40 ALGORITHMIC MOTION PLANNING

Micha Sharir

INTRODUCTION

Motion planning is a fundamental problem in robotics. It comes in a variety of forms, but the simplest version is as follows. We are given a robot system B, which may consist of several rigid objects attached to each other through various joints, hinges, and links, or moving independently, and a two-dimensional or three-dimensional environment V cluttered with obstacles. We assume that the shape and location of the obstacles and the shape of B are known to the planning system. Given an initial placement Z_1 and a final placement Z_2 of B, we wish to determine whether there exists a collision-avoiding motion of B from Z_1 to Z_2, and, if so, to plan such a motion. In this simplified and purely geometric setup, we ignore issues such as incomplete information, nonholonomic constraints, control issues related to inaccuracies in sensing and motion, nonstationary obstacles, optimality of the planned motion, and so on.

Since the early 1980's, motion planning has been an intensive area of study in robotics and computational geometry. In this chapter we will focus on *algorithmic motion planning*, emphasizing theoretical algorithmic analysis of the problem and seeking worst-case asymptotic bounds, and only mention briefly practical heuristic approaches to the problem. The majority of this chapter is devoted to the simplified version of motion planning, as stated above. Section 40.1 presents general techniques and lower bounds. Section 40.2 considers efficient solutions to a variety of specific moving systems with a small number of degrees of freedom. These efficient solutions exploit various sophisticated methods in computational and combinatorial geometry related to arrangements of curves and surfaces (Chapter 21). Section 40.3 then briefly discusses various extensions of the motion planning problem, incorporating uncertainty, moving obstacles, etc. We conclude in Section 40.4 with a brief review of Davenport-Schinzel sequences, a combinatorial structure that plays an important role in many motion planning algorithms.

40.1 GENERAL TECHNIQUES AND LOWER BOUNDS

GLOSSARY

Robot B: A mechanical system consisting of one or more rigid bodies, possibly connected by various joints and hinges.

0-8493-8524-5/97/$0.00+$.50

Physical space: The two- or three-dimensional environment in which the robot moves.

Placement: The portion of physical space occupied by the robot at some instant.

Degrees of freedom k: The number of real parameters that determine the robot B's placements. Each placement can be represented as a point in $\mathbb{R}^k$.

Free placement: A placement at which the robot is disjoint from the obstacles.

Semifree placement: A placement at which the robot does not meet the interior of any obstacle (but may be in contact with some obstacles).

Configuration space $\mathcal{C}$: A portion of k-space (where k is the number of degrees of freedom of B) that represents all possible robot placements; the coordinates of any point in this space specify the corresponding placement.

Expanded obstacle / C-space obstacle / forbidden region: For an obstacle O, this is the portion O^* of configuration space consisting of placements at which the robot intersects (collides with) O.

Free configuration space $\mathcal{F}$: The subset of configuration space consisting of free placements of the robot: $\mathcal{F} = \mathcal{C} \setminus \bigcup_O O^*$. (In the literature, this usually also includes semifree placements.)

Contact surface: For an obstacle feature a (corner, edge, face, etc.) and for a feature b of the robot, this is the locus in $\mathcal{C}$ of placements at which a and b are in contact with each other. In most applications, these surfaces are semialgebraic sets of constant description complexity (see definitions below).

Collision-free motion of B: A path contained in $\mathcal{F}$. Any two placements of B that can be reached from each other via a collision-free path must lie in the same (arcwise-)connected component of $\mathcal{F}$.

Arrangement $\mathcal{A}(\Sigma)$: The decomposition of k-space into cells of various dimensions, induced by a collection Σ of surfaces in $\mathbb{R}^k$. Each cell is a maximal connected portion of the intersection of some fixed subcollection of surfaces that does not meet any other surface. See Chapter 21. Since a collision-free motion should not cross any contact surface, $\mathcal{F}$ is the union of some of the cells of $\mathcal{A}(\Sigma)$, where Σ is the collection of contact surfaces.

Semialgebraic set: A subset of $\mathbb{R}^k$ defined by a Boolean combination of polynomial equalities and inequalities in the k coordinates. See Section 29.2.

Constant description complexity: Said of a semialgebraic set if it is defined by a constant number of polynomial equalities and inequalities of constant maximum degree (where the number of variables is also assumed to be constant).

Example. Let B be a rigid polygon with k edges, moving in a planar polygonal environment V with n edges. The system has three degrees of freedom, (x, y, θ), where (x, y) are the coordinates of some reference point on B, and θ is the orientation of B. Each contact surface is the locus of placements where some vertex of B touches some edge of V, or some edge of B touches some vertex of V. There are $2kn$ contact surfaces, and if we replace θ by $\tan \frac{\theta}{2}$, then each contact surface becomes a portion of some algebraic surface of degree at most 4, bounded by a constant number of algebraic arcs, each of degree at most 2.

40.1.1 GENERAL SOLUTIONS

GLOSSARY

Cylindrical algebraic decomposition of $\mathcal{F}$: A recursive decomposition of $\mathcal{F}$ into cylindrical-like cells originally proposed by Collins. Over each cell of the decomposition, each of the polynomials involved in the definition of $\mathcal{F}$ has a fixed sign (positive, negative, or zero), implying that $\mathcal{F}$ is the union of some of the cells of this decomposition. See Section 29.5 for further details.

Connectivity graph: A graph whose nodes are the (free) cells of a decomposition of $\mathcal{F}$ and whose arcs connect pairs of adjacent cells.

Roadmap $\mathcal{R}$: A network of 1-dimensional curves within $\mathcal{F}$, having the properties that (i) it *preserves the connectivity* of $\mathcal{F}$, in the sense that the portion of $\mathcal{R}$ within each connected component of $\mathcal{F}$ is (nonempty and) connected; and (ii) it is ***reachable***, in the sense that there is a simple procedure to move from any free placement of the robot to a placement on $\mathcal{R}$; we denote the mapping resulting from this procedure by $\phi_{\mathcal{R}}$.

Retraction of $\mathcal{F}$ onto $\mathcal{R}$: A continuous mapping of $\mathcal{F}$ onto $\mathcal{R}$ that is the identity on $\mathcal{R}$. The roadmap mapping $\phi_{\mathcal{R}}$ is usually a retraction. When this is the case, we note that for any path ψ within $\mathcal{F}$, represented as a continuous mapping $\psi : [0,1] \mapsto \mathcal{F}$, $\phi_{\mathcal{R}} \circ \psi$ is a path within $\mathcal{R}$, and, concatenating to it the motions from $\psi(0)$ and $\psi(1)$ to $\mathcal{R}$, we see that there is a collision-free motion of B between two placements Z_1, Z_2 iff there is a path within $\mathcal{R}$ between $\phi_{\mathcal{R}}(Z_1)$ and $\phi_{\mathcal{R}}(Z_2)$.

Silhouette: The set of critical points of a mapping; see Section 29.6.

CELL DECOMPOSITION

$\mathcal{F}$ is a semialgebraic set in $\mathbb{R}^k$. Applying Collins's cylindrical algebraic decomposition results in a collection of cells whose total complexity is $O((nd)^{3^k})$, where d is the maximum algebraic degree of the polynomials defining the contact surfaces; the decomposition can be constructed within a similar time bound. If the coordinate axes are generic, then we can also compute all pairs of cells of $\mathcal{F}$ that are ***adjacent*** to each other (i.e., cells whose closures (within $\mathcal{F}$) overlap), and store this information in the form of a connectivity graph. It is then easy to search for a collision-free path through this graph, if one exists, between the (cell containing the) initial robot placement and the (cell containing the) final placement. This leads to a doubly-exponential general solution for the motion planning problem:

THEOREM 40.1.1 *Cylindrical Cell Decomposition* [SS83]

Any motion planning problem, with k degrees of freedom, for which the contact surfaces are defined by a total of n polynomials of maximum degree d, can be solved by Collins's cylindrical algebraic decomposition, in randomized expected time $O((nd)^{3^k})$.

(The randomization is needed only to choose a generic direction for the coordinate axes.)

ROADMAPS

A more recent and improved solution is given in [Can87, BPR96] based on the notion of a *roadmap* $\mathcal{R}$, a network of 1-dimensional curves within (the closure of) $\mathcal{F}$, having properties defined in the glossary above. Once such a roadmap $\mathcal{R}$ has been constructed, any motion planning instance reduces to path searching within $\mathcal{R}$, which is easy to do. $\mathcal{R}$ is constructed recursively, as follows. One projects $\mathcal{F}$ onto some generic 2-plane, and computes the silhouette of $\mathcal{F}$ under this projection. Next, the critical values of the projection of the silhouette on some line are found, and a roadmap is constructed recursively within each slice of $\mathcal{F}$ at each of these critical values. The resulting "sub-roadmaps" are then merged with the silhouette, to obtain the desired $\mathcal{R}$.

The original algorithm of Canny relies heavily on the polynomials defining $\mathcal{F}$ being in general position, and on the availability of a generic plane of projection. This algorithm runs in $n^k(\log n)d^{O(k^4)}$ deterministic time, and in $n^k(\log n)d^{O(k^2)}$ expected randomized time. Recent work [BPR96] addresses and overcomes the general position issue, and produces a roadmap for any semialgebraic set; the running time of this solution is $n^{k+1}d^{O(k^2)}$.

If we ignore the dependence on the degree d, the algorithm of Canny is close to optimal in the worst case, assuming that some representation of the entire $\mathcal{F}$ has to be output, since there are easy examples where the free configuration space consists of $\Omega(n^k)$ connected components.

THEOREM 40.1.2 *Roadmap Algorithm* [Can87]

Any motion planning problem, as in the preceding theorem, can be solved by the roadmap technique in $n^k(\log n)d^{O(k^4)}$ deterministic time, and in $n^k(\log n)d^{O(k^2)}$ expected randomized time.

40.1.2 LOWER BOUNDS

Both general solutions are (at least) exponential in k (but are polynomial in the other parameters when k is fixed). This raises the problem of calibrating the complexity of the problem when k can be arbitrarily large.

THEOREM 40.1.3 *Lower Bounds*

The motion planning problem, with arbitrarily many degrees of freedom, is PSPACE-hard for the instances of: (a) coordinated motion of many rectangular boxes along a rectangular floor; (b) motion planning of a planar mechanical linkage with many links; and (c) motion planning for a multi-arm robot in a 3-dimensional polyhedral environment.

All these results appear in papers collected in [HSS87]. There are also many NP-hardness results for other systems.

Facing these findings, we can either approach the general problem with heuristic and approximate schemes, or attack specific problems with small values of k, with the goal of obtaining solutions better than those yielded by the general techniques. We will mostly survey here the latter approach, and mention towards the end what has been achieved by the first approach.

40.2 MOTION PLANNING WITH A SMALL NUMBER OF DEGREES OF FREEDOM

In this main section of the chapter, we review solutions to a variety of specific motion planning problems, most of which have 2 or 3 degrees of freedom. Exploiting the special structure of these problems leads to solutions that are more efficient than the general methods described above.

GLOSSARY

Jordan arc/curve: The image of the closed unit interval under a continuous bijective mapping into the plane. A closed Jordan curve is the image of the unit circle under a similar mapping, and an unbounded Jordan curve is an image of the open unit interval (or of the entire real line) that separates the plane.

Randomized algorithm: An algorithm that applies internal randomization ("coin-flips"). We consider here algorithms that always terminate, and produce the correct output, but whose running time is a random variable that depends on the internal coin-flips. We will state upper bounds on the expectation of the running time (the ***randomized expected time***) of such an algorithm, which hold for any input. See Chapter 34.

Minkowski sum: For two planar (or spatial) sets A and B, their Minkowski sum, or pointwise vector addition, is the set $A \oplus B = \{x + y \mid x \in A, y \in B\}$.

General position: The input to a geometric problem is said to be in general position if no nontrivial algebraic identity with integer coefficients holds among the parameters that specify the input (assuming the input is not overspecified). For example: no three input points should be collinear, no four points cocircular, no three lines concurrent, etc.

Convex distance function: A convex region B that contains the origin in its interior induces a convex distance function d_B defined by

$$d_B(p,q) = \min\{\lambda \mid q \in p \oplus \lambda B\}.$$

B-Voronoi diagram: For a set S of sites, and a convex region B as above, the B-Voronoi diagram $\mathrm{Vor}_B(S)$ of S is a decomposition of space into Voronoi cells $V(s)$, for $s \in S$, such that

$$V(s) = \{p \mid d_B(p,s) \le d_B(p,s') \text{ for all } s' \in \mathcal{S}\}.$$

Here $d_B(p,s) = \min_{q \in s} d_B(p,q)$.

$\alpha(n)$: The extremely slowly-growing inverse Ackermann function; see Section 40.4.

Contact segment: The locus of semifree placements of a polygon B translating in the plane, at each of which either some specific vertex of B touches some specific obstacle edge, or vice-versa.

Contact curve: A generalization of "contact segment" to the locus of semifree placements of B, assuming that B has only two degrees of freedom, where some specific feature of B makes contact with some specific obstacle feature.

40.2.1 TWO DEGREES OF FREEDOM

A TRANSLATING POLYGON IN 2D

This is a system with two degrees of freedom (translations in the x and y directions).

A CONVEX POLYGON

Suppose first the translating polygon B is a *convex* k-gon, and there are m convex polygonal obstacles, $A_1, \ldots, A_m$, with pairwise disjoint interiors, having a total of n edges. The region of configuration space where B collides with A_i is the *Minkowski sum*

$$K_i = A_i \oplus (-B) = \{x - y \mid x \in A_i,\ y \in B\}.$$

The free configuration space is the complement of $\bigcup_{i=1}^{m} K_i$. Assuming general position, one can show:

THEOREM 40.2.1 [KLPS86]

(a) *Each K_i is a convex polygon, with $n_i + k$ edges, where n_i is the number of edges of A_i.*

(b) *For each $i \neq j$, the boundaries of K_i and K_j intersect in at most two points. (This also holds when the A_i's and B are not polygons.)*

(c) *Given a collection of planar regions $K_1, \ldots, K_m$, each enclosed by a closed Jordan curve, such that any pair of the bounding curves intersects at most twice, then the boundary of the union $\bigcup_{i=1}^{m} K_i$ consists of at most $6m - 12$ maximal connected portions of the boundaries of the K_i's, provided $m \geq 3$, and this bound is tight in the worst case.*

These properties, combined with several algorithmic techniques, imply:

THEOREM 40.2.2

(a) *The free configuration space for a translating convex polygon, as above, is a polygonal region with at most $6m - 12$ convex vertices and $N = \sum_{i=1}^{m}(n_i + k) = n + km$ nonconvex vertices.*

(b) *$\mathcal{F}$ can be computed in deterministic time $O(N \log^2 n)$, or in randomized expected time $O(N \cdot 2^{\alpha(n)} \log n)$.*

AN ARBITRARY POLYGON

Suppose next that B is an arbitrary polygonal region with k edges. Let A be the union of all obstacles, which is another polygonal region with n edges. As above, the free configuration space is the complement of the Minkowski sum

$$K = A \oplus (-B) = \{x - y \mid x \in A,\ y \in B\}.$$

K is again a polygonal region, but, in this case, its maximum possible complexity is $\Theta(k^2n^2)$, so computing it might be considerably more expensive than in the convex case.

A single face suffices. If the initial placement Z of B is given, then we do not have to compute the entire (complement of) K; it suffices to compute the connected component f of the complement of K that contains Z, because no other placement is reachable from Z via a collision-free motion.

Let Σ be the collection of all contact segments; there are $2kn$ such segments. The desired component f is the face of $\mathcal{A}(\Sigma)$ that contains Z. Using the theory of *Davenport-Schinzel sequences* (Section 40.4), one can show that the maximum possible combinatorial complexity of a single face in a two-dimensional arrangement of N segments is $\Theta(N\alpha(N))$. A more careful analysis [HCA$^+$95] shows:

THEOREM 40.2.3

(a) The maximum combinatorial complexity of a single face in the arrangement of contact segments for the case of an arbitrary translating polygon is $\Theta(kn\alpha(k))$ (this improvement is significant only when $k \ll n$).

(b) Such a face can be computed in deterministic time $O(kn\log^2 n)$, or in randomized expected time $O(kn \cdot 2^{\alpha(n)} \log n)$.

VORONOI DIAGRAMS

Another approach to motion planning for a translating *convex* object B, is via generalized *Voronoi diagrams* (see Chapter 20), based on the convex distance function $d_B(p,q)$. This function effectively places B centered at p and expands it until it hits q. The scaling factor at this moment is the d_B-distance from p to q (if B is a unit disk, d_B is the Euclidean distance). d_B satisfies the triangle inequality, and is thus “almost” a metric, except that it is not symmetric in general; it is symmetric iff B is centrally symmetric with respect to the point of reference.

Using this distance function d_B, a *B-Voronoi diagram* $\mathrm{Vor}_B(\mathcal{S})$ of $\mathcal{S}$ may be defined for a set $\mathcal{S}$ of m pairwise disjoint obstacles.

THEOREM 40.2.4

Assuming that each of B and the obstacles in $\mathcal{S}$ has constant description complexity, and that they are in general position, the B-Voronoi diagram has $O(m)$ complexity, and can be computed in $O(m\log m)$ time (in an appropriate model of computation). If B and the obstacles are convex polygons, as above, then the complexity of $\mathrm{Vor}_B(\mathcal{S})$ is $O(N)$ and it can be computed in time $O(N\log m)$.

One can show that if Z_1 and Z_2 are two free placements of B, then there exists a collision-free motion from Z_1 to Z_2 if and only if there exists a collision-free motion of B where its center moves only along the edges of $\mathrm{Vor}_B(\mathcal{S})$, between two corresponding placements W_1, W_2, where W_i, for $i = 1, 2$, is the placement obtained by pushing B from the placement Z_i away from its d_B-nearest obstacle, until it becomes equally nearest to two or more obstacles (so that its center lies on an edge of $\mathrm{Vor}_B(\mathcal{S})$).

Thus motion planning of B reduces to a path-searching in the 1-dimensional network of edges of $\mathrm{Vor}_B(\mathcal{S})$. This technique is called the ***retraction technique***, and can be regarded as a special case of the general roadmap algorithm. The resulting motions have "high clearance," and so are safer than arbitrary motions, because they stay equally nearest to at least two obstacles.

THEOREM 40.2.5

The motion planning problem for a convex object B translating amidst m convex and pairwise disjoint obstacles can be solved in $O(m \log m)$ time, by constructing and searching in the B-Voronoi diagram of the obstacles, assuming that B and the obstacles have constant description complexity each. If B and the obstacles are convex polygons, then the same technique yields an $O(N \log m)$ solution, where $N = n + km$ is as above.

THE GENERAL MOTION PLANNING PROBLEM WITH TWO DEGREES OF FREEDOM

If B is any system with two degrees of freedom, its configuration space is 2-dimensional, and, for simplicity, let us think of it as the plane (spaces that are topologically more complex can be decomposed into a constant number of "planar" patches). We construct a collection Σ of contact curves, which, under reasonable assumptions concerning B and the obstacles, are each an algebraic Jordan arc or curve of some fixed maximum degree b. In particular, each pair of contact curves will intersect in at most some constant number, $s \leq b^2$, of points.

As above, it suffices to compute the single face of $\mathcal{A}(\Sigma)$ that contains the initial placement of B. The theory of Davenport-Schinzel sequences implies that the complexity of such a face is $O(\lambda_{s+2}(n))$, where $\lambda_{s+2}(n)$ is the maximum length of an $(n, s+2)$-Davenport-Schinzel sequence (Section 40.4), which is slightly super-linear in n when s is fixed.

The face in question can be computed in deterministic time $O(\lambda_{s+2}(n) \log^2 n)$, using a fairly involved divide-and-conquer technique based on line-sweeping; see Section 21.5. (Some slight improvements in the running time have been obtained recently.) Using randomized incremental (or divide-and-conquer) techniques, the face can be computed in randomized expected $O(\lambda_{s+2}(n) \log n)$ time.

THEOREM 40.2.6

Under the above assumptions, the general motion planning problem for systems with two degrees of freedom can be solved in deterministic time $O(\lambda_{s+2}(n) \log^2 n)$, or in $O(\lambda_{s+2}(n) \log n)$ randomized expected time.

40.2.2 THREE DEGREES OF FREEDOM

A ROD IN A PLANAR POLYGONAL ENVIRONMENT

We next pass to systems with three degrees of freedom. Perhaps the simplest instance of such a system is the case of a line segment B ("rod," "ladder," "pipe") moving (translating and rotating) in a planar polygonal environment with n edges. The maximum combinatorial complexity of the free configuration space $\mathcal{F}$ of B is $\Theta(n^2)$ (recall that the naive bound for systems with three degrees of freedom is $O(n^3)$). A cell-decomposition representation of $\mathcal{F}$ can be constructed in (deterministic) $O(n^2 \log n)$ time [LS87b]. Several alternative near-quadratic algorithms have also been developed, including one based on constructing a Voronoi diagram in $\mathcal{F}$ [OSY87].

An $\Omega(n^2)$ lower bound for this problem has been established in [KO88]. It exhibits a polygonal environment with n edges and two free placements of B that are reachable from each other. However, any free motion between them requires $\Omega(n^2)$ "elementary moves," that is, the specification of any such motion requires $\Omega(n^2)$ complexity. This is a fairly strong lower bound, since it does not rely on lower bounding the complexity of the free configuration space (or of a single connected component thereof); after all, it is not clear why a motion planning algorithm should have to produce a full description of the whole free space (or of a single component).

THEOREM 40.2.7

Motion planning for a rod moving in a polygonal environment bounded by n edges can be performed in $O(n^2 \log n)$ time. There are instances where any collision-free motion of the rod between two specified placements requires $\Omega(n^2)$ "elementary moves."

A CONVEX POLYGON IN A PLANAR POLYGONAL ENVIRONMENT

Here B is a convex k-gon, free to move (translate and rotate) in an arbitrary polygonal environment bounded by n edges. The free configuration space is 3-dimensional, and there are at most $2kn$ contact surfaces, of maximum degree 4. The naive bound on the complexity of $\mathcal{F}$ is $O((kn)^3)$ (attained if B is nonconvex), but, using Davenport-Schinzel sequences, one can show that the complexity of $\mathcal{F}$ is only $O(kn\lambda_6(kn))$. Geometrically, a vertex of $\mathcal{F}$ is a semifree placement of B at which it makes simultaneously three obstacle contacts. The above bound implies that the number of such *critical placements* is only slightly super-quadratic (and not cubic) in kn.

Computing $\mathcal{F}$ in time close to this bound has proven more difficult, and only recently has a complete solution, running in $O(kn\lambda_6(kn) \log kn)$ time and constructing the entire $\mathcal{F}$, been attained [AAAS96].

Another approach was given in [CK93]. It computes the Delaunay triangulation of the obstacles under the distance function d_B, when the orientation of B is fixed, and then traces the discrete combinatorial changes in the diagram as the orientation varies. The number of changes was shown to be $O(k^4 n\lambda_3(n))$. Using this structure, the algorithm of [CK93] produces a high-clearance motion of B between any two specified placements, in time $O(k^4 n\lambda_3(n) \log n)$.

Since all these algorithms are fairly complicated, one might consider in practice an alternative approximate scheme, proposed in [AFK$^+$90]. This scheme discretizes the orientation of B, solves the translational motion planning for B at each of the discrete orientations, and finds those placements of B at which it can rotate (without translating) between two successive orientations. This scheme works very well in practice.

THEOREM 40.2.8

Motion planning for a k-sided convex polygon, translating and rotating in a planar *polygonal environment bounded by n edges, can be performed in $O(kn\lambda_6(kn)\log kn)$ or $O(k^4 n\lambda_3(n)\log n)$ time.*

EXTREMAL PLACEMENTS

A related problem is to find the largest free placement of B in the given polygonal environment. This has applications in manufacturing, where one wants to cut out copies of B that are as large as possible from a sheet of some material.

If only translations are allowed, the B-Voronoi diagram can be used to find the largest free homothetic copy of B. If general rigid motions are allowed, the technique of [CK93] computes the largest free similar copy of B in time $O(k^4 n\lambda_3(n)\log n)$. An alternative technique is given in [AAAS96], with randomized expected running time $O(kn\lambda_6(kn)\log^4 kn)$. Both bounds are nearly quadratic in n.

Finally, we mention the special case where the polygonal environment is the interior of a convex n-gon. This is simpler to analyze. The number of free critical placements of (similar copies of) B, at which B makes simultaneously four obstacle contacts, is $O(kn^2)$ [AAAS96], and they can all be computed in $O(kn^2 \log n)$ time.

THEOREM 40.2.9

The largest similar placement of a k-sided convex polygon in a planar polygonal environment bounded by n edges can be computed in randomized expected time $O(kn\lambda_6(kn)\log^4 kn)$ or in deterministic time $O(k^4 n\lambda_3(n)\log n)$. When the environment is the interior of an n-sided convex polygon, the running time improves to $O(kn^2 \log n)$.

A NONCONVEX POLYGON

Next we consider the case where B is an arbitrary polygonal region (not necessarily connected), translating and rotating in a polygonal environment bounded by n edges, as above. Here one can show that the maximum complexity of $\mathcal{F}$ is $\Theta((kn)^3)$. Using standard techniques, $\mathcal{F}$ can be constructed in $\Theta((kn)^3 \log kn)$ time, an algorithm which has been implemented. However, as in the purely translational case, it suffices to construct the connected component of $\mathcal{F}$ containing the initial placement of B. The general result, stated below, for systems with three degrees of freedom, implies that the complexity of such a component is only near-quadratic in kn. An algorithm that computes the component in time $O((kn)^{2+\epsilon})$ is given in [HS96].

THEOREM 40.2.10

Motion planning for an arbitrary k-sided polygon, translating and rotating in a planar polygonal environment bounded by n edges, can be performed in time $O((kn)^{2+\epsilon})$, for any $\epsilon > 0$.

A TRANSLATING POLYTOPE IN A 3-D POLYHEDRAL ENVIRONMENT

Another interesting motion planning problem with three degrees of freedom involves a polytope B, with a total of k vertices, edges, and facets, translating amidst polyhedral obstacles in $\mathbb{R}^3$, with a total of n vertices, edges, and faces. The contact surfaces in this case are planar polygons, composed of a total of $O(kn)$ triangles in 3-space.

Without additional assumptions, the complexity of $\mathcal{F}$ can be $\Theta((kn)^3)$ in the worst case. However, the complexity of a single component is only $O((kn)^2 \log kn)$. Such a component can be constructed in $O((kn)^{2+\epsilon})$ time, for any $\epsilon > 0$ [AS94].

If B is a convex polytope, and the obstacles consist of m convex polyhedra, with pairwise disjoint interiors and with a total of n faces, the complexity of the entire $\mathcal{F}$ is $O(kmn \log m)$ and it can be constructed in $O(kmn \log^2 m)$ time [AS].

THEOREM 40.2.11

Translational motion planning for an arbitrary polytope with k facets, in an arbitrary 3-dimensional polyhedral environment bounded by n facets, can be performed in time $O((kn)^{2+\epsilon})$, for any $\epsilon > 0$. If B is a convex polytope, and there are m convex pairwise disjoint obstacles with a total of n facets, then the motion planning can be performed in $O(kmn \log^2 m)$ time.

THE GENERAL MOTION PLANNING PROBLEM WITH THREE DEGREES OF FREEDOM

The last several instances were special cases of the general motion planning problem with three degrees of freedom. In abstract terms, we have a collection Σ of N contact surfaces in $\mathbb{R}^3$, where these surfaces are assumed to be (patches of) algebraic surfaces of constant maximum degree. The free configuration space consists of some cells of the arrangement $\mathcal{A}(\Sigma)$, and a single connected component of $\mathcal{F}$ is just a single cell in that arrangement.

Inspecting the preceding cases, a unifying observation is that while the maximum complexity of the entire $\mathcal{F}$ can be $\Theta(N^3)$, the complexity of a single component is invariably only near-quadratic in N. This was recently shown in [HS95a] to hold in general: the combinatorial complexity of a single cell of $\mathcal{A}(\Sigma)$ is $O(N^{2+\epsilon})$, for any $\epsilon > 0$, where the constant of proportionality depends on ϵ and on the maximum degree of the surfaces; cf. Section 21.5.

A general-purpose algorithm for computing a single cell in such an arrangement was recently given in [SS96]. It runs in randomized expected time $O(N^{2+\epsilon})$, for any $\epsilon > 0$, and is based on *vertical decompositions* in such arrangements (see Section 21.3.2).

THEOREM 40.2.12

An arbitrary motion planning problem with three degrees of freedom, involving N contact surface patches, each of constant description complexity, can be solved in time $O(N^{2+\epsilon})$, for any $\epsilon > 0$.

40.2.3 OTHER PROBLEMS WITH FEW DEGREES OF FREEDOM

COORDINATED MOTION PLANNING

Another class of motion planning problems involves coordinated motion planning of several independently moving systems. Conceptually, this situation can be handled as just another special case of the general problem: Consider all the moving objects as a single system, with $k = \sum_{i=1}^{t} k_i$ degrees of freedom, where t is the number of moving objects, and k_i is the number of degrees of freedom of the ith object. However, k will generally be too large, and the problem then will be more difficult to tackle.

A better approach is as follows [SS91]. Let $B_1, \ldots, B_t$ be the given independent objects. For each $i = 1, \ldots, t$, construct the free configuration space $\mathcal{F}^{(i)}$ for B_i alone (ignoring the presence of all other moving objects). The actual free configuration space $\mathcal{F}$ is a subset of $\prod_{i=1}^{t} \mathcal{F}^{(i)}$. Suppose we have managed to decompose each $\mathcal{F}^{(i)}$ into subcells of constant description complexity. Then $\mathcal{F}$ is a subset of the union of Cartesian products of the form $c_1 \times c_2 \times \cdots \times c_t$, where c_i is a subcell of $\mathcal{F}^{(i)}$.

We next compute the portion of $\mathcal{F}$ within each such product. Each such subproblem can be intuitively interpreted as the coordinated motion planning of our objects, where each moves within a small portion of space, amidst only a constant number of nearby obstacles; so these subproblems are much easier to solve. Moreover, in typical cases, for most products $P = c_1 \times c_2 \times \cdots \times c_t$ the problem is trivial, because P represents situations where the moving objects are far from one another, and so cannot interact at all, meaning that $\mathcal{F} \cap P = P$. The number of subproblems that really need to be solved will be relatively small.

The connectivity graph that represents $\mathcal{F}$ is also relatively easy to construct. Its nodes are the connected components of the intersections of $\mathcal{F}$ with each of the above cell products P, and two nodes are connected to each other if they are adjacent in the overall $\mathcal{F}$. In many typical cases, determining this adjacency is easy.

As an example, one can apply this technique to the coordinated motion planning of k disks moving in a planar polygonal environment bounded by n edges, to get a solution with $O(n^k)$ running time. Since this problem has $2k$ degrees of freedom, this is a significant improvement over the bound $O(n^{2k} \log n)$ yielded by Canny's general algorithm.

TABLE 40.2.1 Summary of motion planning algorithms.

SYSTEM	MOTION	ENVIRONMENT	df	RUNNING TIME
Convex k-gon	translation	planar polygonal	2	$O(N \log m)$
Arbitrary k-gon	translation	planar polygonal	2	$O(kn \log^2 n)$
General			2	$O(\lambda_{s+2}(n) \log^2 n)$
Line segment	trans & rot	planar polygonal	3	$O(n^2 \log n)$
Convex k-gon	trans & rot	planar polygonal	3	$O(k^4 n \lambda_3(n) \log n)$
				$O(kn\lambda_6(kn) \log n)$
Arbitrary k-gon	trans & rot	planar polygonal	3	$O((kn)^{2+\epsilon})$
Convex polytope	translation	3-d polyhedral	3	$O(kmn \log^2 m)$
Arbitrary polytope	translation	3-d polyhedral	3	$O((kn)^{2+\epsilon})$
General			3	$O(N^{2+\epsilon})$

MOTION PLANNING AND ARRANGEMENTS

As can be seen from the preceding subsections, motion planning is closely related to the study of arrangements of surfaces in higher dimensions. Motion planning has motivated many problems in arrangements, such as the problem of bounding the complexity of, and designing efficient algorithms for, computing a single cell in an arrangement of n low-degree algebraic surface patches in d dimensions. The goal is to obtain bounds close to $O(n^{d-1})$ for both combinatorial and algorithmic problems. This has been settled satisfactorily for $d = 2, 3$, as noted above, but both problems are still open in higher dimensions. See Chapter 21 for further details.

SUMMARY

Some of the above results are summarized in Table 40.2.1. For each specific system, only one or two algorithms are listed.

40.3 VARIANTS OF THE MOTION PLANNING PROBLEM

We now briefly review several variants of the basic motion planning problem, in which additional constraints are imposed on the problem. Further material on many of these problems can be found in Chapter 41.

OPTIMAL MOTION PLANNING

The preceding section described techniques for determining the existence of a collision-free motion between two given placements of some moving system. It paid no attention to the optimality of the motion, which is an important consideration in practice. There are several problems involved in optimal motion planning. First, optimality is a notion that can be defined in many ways, each of which leads to different algorithmic considerations. Second, optimal motion planning is usually much harder than motion planning per se.

SHORTEST PATHS

The simplest case is when the moving system B is a single point. In this case the cost of the motion is simply the length of the path traversed by the point (normally, we use the Euclidean distance, but other metrics have been considered as well). We thus face the problem of computing ***shortest paths*** amidst obstacles in a two- or three-dimensional environment.

The planar case. Let V be a closed planar polygonal environment bounded by n edges, and let s (the "source") be a point in V. For any other point $t \in V$, let $\pi(s, t)$ denote the (Euclidean) shortest path from s to t within V. Finding $\pi(s, t)$ for any t is facilitated by construction of the *shortest path map* $SPM(s, V)$ from s in V, a decomposition of V into regions detailed in Chapter 24. A very recent result computes $\mathrm{SPM}(s, V)$ in optimal $O(n \log n)$ time.

The same problem may be considered in other metrics. For example, it is easier to give an $O(n \log n)$ algorithm for the shortest path problem under the L_1 or L_∞ metric. See Section 24.3.

The three-dimensional case. Let V be a closed polyhedral environment bounded by a total of n faces, edges, and vertices. Again, given two points $s, t \in V$, we wish to compute the shortest path $\pi(s, t)$ within V from s to t. Here $\pi(s, t)$ is a polygonal path, bending at *edges* (sometimes also at vertices) of V. To compute $\pi(s, t)$, we need to solve two subproblems: to find the sequence of edges (and vertices) of V visited by $\pi(s,t)$ (the ***shortest-path sequence*** from s to t), and to compute the actual points of contact of $\pi(s,t)$ with these edges. These points obey the rule that the incoming angle of $\pi(s,t)$ with an edge is equal to the outgoing angle. Hence, given the shortest-path sequence of length m, we need to solve a system of m quartic equations in m variables in order to find the contact points. This can be solved either approximately, using an iterative scheme, or exactly, using techniques of computational real algebraic geometry; the latter method requires exponential time. Even the first, more "combinatorial," problem of computing the shortest-path sequence is NP-hard [CR87], so the general shortest-path problem is certainly much harder in three dimensions.

Many special cases of this problem, with more efficient solutions, have been studied. See Section 24.5.

VARIOUS OPTIMAL MOTION PLANNING PROBLEMS

Suppose next that the moving system B is a rigid body free only to translate in two or three dimensions. Then the notion of optimality is still well defined—it is the total distance traversed by (any reference point attached to) B. One can then apply the same techniques as above, after replacing the obstacles by their expanded versions. For example, if B is a convex polygon in the plane, and the obstacles are m pairwise openly-disjoint convex polygons $A_1, \ldots, A_m$, then we form the Minkowski sums $K_i = A_i \oplus (-B)$, for $i = 1, \ldots, m$, and compute a shortest path in the complement of their union. Since the K_i's may overlap, we first need to compute their union, as above. A similar approach can be used in planning shortest motion of a polyhedron translating amidst polyhedra in 3-space, etc.

If B admits more complex motions, then the notion of optimality begins to be fuzzy. For example, consider the case of a line segment ("rod") translating and rotating in a planar polygonal environment. One could measure the cost of a motion by the total distance traveled by a designated endpoint (or the centerpoint) of B, or by a weighted average between such a distance and the total turning angle of B, etc. See Section 24.3.

The notion of optimality gets even more complicated when one introduces kinematic constraints on the motion of B. It is then often challenging even without obstacles; see Section 41.5.4. A version of this problem, involving obstacles, has recently been shown to be NP-hard [AKY96].

EXPLORATORY MOTION PLANNING

If the environment in which the robot moves is not known to the system a priori, but the system is equipped with sensory devices, motion planning assumes a more "exploratory" character. If only tactile (or proximity) sensing is available, then a plausible strategy might be to move along a straight line (in physical or configuration space) directly to the target position, and when an obstacle is reached, to follow its boundary until the original straight line of motion is reached again. This technique has been developed and refined for arbitrary systems with two degrees of freedom (see, e.g., [LS87]). It can be shown that this strategy provably reaches the goal, if at all possible, with a reasonable bound on the length of the motion. This technique has been implemented on several real and simulated systems, and has applications to maze-searching problems.

One attempt to extend this technique to a system with three degrees of freedom is given in [CY91]. This technique computes within $\mathcal{F}$ a certain 1-dimensional skeleton (roadmap) $\mathcal{R}$ which captures the connectivity of $\mathcal{F}$. The twist here is that $\mathcal{F}$ is not known in advance, so the construction of $\mathcal{R}$ has to be done in an incremental, exploratory manner. This exploration can be implemented in a controlled manner that does not require too many "probing" steps, and which enables the system to recognize when the construction of $\mathcal{R}$ has been completed (if the goal has not been reached beforehand).

If vision is also available, then other possibilities need to be considered, e.g., the system can obtain partial information about its environment by viewing it from the present placement, and then "explore" it to gain progressively more information until the desired motion can be fully planned. Results of this type can be found in [GMR92] and Section 41.7.

TIME-VARYING ENVIRONMENTS

Interesting generalizations of the motion planning problem arise when some of the obstacles in the robot's environment are assumed to be moving along known trajectories. In this case the robot's goal will be to "dodge" the moving obstacles while moving to its target placement. In this "dynamic" motion planning problem, it is reasonable to assume some limit on the robot's velocity and/or acceleration. Two studies of this problem are [SM88, RS94]. They show that the problem of avoiding moving obstacles is substantially harder than the corresponding static problem. By using time-related configuration changes to encode Turing machine states, they show that the problem is PSPACE-hard even for systems with a small and fixed number of degrees of freedom. However, polynomial-time algorithms are available in a few particularly simple special cases. Another variant of this problem involves movable obstacles, which the robot B can, say, push aside to clear its passage. Again, it can be shown that the general problem of this kind is PSPACE-hard, but that polynomial-time algorithms are available in certain special cases [Wil91].

COMPLIANT MOTION PLANNING

In realistic situations, the moving system has only approximate knowledge of the geometry of the obstacles and/or of its current position and velocity, and it has an inherent amount of error in controlling its motion. The objective is to devise a strategy that will guarantee that the system reaches its goal, where such a strategy usually proceeds through a sequence of free motions (until an obstacle is hit) intermixed with *compliant motions* (sliding along surfaces of contacted obstacles) until it can be ascertained that the goal has been reached.

A standard approach to this problem is through the construction of pre-images (or back projections). See Section 41.5.3.

NONHOLONOMIC MOTION PLANNING

Another realistic constraint on the possible motions of a given system is kinematic (or *kinodynamic*). For example, the moving object B might be constrained not to exceed certain velocity or acceleration thresholds, or has only limited steering capability. Even without any obstacles, such problems are usually quite hard, and the presence of (stationary or moving) obstacles makes them extremely complicated to solve. These so-called *nonholonomic motion planning* problems are usually handled using tools from control theory. See Section 41.5.2.

GENERAL TASK AND ASSEMBLY PLANNING

In task planning problems, the system is given a complex task to perform, such as assembling a part from several components or restructuring its workcell into a new layout, but the precise sequence of substeps needed to attain the final goal is not specified and must be inferred by the system.

Suppose we want to manufacture a product consisting of several parts. Let S be the set of parts in their final assembled form. The first question is whether the product can be disassembled by translating in some fixed direction one part after the other, so that no collision occurs. An order of the parts that satisfies this property is called a ***depth order***. It need not always exist, but when it does, the product can be assembled by translating the constituent parts one after another, in the reverse of the depth order, to their target positions. Products that can be assembled in this manner are called ***stack products*** [WL94]. The simplicity of the assembly process makes stack products attractive to manufacture. Computing a depth order in a given direction (or deciding that no such order exists) can be done in $O(m^{4/3+\epsilon})$ time, for any $\epsilon > 0$, for a set of polygons in 3-space with m vertices in total [dBOS94]. Faster algorithms are known for the special cases of axis-parallel polygons, c-oriented polygons, and "fat" objects.

Many products, however, are not stack products, that is, a single direction in which the parts must be moved is not sufficient to assemble the product. One solution is to search for an assembly sequence that allows a subcollection of parts to be moved as a rigid body in *some* direction. This can be accomplished in polynomial time, though the running time is rather high in the worst case: it may require $\Omega(m^4)$ time for a collection of m tetrahedra in 3-space. A more modest, but considerably more efficient, solution allows each disassembly step to proceed in one of a few given directions [ABHS96]. It has running time $O(m^{4/3+\epsilon})$, for any $\epsilon > 0$. See Section 41.3 for further details on assembly sequencing, and Chapter 46 for related problems.

ON-LINE MOTION PLANNING

Consider the problem of a point robot moving through a planar environment filled with polygonal obstacles, where the robot has no a priori information about the obstacles that lie ahead. One models this situation by assuming that the robot knows the location of the target position and of its own absolute position, but that it only acquires knowledge about the obstacles as it contacts them. The goal is to minimize the distance that the robot travels. See also the discussion on exploratory motion planning above.

Because the robot must make decisions without knowing what lies ahead, it is natural to use the ***competitive ratio*** to evaluate the performance of a strategy.

In particular, one would like to minimize the ratio between the distance traveled by the robot and the length of the shortest start-to-target path in that scene. The competitive ratio is the worst-case ratio achieved over all scenes having a given source-target distance. A special case of interest is when all obstacles are axis-parallel rectangles of width at least 1 located in the infinite Euclidean plane. Natural greedy strategies yield a competitive ratio of $\Theta(n)$, where n is the Euclidean source-target distance. More sophisticated algorithms obtain competitive ratios of $\Theta(\sqrt{n})$ [BRS91]. Randomized algorithms can do much better [BBF$^+$96]. Through the use of randomization, one can translate the case of arbitrary convex obstacles [BRS91] to rectilinearly-aligned rectangles, at the cost of some increase in the competitive ratio. If the scene is not on an infinite plane but rather within some finite rectangular "warehouse," and the start location is one of the warehouse corners, then the competitive ratio drops to $\log n$ [BBFY92].

PRACTICAL APPROACHES TO MOTION PLANNING

When the number of degrees of freedom is even moderately large, exact solutions of the motion planning problem are very inefficient in practice, so one seeks heuristic but practical solutions. Several such techniques have been developed.

Potential field and probabilistic techniques. The first heuristic regards the robot as moving in a potential field induced by the obstacles and by the target placement, where the obstacles act as repulsive barriers, and the target as a strongly attracting source. By letting the robot follow the gradient of such a potential field, we obtain a motion that avoids the obstacles and that can be expected to reach the goal. An attractive feature of this technique is that planning and executing the desired motion are done in a single stage. Another important feature is the generality of the approach; it can easily be applied to systems with many degrees of freedom.

This technique, however, may lead to a motion where the robot gets stuck at a local minimum of the potential field, leaving no guarantee that the goal will be reached. To overcome this problem, several solutions have been proposed. One is to try to escape from such a "potential well" by making a few small random moves, in the hope that one of them will put the robot in a position from which the field leads it away from this well. Another approach is to use the potential field only for subproblems where the initial and final placements are close to each other, so the chance to get stuck at a local minimum is small. One then generates many random placements throughout the workspace, and applies the potential field technique to attempt to connect many pairs of them, until a path is generated from start to goal. (In this randomized technique, any convenient local planner may be used.) See [Lat91, KSLO] and Section 41.4 for more details concerning this technique.

Fat obstacles. Another technique exploits the fact that, in typical layouts, the obstacles can be expected to be "fat" (this has several definitions; intuitively, they do not have long and skinny parts). Also, the obstacles tend not to be too clustered, in the sense that each placement of the robot can interact with only a constant number of obstacles. These facts tend to make the problem easier to solve. See [SO94] for such a solution.

40.4 DAVENPORT-SCHINZEL SEQUENCES

Davenport-Schinzel sequences are interesting and powerful combinatorial structures that arise in the analysis and calculation of the lower or upper envelope of collections of functions, and therefore have applications in many geometric problems, including numerous motion planning problems, which can be reduced to the calculation of such an envelope. A recent comprehensive survey of Davenport-Schinzel sequences and their geometric applications can be found in [SA95].

An (n, s) ***Davenport-Schinzel sequence***, where n and s are positive integers, is a sequence $U = (u_1, \ldots, u_m)$ composed of n symbols with the properties:

(i) No two adjacent elements of U are equal: $u_i \neq u_{i+1}$ for $i = 1, \ldots, m-1$.

(ii) U does not contain as a subsequence any alternation of length $s+2$ between two distinct symbols: there do not exist $s+2$ indices $i_1 < i_2 < \cdots < i_{s+2}$ so that $u_{i_1} = u_{i_3} = u_{i_5} = \cdots = a$ and $u_{i_2} = u_{i_4} = u_{i_6} = \cdots = b$, for two distinct symbols a and b.

Thus, for example, an $(n, 3)$ sequence is not allowed to contain any subsequence of the form $(a \cdots b \cdots a \cdots b \cdots a)$. Let $\lambda_s(n)$ denote the maximum possible length of an (n, s) Davenport-Schinzel sequence.

The importance of Davenport-Schinzel sequences lies in their relationship to the combinatorial structure of the lower (or upper) envelope of a collection of functions (Section 21.2). Specifically, for any collection of n real-valued continuous functions $f_1, \ldots, f_n$ defined on the real line, having the property that each pair of them intersect in at most s points, one can show that the sequence of function indices i in the order in which these functions attain their lower envelope (i.e., their pointwise minimum $f = \min_i f_i$) from left to right is an (n, s) Davenport-Schinzel sequence. Conversely, any (n, s) Davenport-Schinzel sequence can be realized in this way for an appropriate collection of n continuous univariate functions, each pair of which intersect in at most s points.

The crucial and surprising property of Davenport-Schinzel sequences is that, for a fixed s, the maximal length $\lambda_s(n)$ is nearly linear in n, although for $s \geq 3$ it is slightly super-linear. Specifically, one has

$$\begin{aligned}
\lambda_1(n) &= n \\
\lambda_2(n) &= 2n - 1 \\
\lambda_3(n) &= \Theta(n\alpha(n)) \\
\lambda_4(n) &= \Theta(n \cdot 2^{\alpha(n)}) \\
\lambda_{2s}(n) &\leq n \cdot 2^{\alpha(n)^{s-1} + C_{2s}(n)} \\
\lambda_{2s+1}(n) &\leq n \cdot 2^{\alpha(n)^{s-1} \log \alpha(n) + C_{2s+1}(n)} \\
\lambda_{2s}(n) &= \Omega(n \cdot 2^{\frac{1}{(s-1)!}\alpha(n)^{s-1} + C'_{2s}(n)}),
\end{aligned}$$

where $\alpha(n)$ is the inverse of Ackermann's function, and where $C_r(n)$, $C'_r(n)$ are asymptotically smaller than the leading terms in the respective exponents. Ackermann's function $A(n)$ grows extremely quickly, with $A(4)$ an exponential "tower" of 65636 2's. Thus $\alpha(n) \leq 4$ for all practical values of n. See [SA95].

If one considers the lower envelope of n continuous, but only partially defined, functions, then the complexity of the envelope is at most $\lambda_{s+2}(n)$, where s is the maximum number of intersections between any pair of functions. Thus for a collection of n line segments (for which $s = 1$), the lower envelope consists of at most $O(n\alpha(n))$ subsegments. A surprising result is that this bound is tight in the worst case: there are collections of n segments, for arbitrarily large n, whose lower envelope does consist of $\Omega(n\alpha(n))$ subsegments. This is perhaps the most natural example of a combinatorial structure defined in terms of n simple objects, whose complexity involves the inverse Ackermann's function.

Algorithms. The lower envelope of n given total or partial continuous functions, each pair of which intersect in at most s points, can be computed by a simple divide-and-conquer technique that runs (in an appropriate model of computation) in time $O(\lambda_s(n)\log n)$ or $O(\lambda_{s+2}(n)\log n)$ (depending on whether the functions are totally or partially defined). A refined technique reduces the time for partially-defined functions to $O(\lambda_{s+1}(n)\log n)$. Thus, in the case of segments, the algorithm computes their lower envelope in optimal $O(n\log n)$ time. More complex combinatorial and algorithmic applications of Davenport-Schinzel sequences (such as the complexity and construction of a single face in a planar arrangement) are mentioned throughout this chapter.

40.5 SOURCES AND RELATED MATERIAL

SURVEYS

All results not given an explicit reference above, and additional material on motion planning and related problems may be traced in these surveys:

[Lat91]: A book devoted to robot motion planning.

[HSS87]: A collection of early papers on motion planning.

[SA95]: A book on Davenport-Schinzel sequences and their geometric applications; contains a section on motion planning.

[HS95b]: A recent review on arrangements and their applications to motion planning.

[SS88, SS90, Sha89, Sha95, AY90]: Several survey papers on algorithmic motion planning.

RELATED CHAPTERS

Chapter 20: Voronoi diagrams and Delaunay triangulations
Chapter 21: Arrangements
Chapter 24: Shortest paths and networks
Chapter 29: Computational real algebraic geometry
Chapter 41: Robotics

REFERENCES

[AAAS96] P.K. Agarwal, N. Amenta, B. Aronov, and M. Sharir. Largest placements and motion planning of a convex polygon. In *Proc. 2nd Annu. Workshop Algorithmic Found. Robot.*, Toulouse, 1996.

[ABHS96] P. Agarwal, M. de Berg, D. Halperin, and M. Sharir. Efficient generation of k-directional assembly sequences. In *Proc. 7th ACM-SIAM Sympos. Discrete Algorithms*, pages 122–131, 1996.

[AFK+90] H. Alt, R. Fleischer, M. Kaufmann, K. Mehlhorn, S. Näher, S. Schirra, and C. Uhrig. Approximate motion planning and the complexity of the boundary of the union of simple geometric figures. In *Proc. 6th Annu. ACM Sympos. Comput. Geom.*, pages 281–289, 1990.

[AKY96] T. Asano, D. Kirkpatrick, and C.K. Yap. d_1-optimal motion for a rod. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, pages 252–263, 1996.

[AS94] B. Aronov and M. Sharir. Castles in the air revisited. *Discrete Comput. Geom.*, 12:119–150, 1994.

[AS] B. Aronov and M. Sharir. On translational motion planning of a convex polyhedron in 3-space. *SIAM J. Comput.*, to appear.

[AY90] H. Alt and C.K. Yap. Algorithmic aspects of motion planning: A tutorial, Parts 1 and 2. *Algorithms Rev.*, 1:43–60, 61–77, 1990.

[BBFY92] E. Bar-Eli, P. Berman, A. Fiat, and P. Yan. On-line navigation in a room. In *Proc. 3rd ACM-SIAM Sympos. Discrete Algorithms*, pages 237–249, 1992.

[BBF+96] P. Berman, A. Blum, A. Fiat, H. Karloff, A. Rosen, and M. Saks. Randomized robot navigation algorithms. In *Proc. 7th ACM-SIAM Sympos. Discrete Algorithms*, pages 75–84, 1996.

[BPR96] S. Basu, R. Pollack, and M.-F. Roy. Computing roadmaps of semi-algebraic sets. In *Proc. 28th Annu. ACM Sympos. Theory Comput.*, pages 168–173, 1996.

[BRS91] A. Blum, P. Raghavan, and B. Schieber. Navigating in unfamiliar geometric terrain. In *Proc. 23rd Annu. ACM Sympos. Theory Comput.*, pages 494–504, 1991.

[Can87] J. Canny. *The Complexity of Robot Motion Planning.* MIT Press, Cambridge, 1987. See also: Computing roadmaps in general semi-algebraic sets. *Comput. J.*, 36:504–514, 1993.

[CK93] L.P. Chew and K. Kedem. A convex polygon among polygonal obstacles: placement and high-clearance motion. *Comput. Geom. Theory Appl.*, 3:59–89, 1993.

[CR87] 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.

[CY91] J. Cox and C.K. Yap. On-line motion planning: Case of a planar rod. *Ann. Math. Artif. Intell.*, 3:1–20, 1991.

[dBOS94] M. de Berg, M. Overmars, and O. Schwarzkopf. Computing and verifying depth orders. *SIAM J. Comput.*, 23:437–446, 1994.

[GMR92] L. Guibas, R. Motwani, and P. Raghavan. The robot localization problem in two dimensions. In *Proc. 3rd ACM-SIAM Sympos. Discrete Algorithms*, pages 259–268, 1992.

[HCA+95] S. Har-Peled, T.M. Chan, B. Aronov, D. Halperin, and J. Snoeyink. The complexity of a single face of a Minkowski sum. In *Proc. 7th Canad. Conf. Comput. Geom.*, Québec City, pages 91–96, 1995.

[HS95a] D. Halperin and M. Sharir. Almost tight upper bounds for the single cell and zone problems in three dimensions. *Discrete Comput. Geom.*, 14:385–410, 1995.

[HS95b] D. Halperin and M. Sharir. Arrangements and their applications in robotics: Recent developments. In K. Goldberg, D. Halperin, J.-C. Latombe, and R. Wilson, editors, *The Algorithmic Foundations of Robotics*, pages 495–511. A K Peters, Boston, 1995.

[HS96] D. Halperin and M. Sharir. A near-quadratic algorithm for planning the motion of a polygon in a polygonal environment. *Discrete Comput. Geom.*, 16:121–134, 1996.

[HSS87] J.E. Hopcroft, J.T. Schwartz, and M. Sharir, editors. *Planning, Geometry, and Complexity of Robot Motion*. Ablex, Norwood, 1987.

[KLPS86] K. Kedem, R. Livne, J. Pach, and M. Sharir. On the union of Jordan regions and collision-free translational motion amidst polygonal obstacles. *Discrete Comput. Geom.*, 1:59–71, 1986.

[KO88] Y. Ke and J. O'Rourke. Lower bounds on moving a ladder in two and three dimensions. *Discrete Comput. Geom.*, 3:197–217, 1988.

[KSLO] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high dimensional configuration spaces. *IEEE Trans. Robot. Autom.*, to appear. Also Tech. Rept. STANFORD-CS-TR-94-1519.

[Lat91] J.-C. Latombe. *Robot Motion Planning*. Kluwer, Boston, 1991.

[LS87a] D. Leven and M. Sharir. Planning a purely translational motion for a convex object in two-dimensional space using generalized Voronoi diagrams. *Discrete Comput. Geom.*, 2:9–31, 1987.

[LS87b] D. Leven and M. Sharir. An efficient and simple motion planning algorithm for a ladder moving in 2-dimensional space amidst polygonal barriers. *J. Algorithms*, 8:192–215, 1987.

[LS87] V.J. Lumelsky and A.A. Stepanov. Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape. *Algorithmica*, 2:403–430, 1987.

[OSY87] C. Ó'Dúnlaing, M. Sharir, and C.K. Yap. Generalized Voronoi diagrams for a ladder: II. Efficient construction of the diagram. *Algorithmica*, 2:27–59, 1987.

[RS94] J. Reif and M. Sharir. Motion planning in the presence of moving obstacles. *J. Assoc. Comput. Mach.*, 41:764–790, 1994.

[SA95] M. Sharir and P.K. Agarwal. *Davenport-Schinzel Sequences and Their Geometric Applications*. Cambridge University Press, 1995.

[Sha89] M. Sharir. Algorithmic motion planning in robotics. *Computer*, 22:9–20, 1989.

[Sha95] M. Sharir. Robot motion planning. *Comm. Pure Appl. Math.*, 48:1173–1186, 1995. Also in E. Schonberg, editor, *The Houses that Jack Built*. Courant Institute, New York, 1995, 287–300.

[SO94] A.F. van der Stappen and M.H. Overmars. Motion planning amidst fat obstacles. In *Proc. 10th Annu. ACM Sympos. Comput. Geom.*, pages 31–40, 1994.

[SM88] K. Sutner and W. Maass. Motion planning among time-dependent obstacles. *Acta Inform.*, 26:93–122, 1988.

[SS83] J.T. Schwartz and M. Sharir. On the piano movers' problem: II. General techniques for computing topological properties of real algebraic manifolds. *Adv. Appl. Math.*, 4:298–351, 1983.

[SS88] J.T. Schwartz and M. Sharir. A survey of motion planning and related geometric algorithms. *Artif. Intell.*, 37:157–169, 1988. Also in D. Kapur and J. Mundy, editors, *Geometric Reasoning*, pages 157–169. MIT Press, Cambridge, 1989. And in S.S. Iyengar and A. Elfes, editors, *Autonomous Mobile Robots*, volume I, pages 365–374. IEEE Computer Society Press, Los Alamitos, 1991.

[SS90] J.T. Schwartz and M. Sharir. Algorithmic motion planning in robotics. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity*, pages 391–430. Elsevier, Amsterdam, 1990.

[SS91] M. Sharir and S. Sifrony. Coordinated motion planning for two independent robots. *Ann. Math. Artif. Intell.*, 3:107–130, 1991.

[SS96] O. Schwarzkopf and M. Sharir. Vertical decomposition of a single cell in a 3-dimensional arrangement of surfaces and its applications. In *Proc. 12th Annu. ACM Sympos. Comput. Geom.*, pages 20–29, 1996.

[Wil91] G. Wilfong. Motion planning in the presence of movable obstacles. *Ann. Math. Artif. Intell.*, 3:131–150, 1991.

[WL94] R.H. Wilson and J.-C. Latombe. Geometric reasoning about mechanical assembly. *Artif. Intell.*, 71:371–396, 1994.

[Yap87] C.K. Yap. An $O(n \log n)$ algorithm for the Voronoi diagram of a set of simple curve segments. *Discrete Comput. Geom.*, 2:365–393, 1987.