traveling salesman problems. *ORSA J. Comput.*, 4(4):387–411, 1992.

[25] M. Bern and D. Eppstein. Approximation algorithms for geometric problems. In Dorit Hochbaum, editor, *Approximation Problems for NP-Complete Problems*, page To appear. PWS Publications, 1996.

[26] Jean-Daniel Boissonnat and Sylvain Lazard. A polynomial-time algorithm for computing a shortest path of bounded curvature amidst moderate obstacles. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, page ??, 1996.

[27] J. Canny, B. R. Donald, J. Reif, and P. Xavier. On the complexity of kinodynamic planning. In *Proc. 29th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 306–316, 1988.

[28] J. Canny, A. Rege, and J. Reif. An exact algorithm for kinodynamic planning in the plane. *Discrete Comput. Geom.*, 6:461–484, 1991.

[29] J. Canny and J. H. Reif. New lower bound techniques for robot motion planning problems. In *Proc. 28th Annu. IEEE Sympos. Found. Comput. Sci.*, pages 49–60, 1987.

[30] Svante Carlsson and Hakan Jonsson. Computing a shortest watchman path in a simple polygon in polynomial-time. In *Proc. 4th Workshop Algorithms Data Struct.*, volume 955 of *Lecture Notes in Computer Science*, pages 122–134. Springer-Verlag, 1995.

[31] Svante Carlsson, Hakan Jonsson, and Bengt J. Nilsson. Finding the shortest watchman route in a simple polygon. In *Proc. 4th Annu. Internat. Sympos. Algorithms Comput. (ISAAC 93)*, volume 762 of *Lecture Notes in Computer Science*, pages 58–67. Springer-Verlag, 1993.

[32] B. Chazelle. A theorem on polygon cutting with applications. In *Proc. 23rd Annu. IEEE Sympos. Found. Comput. Sci.*, pages 339–349, 1982.

[33] B. Chazelle. Triangulating a simple polygon in linear time. *Discrete Comput. Geom.*, 6:485–524, 1991.

[34] B. Chazelle, H. Edelsbrunner, M. Grigni, L. Guibas, J. Hershberger, M. Sharir, and J. Snoeyink. Ray shooting in polygons using geodesic triangulations. In *Proc. 18th Internat. Colloq. Automata Lang. Program.*, volume 510 of *Lecture Notes in Computer Science*, pages 661–673. Springer-Verlag, 1991.

[35] D. Z. Chen, K. S. Klenk, and H-Y. T. Tu. Shortest path queries among weighted obstacles in the rectilinear plane. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 370–379, 1995.

[36] Danny Z. Chen. On the all-pairs Euclidean short path problem. In *Proc. 6th ACM-SIAM Sympos. Discrete Algorithms (SODA '95)*, pages 292–301, San Francisco, CA, 1995.

[37] Danny Z. Chen and Kevin S. Klenk. Rectilinear short path queries among rectangular obstacles. In *Proc. 7th Canad. Conf. Comput. Geom.*, pages 169–174, 1995.

[38] J. Chen and Y. Han. Shortest paths on a polyhedron. In *Proc. 6th Annu. ACM Sympos. Comput. Geom.*, pages 360–369, 1990.

[39] L. P. Chew. Planning the shortest path for a disc in $O(n^2 \log n)$ time. In *Proc. 1st Annu. ACM Sympos. Comput. Geom.*, pages 214–220, 1985.

[40] Y.-J. Chiang, F. P. Preparata, and R. Tamassia. A unified approach to dynamic point location, ray shooting, and shortest paths in planar maps. In *Proc. 4th ACM-SIAM Sympos. Discrete Algorithms*, pages 44–53, 1993.

[41] Y.-J. Chiang, F. P. Preparata, and R. Tamassia. A unified approach to dynamic point location, ray shooting, and shortest paths in planar maps. *SIAM J. Comput.*, 25:207–233, 1996.

[42] Y.-J. Chiang and R. Tamassia. Optimal shortest path and minimum-link path queries between two convex polygons inside a simple polygonal obstacle. *Internat. J. Comput. Geom. Appl.*, page to appear, 1996.

[43] W. Chin and S. Ntafos. Optimum watchman routes. *Inform. Process. Lett.*, 28:39–44, 1988.

[44] W.-P. Chin and S. Ntafos. Watchman routes in simple polygons. *Discrete Comput. Geom.*, 6(1):9–31, 1991.

[45] J. Choi, J. Sellen, and C. K. Yap. Approximate Euclidean shortest path in 3-space. In *Proc. 10th Annu. ACM Sympos. Comput. Geom.*, pages 41–48, 1994.

[46] J. S. Choi. *Geodesic problems in high dimensions*. PhD thesis, Courant Institute, New York University, New York, June 1995.

[47] Joonsoo Choi, Jürgen Sellen, and Chee-Keng Yap. Precision-sensitive Euclidean shortest path in 3-space. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 350–359, 1995.

[48] Joonsoo Choi and Chee K. Yap. Monotonicity of rectilinear geodesics in d-space. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, page to appear, 1996.

[49] Joonsoo Choi and Chee-Keng Yap. Rectilinear geodesics in 3-space. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 380–389, 1995.

[50] Marek Chrobak and G. Sundaram. Paths among points inside a simple polygon. In *Proc. 5th MSI Stony Brook Workshop on Computational Geometry*, 1995.

[51] K. L. Clarkson. Approximation algorithms for shortest path motion planning. In *Proc. 19th Annu. ACM Sympos. Theory Comput.*, pages 56–65, 1987.

[52] K. L. Clarkson, S. Kapoor, and P. M. Vaidya. Rectilinear shortest paths through polygonal obstacles in $O(n(\log n)^2)$ time. In *Proc. 3rd Annu. ACM Sympos. Comput. Geom.*, pages 251–257, 1987.

[53] J. Czyzowicz, P. Egyed, H. Everett, D. Rappaport, T. Shermer, D. Souvaine, G. Toussaint, and J. Urrutia. The aquarium keeper's problem. In *Proc. 2nd ACM-SIAM Sympos. Discrete Algorithms*, pages 459–464, 1991.

[54] G. Das and G. Narasimhan. Geometric searching and link distances. In *Proc. 2nd Workshop Algorithms Data Struct.*, volume 519 of *Lecture Notes in Computer Science*, pages 261–272. Springer-Verlag, 1991.

[55] M. de Berg. On rectilinear link distance. *Comput. Geom. Theory Appl.*, 1(1):13–34, July 1991.

[56] M. de Berg, M. van Kreveld, and B. J. Nilsson. Shortest path queries in rectilinear worlds of higher dimension. In *Proc. 7th Annu. ACM Sympos. Comput. Geom.*, pages 51–60, 1991.

[57] M. de Berg, M. van Kreveld, B. J. Nilsson, and M. H. Overmars. Shortest path queries in rectilinear worlds. *Internat. J. Comput. Geom. Appl.*, 2(3):287–309, 1992.

[58] Mark de Berg and Marc van Kreveld. Trekking in the alps without freezing or getting tired. In *1st Annual European Symposium on Algorithms (ESA '93)*, volume 726 of *Lecture Notes in Computer Science*, pages 121–132. Springer-Verlag, 1993.

[59] H. N. Djidjev, A. Lingas, and J. Sack. An $O(n \log n)$ algorithm for computing the link center of a simple polygon. *Discrete Comput. Geom.*, 8(2):131–152, 1992.

[60] J. R. Driscoll, H. N. Gabow, R. Shrairaman, and R. E. Tarjan. Relaxed heaps: An alternative to Fibonacci heaps with applications to parallel computation. *Commun. ACM*, 31:1343–1354, 1988.

[61] D.-Z. Du and F. K. Hwang. The state of art on Steiner ratio problems. In D.-Z. Du and F. K. Hwang, editors, *Computing in Euclidean Geometry*, volume 1 of *Lecture Notes Series on Computing*, pages 163–191. World Scientific, Singapore, 1992.

[62] H. ElGindy and P. Mitra. Orthogonal shortest route queries among axis parallel rectangular obstacles. *Internat. J. Comput. Geom. Appl.*, 4(1):3–24, 1994.

[63] S. P. Fekete and W. R. Pulleyblank. Area optimization of simple polygons. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 173–182, 1993.

[64] M. Fredman and R. E. Tarjan. Fibonacci heaps and their uses in improved network optimization problems. *J. ACM*, 34:596–615, 1987.

[65] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, New York, NY, 1979.

[66] L. Gewali, A. Meng, J. S. B. Mitchell, and S. Ntafos. Path planning in $0/1/\infty$ weighted regions with applications. *ORSA J. Comput.*, 2(3):253–272, Summer 1990.

[67] L. Gewali, S. Ntafos, and I. G. Tollis. Path planning in the presence of vertical obstacles. Technical report, Computer Science, University of Texas at Dallas, 1989.

[68] S. K. Ghosh and D. M. Mount. An output-sensitive algorithm for computing visibility graphs. *SIAM J. Comput.*, 20:888–910, 1991.

[69] M. T. Goodrich and R. Tamassia. Dynamic ray shooting and shortest paths via balanced geodesic triangulations. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 318–327, 1993.

[70] L. J. Guibas and J. Hershberger. Optimal shortest path queries in a simple polygon. *J. Comput. Syst. Sci.*, 39:126–152, 1989.

[71] L. J. Guibas, J. Hershberger, D. Leven, M. Sharir, and R. E. Tarjan. Linear-time algorithms for visibility and shortest path problems inside triangulated simple polygons. *Algorithmica*, 2:209–233, 1987.

[72] L. J. Guibas, J. E. Hershberger, J. S. B. Mitchell, and J. S. Snoeyink. Approximating polygons and subdivisions with minimum link paths. *Internat. J. Comput. Geom. Appl.*, 3(4):383–415, December 1993.

[73] G. Y. Handler and I. Zang. A dual algorithm for the constrained shortest path problem. *Networks*, 10:293–310, 1980.

[74] M.I. Henig. The shortest path problem with two objective functions. *European J. of Operational Research*, 25:281–291, 1985.

[75] J. Hershberger. An optimal visibility graph algorithm for triangulated simple polygons. *Algorithmica*, 4:141–155, 1989.

[76] J. Hershberger. A new data structure for shortest path queries in a simple polygon. *Inform. Process. Lett.*, 38:231–235, 1991.

[77] J. Hershberger. Optimal parallel algorithms for triangulated simple polygons. In *Proc. 8th Annu. ACM Sympos. Comput. Geom.*, pages 33–42, 1992.

[78] J. Hershberger and L. J. Guibas. An $O(n^2)$ shortest path algorithm for a non-rotating convex body. *J. Algorithms*, 9:18–46, 1988.

[79] J. Hershberger and J. Snoeyink. Computing minimum length paths of a given homotopy class. *Comput. Geom. Theory Appl.*, 4:63–98, 1994.

[80] J. Hershberger and J. Snoeyink. An efficient solution to the zookeeper's problem. In *Proc. 6th Canad. Conf. Comput. Geom.*, pages 104–109, 1994.

[81] J. Hershberger and S. Suri. Matrix searching with the shortest path metric. In *Proc. 25th Annu. ACM Sympos. Theory Comput. (STOC 93)*, pages 485–494, 1993.

[82] J. Hershberger and S. Suri. A pedestrian approach to ray shooting: Shoot a ray, take a walk. volume 18, pages 403–431, 1995.

[83] John Hershberger and Subhash Suri. Efficient computation of Euclidean shortest paths in the plane. In *Proc. 34th Annu. IEEE Sympos. Found. Comput. Sci. (FOCS 93)*, pages 508–517, 1993.

[84] John Hershberger and Subhash Suri. An optimal algorithm for Euclidean shortest paths in the plane. Manuscript, Washington University, 1995.

[85] John Hershberger and Subhash Suri. Practical methods for approximating shortest paths on a convex polytope in $\Re^3$. In *Proc. 6th ACM-SIAM Sympos. Discrete Algorithms (SODA '95)*, pages 447–456, San Francisco, CA, 1995.

[86] J. E. Hopcroft, J. T. Schwartz, and M. Sharir. *Planning, Geometry, and Complexity of Robot Motion*. Ablex Publishing, Norwood, NJ, 1987.

[87] C. Icking, G. Rote, E. Welzl, and C. Yap. Shortest paths for line segments. *Algorithmica*, 10:182–200, 1993.

[88] H. Imai and M. Iri. Polygonal approximations of a curve-formulations and algorithms. In G. T. Toussaint, editor, *Computational Morphology*, pages 71–86. North-Holland, Amsterdam, Netherlands, 1988.

[89] K. Iwano, P. Raghavan, and H. Tamaki. The traveling cameraman problem, with applications to automatic optical inspection. In *Proc. 5th Annu. Internat. Sympos. Algorithms Comput. (ISAAC '94)*, volume ?? of *Lecture Notes in Computer Science*, page ??, Beijing, 1994. Springer-Verlag.

[90] Simon Kahan and Jack Snoeyink. On the bit complexity of minimum link paths: Superquadratic algorithms for problems solvable in linear time. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, pages 151–158, 1996.

[91] S. Kapoor and S. N. Maheshwari. An efficient algorithm for Euclidean shortest path with polygonal obstacles. Technical Report, IIT, New Delhi, 1994.

[92] Y. Ke. An efficient algorithm for link-distance problems. In *Proc. 5th Annu. ACM Sympos. Comput. Geom.*, pages 69–78, 1989.

[93] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys, editors. *The Traveling Salesman Problem.* Wiley, New York, NY, 1985.

[94] D. T. Lee and F. P. Preparata. Euclidean shortest paths in the presence of rectilinear barriers. *Networks*, 14:393–410, 1984.

[95] D. T. Lee, C. D. Yang, and C. K. Wong. Rectilinear paths among rectilinear obstacles. *Discrete Appl. Math.*, 70:185–215, 1996.

[96] C. Mata and J. S. Mitchell. Approximation algorithms for geometric tour and network design problems. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 360–369, 1995.

[97] J. S. B. Mitchell. *Planning shortest paths.* Ph.D. thesis, Stanford Univ., Stanford, CA, 1986.

[98] J. S. B. Mitchell. An optimal algorithm for shortest rectilinear paths among obstacles. In *Abstracts 1st Canad. Conf. Comput. Geom.*, page 22, 1989.

[99] J. S. B. Mitchell. On maximum flows in polyhedral domains. *J. Comput. Syst. Sci.*, 40:88–123, 1990.

[100] J. S. B. Mitchell. A new algorithm for shortest paths among obstacles in the plane. *Ann. Math. Artif. Intell.*, 3:83–106, 1991.

[101] J. S. B. Mitchell. L_1 shortest paths among polygonal obstacles in the plane. *Algorithmica*, 8:55–88, 1992.

[102] J. S. B. Mitchell. Shortest paths among obstacles in the plane. In *Proc. 9th Annu. ACM Sympos. Comput. Geom.*, pages 308–317, 1993.

[103] J. S. B. Mitchell. Euclidean shortest paths among polygonal obstacles in the plane. Technical Report, University at Stony Brook, 1994.

[104] J. S. B. Mitchell. Shortest paths among obstacles in the plane. *Internat. J. Comput. Geom. Appl.*, 6:to appear, 1996.

[105] J. S. B. Mitchell. Shortest paths and networks. Technical Report, University at Stony Brook, 1996.

[106] J. S. B. Mitchell, D. M. Mount, and C. H. Papadimitriou. The discrete geodesic problem. *SIAM J. Comput.*, 16:647–668, 1987.

[107] J. S. B. Mitchell and C. H. Papadimitriou. The weighted region problem: finding shortest paths through a weighted planar subdivision. *J. ACM*, 38:18–73, 1991.

[108] J. S. B. Mitchell, C. Piatko, and E. M. Arkin. Computing a shortest k-link path in a polygon. In *Proc. 33rd Annu. IEEE Sympos. Found. Comput. Sci.*, pages 573–582, 1992.

[109] J. S. B. Mitchell, G. Rote, and G. Woeginger. Minimum-link paths among obstacles in the plane. *Algorithmica*, 8:431–459, 1992.

[110] J. S. B. Mitchell and S. Suri. Geometric algorithms. In M. O. Ball, T. L. Magnanti, C. L. Monma, and G. L. Nemhauser, editors, *Network Models*, Handbook of Operations Research/Management Science, pages 425–479. Elsevier Science, Amsterdam, 1995.

[111] J. S. B. Mitchell and S. Suri. Separation and approximation of polyhedral objects. volume 5, pages 95–114, 1995.

[112] Joseph S. B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions: A simple new method for the geometric k-MST problem. In *Proc. 7th ACM-SIAM Sympos. Discrete Algorithms*, pages 402–408, 1996.

[113] Joseph S. B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions: Part II – A simple polynomial-time approximation scheme for geometric k-MST, TSP, and related problems. Manuscript, University at Stony Brook, 1996.

[114] D. M. Mount. The number of shortest paths on the surface of a polyhedron. *SIAM J. Comput.*, 19:593–611, 1990.

[115] B. J. Nilsson. *Guarding Art Galleries — Methods for Mobile Guards*. PhD thesis, Lund University, 1995.

[116] B. J. Nilsson and S. Schuierer. An optimal algorithm for the rectilinear link center of a rectilinear polygon. *Comput. Geom. Theory Appl.*, 6:to appear, 1996.

[117] Bengt J. Nilsson and Sven Schuierer. Computing the rectilinear link diameter of a polygon. In *Computational Geometry — Methods, Algorithms and Applications: Proc. Internat. Workshop Comput. Geom. CG '91*, volume 553 of *Lecture Notes in Computer Science*, pages 203–215. Springer-Verlag, 1991.

[118] J. O'Rourke. Finding a shortest ladder path: a special case. IMA Preprint Series 353, Inst. Math. Appl., Univ. Minnesota, Minneapolis, MN, 1987.

[119] C. H. Papadimitriou. The Euclidean traveling salesman problem is NP-complete. *Theoret. Comput. Sci.*, 4:237–244, 1977.

[120] C. H. Papadimitriou. An algorithm for shortest-path motion in three dimensions. *Inform. Process. Lett.*, 20:259–263, 1985.

[121] C. H. Papadimitriou and E. B. Silverberg. Optimal piecewise linear motion of an object among obstacles. *Algorithmica*, 2:523–539, 1987.

[122] Evanthia Papadopoulou and D. T. Lee. Efficient computation of the geodesic voronoi diagram of points in a simple polygon. In ??, editor, *Algorithms – ESA'95*, LNCS ??, page ??, ??, September 1995.

[123] M. Pocchiola and G. Vegter. Computing the visibility graph via pseudo-triangulations. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 248–257, 1995.

[124] R. Pollack, M. Sharir, and G. Rote. Computing of the geodesic center of a simple polygon. *Discrete Comput. Geom.*, 4:611–626, 1989.

[125] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, New York, NY, 1985.

[126] G. Reinelt. Fast heuristics for large geometric traveling salesman problems. *ORSA J. Comput.*, 4:206–217, 1992.

[127] Stéphane Rivière. Topologically sweeping the visibility complex of polygonal scenes. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages C36–C37, 1995.

[128] C. Schevon and J. O'Rourke. The number of maximal edge sequences on a convex polytope. In *Proc. 26th Allerton Conf. Commun. Control Comput.*, pages 49–57, University of Illinois at Urbana-Champaign, October 1988.

[129] J. Sellen. Direction weighted shortest path planning. In *Proc. IEEE Internat. Conf. Robot. Autom.*, pages 1970–1975, Nagoya, 1995.

[130] M. Sharir. On shortest paths amidst convex polyhedra. *SIAM J. Comput.*, 16:561–572, 1987.

[131] M. Sharir. A note on the Papadimitriou-Silverberg algorithm for planning optimal piecewise linear motion of a ladder. *Inform. Process. Lett.*, 32:187–190, 1989.

[132] J. M. Smith and P. Winter. Computational geometry and topological network design. In D.-Z. Du and F. K. Hwang, editors, *Computing in Euclidean Geometry*, volume 1 of *Lecture Notes Series on Computing*, pages 287–385. World Scientific, Singapore, 1992.

[133] S. Suri. A linear time algorithm for minimum link paths inside a simple polygon. *Comput. Vision Graph. Image Process.*, 35:99–110, 1986.

[134] S. Suri. *Minimum link paths in polygons and related problems.* Ph.D. thesis, Dept. Comput. Sci., Johns Hopkins Univ., Baltimore, MD, 1987.

[135] S. Suri. On some link distance problems in a simple polygon. *IEEE Trans. Robot. Autom.*, 6:108–113, 1990.

[136] Xuehou Tan and Tomio Hirata. Constructing shortest watchman routes by divide-and-conquer. In *Proc. 4th Annu. Internat. Sympos. Algorithms Comput. (ISAAC 93)*, volume 762 of *Lecture Notes in Computer Science*, pages 68–77. Springer-Verlag, 1993.

[137] H. Wang and P. K. Agarwal. Approximation algorithms for curvature constrained shortest paths. In *Proc. 7th ACM-SIAM Sympos. Discrete Algorithms*, pages 409–418, 1996.

[138] C. D. Yang, D. T. Lee, and C. K. Wong. On bends and lengths of rectilinear paths: a graph theoretic approach. *Internat. J. Comput. Geom. Appl.*, 2(1):61–74, 1992.

[139] C. D. Yang, D. T. Lee, and C. K. Wong. Rectilinear paths problems among rectilinear obstacles revisited. Technical Report, Northwestern University, 1992.