J Intell Robot Syst (2010) 57:65–100
DOI 10.1007/s10846-009-9383-1

# A Survey of Motion Planning Algorithms from the Perspective of Autonomous UAV Guidance

**C. Goerzen · Z. Kong · B. Mettler**

Received: 1 February 2009 / Accepted: 1 August 2009 / Published online: 17 November 2009


**Abstract** A fundamental aspect of autonomous vehicle guidance is planning trajectories. Historically, two fields have contributed to trajectory or motion planning methods: robotics and dynamics and control. The former typically have a stronger focus on computational issues and real-time robot control, while the latter emphasize the dynamic behavior and more specific aspects of trajectory performance. Guidance for Unmanned Aerial Vehicles (UAVs), including fixed- and rotary-wing aircraft, involves significant differences from most traditionally defined mobile and manipulator robots. Qualities characteristic to UAVs include non-trivial dynamics, three-dimensional environments, disturbed operating conditions, and high levels of uncertainty in state knowledge. Otherwise, UAV guidance shares qualities with typical robotic motion planning problems, including partial knowledge of the environment and tasks that can range from basic goal interception, which can be precisely specified, to more general tasks like surveillance and reconnaissance, which are harder to specify. These basic planning problems involve continual interaction with the environment. The purpose of this paper is to provide an overview of existing motion planning algorithms while adding perspectives and practical examples from UAV guidance approaches.

**Keywords** Autonomous · UAV · Guidance · Trajectory · Motion planning · Optimization · Heuristics · Complexity · Algorithm

This research was completed under grants NNX08A134A (San Jose State University Research Foundation) and NNX07AN31A (University of Minnesota) as part of the US Army Aeroflightdynamics Directorate (RDECOM) flight control program.

C. Goerzen
NASA Ames Research Center, San Jose State University Research Foundation,
Moffett Field, CA 94035, USA
e-mail: chad.goerzen@us.army.mil

Z. Kong · B. Mettler (✉)
Department of Aerospace Engineering and Mechanics,
University of Minnesota, Minneapolis, USA
e-mail: mettler@aem.umn.edu

## 1 Introduction

In the last decade, significant changes have occurred in the field of vehicle motion planning, and for UAVs in particular. UAV motion planning is especially difficult due to several complexities not considered by earlier planning strategies: the increased importance of differential constraints, atmospheric turbulence which makes it impossible to follow a pre-computed plan precisely, uncertainty in the vehicle state, and limited knowledge about the environment due to limited sensor capabilities. These differences have motivated the increased use of feedback and other control engineering techniques for motion planning. The lack of exact algorithms for these problems and difficulty inherent in characterizing approximation algorithms makes it impractical to determine algorithm time complexity, completeness, and even soundness. This gap has not yet been addressed by statistical characterization of experimental performance of algorithms and benchmarking. Because of this overall lack of knowledge, it is difficult to design a guidance system, let alone choose the algorithm.

Throughout this paper we keep in mind some of the general characteristics and requirements pertaining to UAVs. A UAV is typically modeled as having velocity and acceleration constraints (and potentially the higher-order differential constraints associated with the equations of motion), and the objective is to guide the vehicle towards a goal through an obstacle field. A UAV guidance problem is typically characterized by a three-dimensional problem space, limited information about the environment, on-board sensors with limited range, speed and acceleration constraints, and uncertainty in vehicle state and sensor data.

In order to be able to compare algorithms on a rigorous basis it is necessary to establish their degree of soundness, completeness, optimality, precision, and computational complexity. Once accomplished, this work will help in choosing an algorithm based on the requirements called for by a particular application. This brings up the dual of the problem of understanding the algorithm, which is that of understanding the requirements of a particular application. In the end, these are two sides of one larger problem. Before understanding the larger problem of UAV guidance we start with a comprehensive perspective of the field and a general understanding of the UAV guidance problem. This review covers recent developments in the robotics and aerospace guidance and control fields. It summarizes as comprehensibly as possible what has been claimed or proven regarding these algorithms. It does not attempt to fill in characteristics for algorithms that have not been published. There is a great deal of work remaining to be done in order to characterize existing algorithms, in particular those capable of handling differential constraints.

The paper is organized as follows. The next section gives an overview of the fundamental issues in vehicle motion planning, including key terminology, definitions and concepts, problem types, problem metrics, and algorithm performance criteria. Section 3 provides a survey of recent work focusing on path planning without differential constraints. These include: roadmap methods, exact cell decomposition, approximate cell decomposition, potential field methods, probabilistic approaches, and weighted region problems. Section 4 provides a survey of trajectory planning with differential constraints. These include: state-space sampling methods, minimum distance discrete path followed by trajectory forming, mathematical programming, potential field methods, and solutions given uncertainty. We conclude the paper with a brief discussion based on the broad perspective offered by the survey.

## 2 Background and Overview

The main focus of this article is to survey algorithms that solve the problem of trajectory planning of a dynamics-constrained vehicle through an environment with obstacles. For many UAV applications, a point vehicle representation usually suffices as a slightly conservative assumption that vastly simplifies the problem. In addition, the problem of finding a trajectory that minimizes some cost functional is of interest. Given that an exact solution of trajectory planning in an obstacle field is a variational calculus problem with analytic solutions that are computable only for some of the simplest cases, all the algorithms used to solve this problem in three-dimensional space are approximation algorithms. Many of the algorithms designed to solve the dynamics-constrained problem rely on a decomposition approach, first solving a path planning problem, applying smoothing constraints, forming a trajectory that conforms to the path, and using a control loop to follow this trajectory [70]. Since any of the traditional dynamics-unconstrained algorithms found in the field of robotics may be used for the path planning stage in this approach, these are covered briefly in this review.

### 2.1 Overall Problem Description

Vehicle motion planning is a special case of the general motion planning problem, which in general is very difficult to solve, especially as the number of degrees of freedom increases. In a typical UAV application, the vehicle operates in three-dimensional space, has two to four degrees of freedom, and has differential constraints, including limited speed and maximum acceleration. The resulting problem space has at least five to 12 dimensions, associated with the equations of motion and involving constraints on states and input variables. There does not exist an algorithm that provides an exact analytic solution to such a problem. Indeed, even state-of-the-art approximation algorithms operating on a three-dimensional subspace of this problem space are difficult to compute in real time. Furthermore, several simplifications and sub-cases of the general problem have been proven to be unsolvable in polynomial time [18]. Approximation algorithms are possible, and often rely on exact solutions to simplified sub-problems.

### 2.2 Previous Surveys

There are several important works covering the field of motion planning, most of them in the form of textbooks. The book *Complexity of Robot Motion Planning* [18] proves bounds on several motion planning problems that are still relevant. The book by Schwartz and Yap [98] contains another review. The classic textbook *Robot Motion Planning* [66] gives detailed explanations of the algorithms used until 1990, and remains one of the best sources of information on algorithms for solving path planning problems on polygonal obstacle field representations. Another textbook, [39] titled *Motion Planning in Dynamic Environments*, systematically covers the dynamic environment problem and describes computational bounds for several problems. Hwang and Ahuja [48] is a comprehensive survey of the field of motion planning from the same year.

More recently, Barto et al. [4] includes a comprehensive overview of optimal control type approaches, especially dynamic programming, in its first 10 pages. Tarabanis [64] discusses closely related problems of sensor based planning, and [109] reviews probabilistic approaches. The most significant recent work is [70], a textbook titled *Planning Algorithms*. Although not designed to be a comprehensive survey, this book provides broad coverage of the field of motion planning algorithms, and has a strong focus on vehicle motion planning. Chapters 8 and 14 of this book are of particular interest to problems covered here, since they present material not found in the above-mentioned survey works. Chapter 8: Feedback Motion Planning, covers practical planning algorithms that provide feedback for all possible configurations of the vehicle, generally using navigation functions. Chapter 14: Sampling-Based Planning Under Differential Constraints, discusses approximate solutions for vehicles where differential constraints are important. Another important recent work is [33], which covers the heuristic-based algorithms (such as A* and D*) with a focus on dynamic problems, and contain concise summaries of the algorithms and many important references.

The following does not attempt to duplicate the efforts in these previous surveys, but rather to provide a summary of developments in the field since 1992, with a focus on the problem of trajectory planning for the point vehicle with differential constraints. As a result, there are many references below to the more important of these previous surveys. Especially in the review of path planning literature, for many algorithms a reference to one of these surveys is given as an only source.

## 2.3 Definitions and Terminology

One difficulty when reviewing and studying motion planning or trajectory planning comes from the varied background of the work. We start with an introduction of the key definitions and concepts. These will then be used consistently throughout this paper. Some terms are defined anew here but most are consistent with terminology used in Steven LaValle [70] and Hwang and Ahuja [48].

### *2.3.1 Problem Space*

A *vehicle* is an object that is capable of motion. Generally, vehicles are represented by a position vector in two- or three-dimensional space and an orientation vector, along with a geometric model of the vehicle. A vehicle is defined in a similar way to Hwang's *robot*, but is simpler in that it does not contain a manipulator. A *world space* (Fig. 1) is the physical space in which a vehicle exists. For example, a helicopter may exist in three-dimensional Euclidean space, while a ground vehicle may be thought to exist on a two-dimensional surface embedded in three-dimensional space. A *configuration* is a vector of parameters that define the shape of the vehicle—most vehicles can be considered to be rigid bodies in three-dimensional space, and thus defined uniquely by six numbers: three position coordinates and three orientation coordinates. A robot with a manipulator will generally have a much larger number of parameters, because each degree of freedom of the manipulator adds a parameter to the configuration space. The set of all possible configurations of a vehicle is called the *configuration space* or *C-space*. It is often necessary, especially in the case of aircraft, to include a *state*, which consists of the configuration coupled with rates of change of the configuration. In our aircraft example, the state is given by three

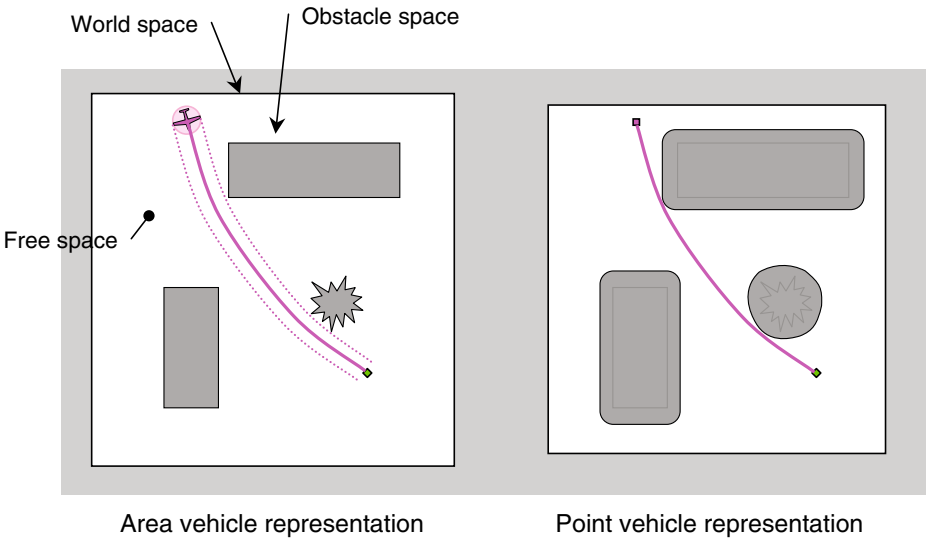



**Fig. 1** Illustration of terms. Two representations of a simple two-dimensional example case are shown in order to illustrate terms. On the *left* the vehicle is represented by a disk, and on the *right* the vehicle is represented by a point in the expanded terrain representation: the vehicle is a UAV, initial state is marked by a *square*, and goal state is marked by a *diamond*; in both cases the configuration space reduces to the world space; the state space is not shown, but velocity and acceleration constraints are implied. Only the path is shown as a partial representation of the trajectory, without timing information

positions coordinates, three velocity coordinates, three orientation angles, and three orientation rate angles, for a total of 12 variables. The set of all possible states is known as the *state space*. The dynamic characteristics of the vehicle determine the dimension of the system, and many systems may use a reduced set of variables that adequately describe the physical state of the vehicle. It is common to consider smaller state spaces with coupled states, or to extend the state space to include higher-order derivatives. The number of coordinates needed to uniquely represent each configuration or state is known as the number of *degrees of freedom*, and is equivalent to the dimension of the configuration or state space. The world, configuration, or state space is divided into *free space* and *obstacle space*; these are disjoint subsets of the space. Free space is the subset of points in which the vehicle can move without contacting obstacles, and obstacle space is the subset of points representing a collision (in the case of state space, this includes the *region of inevitable collision*) between the vehicle and an obstacle. A *path* is a curve traced by the vehicle in the configuration space, and a *trajectory* is a path that includes the time along the path. A trajectory is typically associated with an equation of motion since the differential equation describes the coupling between the spatial and temporal evolution of the system states. Motion planning refers to either path or trajectory planning, and produces a path or trajectory from an *initial* state or configuration to a *goal* state or configuration. More generally, this can be considered as set of paths from a set of initial states to a set of goal states.

### *2.3.2 Motion Planning Terminology*

A motion planning algorithm is considered to be *complete* if and only if it finds a path when one exists, and returns a variable stating no path exists when none exists. It is considered to be *optimal* when it returns the optimal path with respect to some criterion. Note that any optimal planner is also complete. Two weaker forms of completeness and optimality are also commonly used: *resolution completeness/optimality* and *probabilistic completeness/optimality*. Resolution completeness/optimality is related to the discretization of the solution space, and means that as the resolution of the discretization increases, an exact solution is achieved as the discretization approaches the continuum limit. Probabilistic completeness/optimality means that as computing time approaches infinity, the probability of finding an exact solution approaches one. Anthony Lazanas [3] discusses motion planning with uncertainty, and makes use of additional definitions. A *sound* planner is one that always guarantees the vehicle will enter the goal region and stop there without hitting any obstacle despite uncertainty in sensing and control. This definition implies that the uncertainties are bounded. Soundness is perhaps the most crucial property for UAVs, because the results of a collision are so catastrophic.

## 2.4 Algorithm Characteristics

### *2.4.1 Algorithmic Complexity*

Standard algorithmic asymptotic complexity analysis methods are used in this paper. The variable $N$ (or $n$) denotes the complexities of describing the obstacle space (i.e. the number of obstacles), and represent the number of bits needed to define this space. Whenever a continuous space is approximated by a finite set of variables, $M$ denotes the number of variables used to approximate this space, or the level of discretization. In robotics literature, $M$ (or $m$) is sometimes used to denote complexity of describing the robot shape—however, in this article $M$ is used exclusively for the level of discretization. The number of dimensions of the configuration or state-space is denoted by $D$. Asymptotic notation is used for complexity analysis whenever possible: $O$ () means the algorithm is bounded from above (within a constant factor), $\Omega$ () denotes an algorithm bounded from below, and $\Theta$ () denotes an algorithm bounded both above and below. Unless otherwise stated, complexity is based on time rather then on memory. Additionally, the standard notation of P, NP, PSPACE, EXPTIME, and EXPSPACE are used to characterize algorithms. The relation:

$$P \subseteq NP \subseteq PSPACE \subseteq EXPTIME \subseteq EXPSPACE$$

holds (with the condition that it is an open problem whether these spaces are proper or improper subsets of each other). Practically speaking, only algorithms in P are solvable in real time. However, approximation algorithms often exist for more difficult problems which admit an approximate solution with lower complexity then the exact solution.

### *2.4.2 Performance of Approximation Algorithms*

Standard notation is also used for approximation bounds. A *relative performance guarantee* is denoted by the quantity $\rho$: a $\rho$-approximation algorithm is accurate within a factor $\rho$ of the true solution. An *absolute performance guarantee* is denoted by the quantity $\varepsilon$, and means the approximation algorithm is accurate within a constant value $\varepsilon$ of the true solution.

## 2.5 Problem Types

There are a variety of problem types defined in the literature. A problem is considered *static* if there is perfect knowledge of the environment, and *dynamic* if knowledge of the environment is imperfect or changes as the task unfolds. When the obstacles are fixed in space, the problem is called *time-invariant*, and when they are allowed to move, the problem is called *time-variant*. The term *differentially constrained* (or *kinodynamic*) means that vehicle's equations of motion act as constraints on the path, i.e., the path must be a trajectory of the dynamic system (for *differentially unconstrained* problems, the vehicle may use infinite accelerations to achieve a path).

It is possible to further categorize problems based on the assumed vehicle shape, environment type and behavior. The common problem types used in literature are described below.

### *2.5.1 Point Vehicle (Point Robot)*

In this problem, the vehicle is modeled as point within the world space. Thus the configuration space is the same as the world space. Often, a vehicle is modeled by fitting it inside a bounding ball (in two-dimensional Euclidean space this is a circle and in three-dimensional Euclidean space a sphere), and the configuration space is simply the world space with the obstacles expanded by the radius of the vehicle's bounding ball. Thus the ball-shaped-vehicle problem is the same as the point vehicle problem. This is a conservative approximation to and simplification of the mover's problem (described below).

This is perhaps the simplest problem, and optimality is defined in terms of the distance between initial and goal points, i.e., the minimum-length path is the optimal path.

### *2.5.2 Point Vehicle with Differential Constraints*

In problems with differential constraints, time and states have to satisfy the equations of motion of the vehicle (associated with Newton's second law). Typically the states are constrained by hard limits on velocity and acceleration, and sometimes also on higher-order derivatives of position. For many UAVs, this more realistic model is needed for stable control of the vehicle. In the limit of a vehicle constrained only by velocity bounds (a vehicle capable of infinite acceleration), this problem becomes simplified to a point vehicle problem without differential constraints.

Optimality may be defined as minimizing the flight time between initial and goal points, or by minimizing some other attributes of the trajectory, such as energy consumed.

### *2.5.3 Jogger's Problem*

This problem, named in Shkel and Lumelsky [101], deals with the dynamic problem of a jogger with a limited field of view attempting to reach a goal position. The jogger's problem is representative of a vehicle with differential constraints, operating in a dynamic and possibly time-variant environment, with limited sensory range.

Optimality is defined the in the same manner as in other differentially constrained problems.

### *2.5.4 Bug Problem*

This is actually a special case of the jogger's problem, in the limit when the field of view goes to zero. A complete algorithm for this problem is described in [118]. In this case, the vehicle must touch the obstacle (or come in very close proximity) in order to sense it.

### *2.5.5 Weighted Region Problem*

When some regions of the world, configuration, or state-space are known to be more desirable then others, the problem is better modeled as a weighted-region problem. An example of this problem is the case of an off-road ground vehicle that drives more quickly over smooth terrain and more slowly over rough terrain. The terrain is represented by a function close to 0 for smooth terrain, and with an infinite value representing un-navigable terrain. This type of problem may be formulated either as an unconstrained or a dynamics-constrained vehicle.

In this problem, an optimum planner minimizes the integral of the weight over the path.

### *2.5.6 Mover's Problem*

This is the problem of moving an object through an obstacle field to a goal state. The vehicle is usually modeled as a rigid body, thus the configuration space has a larger dimension than the world space. A classical problem of this case is the famous *piano mover's problem*. For this kind of problem, it is usually assumed that the object has no dynamic constraints. Mover's problems measure complexity of the vehicle in addition to that of the obstacle field, and call this number $m$ or $M$. A more general case is to allow dynamic constraints, and is most commonly used in manufacturing. The differentially-constrained mover's problem is of very high dimension: for example, a three-dimensional differentially-constrained mover's problem has 12 dimensions. Given this difficulty, this problem is generally solved in configuration space, and many algorithms are merely complete and ignore the optimization problem. Some algorithms maximize closest approach between the vehicle and obstacle field.

### *2.5.7 General Vehicle with Differential Constraints*

This is the most sophisticated problem type that is investigated. The differential constraints typically arise in two forms: one is on kinematics, and this kind of problem are usually called nonholonomic problem. Another one is on dynamics, this means that it involves second-order or higher differential constraints. The difference between this problem and the point vehicle problem is that now it is insufficient to model the vehicle with only a point in the world space, since we now need six

variables to indicate the position of the vehicle in a three dimensional Euclidean space. But for a space with no obstacles, the location can be converted to a point in a higher dimension configuration space. For most cases, the configuration space is not a simple Euclidean space. To make things worse, when obstacles are involved, the configuration space itself is not adequate to represent the obstacle avoidance requirements, a higher order phase space has to be employed.

#### *2.5.8 Time-Varying Environments*

In this problem, the vehicle has to avoid obstacles that are moving in time. Fujimura and Tunii [39] discusses this problem in depth, and determines some upper and lower bounds for algorithm complexity.

Optimal planners for time-varying environments generally attempt to minimize path length or time.

#### *2.5.9 Multiple Movers Problem (Multimovers)*

This is the case where there are multiple vehicles moving in the same space. It can be seen as a combination of the mover's problem and the time varying environment case. An example is a factory floor where many robots move between stations. As the dimension of the configuration space increases for each robot added, algorithms scale poorly with the number of robots.

One possible way of measuring optimality is to minimize the sum of the distance, time, or some other quantity covered by all vehicles in the problem space.

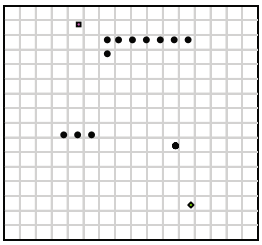
### 2.6 Problem Metrics

The choice of algorithm to use depends on the type of problem to be solved. For example, if the problem is one of plotting a course for a truck to travel across a continent, there is no need to use a dynamics-constrained planner. The most commonly-used metric is obstacle complexity, or the amount of information used store a computer model of the environment. It is generally measured in terms of number of obstacles, obstacle edges, or obstacle vertices, and is called $N$. Other metrics are the fill ratio (percentage of the configuration space occupied by obstacles), along with higher-order characteristics, such as mean passage width or degree of clustering of obstacles. A useful and thorough discussion on metrics useful for vehicle motion planning is available in Rhinehart [90].

### 2.7 Algorithm Performance Criteria

One of the purposes of this paper is to list the performance of various motion planning algorithms published in literature. Criteria that are important for a successful practical implementation include operational and computational aspects. Consistently keeping a safe distance from obstacles, and producing smooth paths that exhibit desirable properties (e.g. duration, energy usage) are typical requirements. A reliable, real-time computation without unexpected lags is critical since it can impede maintaining the operational requirements. Low computational complexity is therefore generally an important goal for an algorithm. A faster algorithm can allow a more rapid update of the solution. The main algorithm characteristics

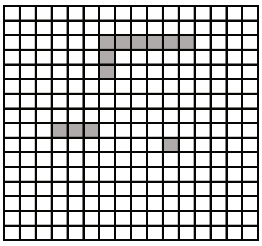
used for describing algorithms include computational complexity (often described in asymptotic big-O notation), and degree of approximation to the optimal solution (generally described by a maximum error factor or bound). Given the often uncertain



**Sensor model:**

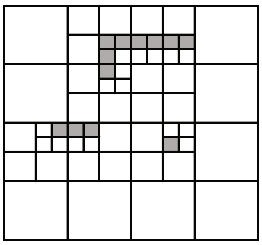
Sensor data representation:

- Black points represent sensed obstacle points
- Initial position marked by a square, goal position marked by a diamond
-The grid is not actually part of the representation



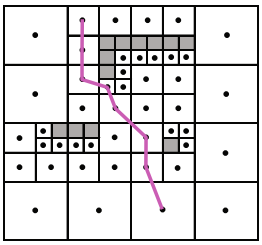
**Terrain representation:**

Obstacle space represented in this example by grey pixels and free space by white pixels



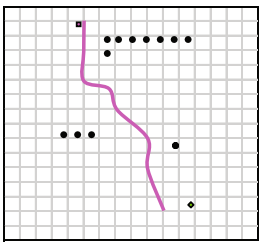
**Roadmap:**

Many types of roadmap are possible – shown here is a quadtree spatial decomposition map



**Graph search:**

The roadmap step produces a graph, and graph search algorithms such as Dijkstra's method or A* are used to find a sequence of waypoints



**Trajectory generator:**

This starts with waypoints produced by the graph search, and is smoothed into a flyable trajectory. This may be divided into two steps: smoothing and speed control. Only smoothing is shown in this image.

**Fig. 2** Stages in a typical example multi-level decoupled control type planner

nature of real-world problems, the ability to deal with various types of uncertainty and system errors is also crucial for algorithms in practical applications. Another way to deal with uncertainty is to have adaptive algorithms which do not require re-computing.

Asymptotic polynomial time computational complexity for an exact algorithm has been demonstrated for some simple problems (especially in two-dimensional problem spaces). For most problems considered for UAV tasks, it has been proven that no algorithm exists that can find exact solutions in polynomial time. In the absence of a proven computational bound, it is important to have methods that allow precise benchmarking in order to characterize the algorithm, and demonstrate experimental performance. Ultimately, an algorithm's performance is judged by the operators of the actual vehicle in the field; this practical perspective needs to be kept in mind when analyzing computational complexity.

### 2.8 Problem Definition for a UAV as an Illustrative Example

Many UAVs can be modeled as a point vehicle, as described above in Section 2.2. This is the simplest model that is sufficiently detailed to produce a practically meaningful result: Since UAVs don't have to fit into tight spaces while flying, the simplification of bounding the UAV by a rotationally-symmetrical solid has little effect on the trajectory generated by the algorithm. The mover's problem therefore has more complexity then what is actually needed for UAV tasks.

In many UAV problems, the differential constraints are significant since not accounting for the equations of motion may produce conservative results and also precludes explicitly taking into account for criteria like duration or energy.

The jogger's problem, weighted region problem, and multiple vehicle problem are also useful to consider. Poor GPS telemetry, air turbulence, or obstacle detection and registration are common issues in UAV applications. Properly dealing with such sources of uncertainties has not yet been adequately posed or studied for this class of planning problems. In fact, the traditional problem statement of motion planning does not admit solution that is sound or complete, whenever uncertainty is characterized by an infinite-tailed distribution (such as the normal distribution). In other words, the motion planning is ill-posed for most practical problems involving uncertainty.

Figure 2 shows stages in an example multi-stage algorithm, and illustrates the close interaction between sensor data processing and motion planning. Many algorithms are able to combine two or more of these stages.

## 3 Recent Work in Path Planning Without Differential Constraints

Path planning without differential constraints is included in this article because it is often used as a basis for algorithms with differential constraints. Most of the algorithms are well covered by other surveys and textbooks, so comprehensive references are not provided for this class of problems unless significant advancements have been published. The reader is advised to refer to the references given in Section 2.2: Previous surveys. Table 1 provides a direct side-to-side comparison of all the algorithms presented in this section.

**Table 1** Comparison of algorithms for path planning without differential constraints

| Algorithm name | Type of problem solved | Number of dimensions | Completeness | Optimality | Proven time complexity |
|---|---|---|---|---|---|
| **Roadmap** | | | | | |
| Visibility graph | Point vehicle | 2 | Complete | Pptimal | $O(N^2)$ |
| Edge-sampled visibility graph | Point vehicle | 3 | Complete | Resolution optimal | Exact solution is NP hard, approximation polynomial in N and 1/eta |
| Voronoi roadmap | Point vehicle | 2 | Complete | Non-optimal | $O(N \log N)$ |
| Freeway method | Point vehicle | 2 | Incomplete | Non-optimal | |
| Silhouette method | Point vehicle | Arbitrary | Complete | Non-optimal | $O(\text{Exp}[D,N])$ |
| **Exact cell decomposition** | | | | | |
| Trapezoidal decomposition | Point vehicle | 2 | Complete | Non-optimal | $O(N \log N)$ |
| Critical curves and non-critical regions | Regid vehicle | 2 | Complete | Non-optimal | $O(N^2 \log N)$ |
| Cylindrical algebraic decomposition | Regid vehicle | Arbitrary | Complete | Bon-optimal | $O(\text{Exp}(\text{Exp}(N))$, poly$(P)$ |
| **Approximate cell decomposition** | | | | | |
| Rectanguloid | Point vehicle | 2 | Resolution complete | Non-optimal | |
| Quadtreee, octree, or 2ˆm tree decomposition | Point vehicle | Arbitrary | Resolution complete | Non-optimal | |
| Approximate and decompose | Point vehicle | 2 | Resolution complete | Non-optimal | |

| | | | | | |
|---|---|---|---|---|---|
| **Potential-based methods** | | | | | |
| Potential field | All types | Arbitrary | Incomplete | Non-optimal | |
| Potential guided | All types | Arbitrary | Complete | Non-optimal | |
| Harmonic potential functions | Point vehicle | Arbitrary | Complete | Non-optimal | $O(M \log M)$ |
| Wavefront expansion | Regid vehicle | Arbitrary | Resolution complete | Resolution optimal | $O(M \log M)$ |
| Wavefront expansion with a skeleton (NF2) | Regid vehicle | Arbitrary | Resolution complete | Non-optimal | $O(M \log M)$ |
| Continuous Dijkstra | Regid vehicle | 2 | Complete | Optimal | $O(N^2 \log N)$ |
| **Probabilistic approaches** | | | | | |
| Randomized potential fields | All types | Arbitrary | | Non-optimal | |
| Global learning | All types | Arbitrary | Probabilistically complete when complete search used | Non-optimal | |
| **Weighted region problem** | | | | | |
| Exact algorithm | Polygonal weighted regions | 2 | Complete | Optimal | $O(N^8 \log N)$ |
| Approximation algorithm | Polygonal weighted regions | 2 | Complete | Resolution optimal | $O(N^*M \log N^*M)$ |
| Geodesic distance on a grid | Weighted grid | Arbitrary | Resolution complete | Resolution optimal | $O(M \log M)$ |

## 3.1 Roadmap Methods

These methods reduce the problem to that of a graph search by fitting a graph (ie. a roadmap) to the space. In order for the algorithm based on a given roadmap to be complete, it must follow certain topographical properties [66].

A couple of more recent roadmap methods have been published. Dechter and Pearl [29] verifies the optimality of A*, in comparison with generalized search strategies.

### *3.1.1 Visibility Graph*

This is an exact solution to the point vehicle problem, is both complete and optimal, and runs in $O(N^2)$ time. However, it is computable in only two dimensions. In a $C$-space of more then two dimensions, an exact solution is proven to be NP-hard. This approach uses the knowledge that the shortest path grazes polygonal obstacles at their vertices, and builds a roadmap of lines connecting each vertex with all vertices visible from its position. Since the minimum-length path comes arbitrarily close to obstacles many times in a typical path, this approach offers no safety buffer to prevent collisions in the case of systems with uncertainty in their position. One way researchers have attempted to avoid this problem is by expanding the obstacle space by a ball larger then the vehicle's longest radius.

### *3.1.2 Edge-Sampled Visibility Graph*

This algorithm approximately solves the three-dimensional path length minimization point vehicle problem, and runs in complexity that is polynomial in both error and in computation time. This algorithm assigns multiple vertices along edges of polyhedral obstacles so that there is a minimum edge length $\eta$, and builds a visibility graph from this expanded set of vertices. It is complete and resolution-optimal.

### *3.1.3 Voronoi Roadmap*

Given the difficulty in controlling vehicles precisely enough to follow the minimum-distance path without risk of colliding with obstacles, many skeleton-based roadmap approaches have been taken. The Voronoi approach builds a skeleton that is maximally distant from the obstacles, and finds the minimum distance path that follows this skeleton. This computationally efficient algorithm runs in $O(N \log N)$ time. Although this algorithm is a two-dimensional algorithm, there have been several efforts to convert it to 3 dimensions. It is complete, but not optimal.

More recently, Choset and Burdick [21] proposed a hierarchical Voronoi graph generalized to multiple dimensions, and in [22] the algorithm is updated to allow incremental construction. Howlett et al. [45] discusses use of Voronoi roadmap methods for practical unmanned helicopter operation.

### *3.1.4 Freeway Method*

This method, like the Voronoi roadmap, builds a skeleton that is distant from the obstacles, by fitting free space with generalized cylinders. It is not limited to two dimensions, but it is incomplete and non-optimal.

#### *3.1.5 Silhouette Method*

While not used in practical applications, this method is useful for proving bounds on complexity. This is the only algorithm that is proven to be complete for an arbitrary number of dimensions with arbitrary obstacle geometry. Canny [18], who designed the algorithm, proved that it can solve the problem in double exponential time. The algorithm is complete but not optimal.

### 3.2 Exact Cell Decomposition

These methods decompose the free configuration space into smaller convex polygons, which are then connected by a graph and searched using a graph search.

#### *3.2.1 Trapezoidal (Vertical) Decomposition*

This approach divides the free space into trapezoidal regions by dividing it with vertical lines from each of the obstacle vertices. The vertical lines are trimmed so that they do not bisect the obstacles themselves. A roadmap is then formed by connecting the midpoints of adjacent trapezoids, and searched using a graph searching algorithm. This approach is complete but not optimal, and runs in $O(N \log N)$ time.

#### *3.2.2 Critical-Curve Based Decomposition*

While the trapezoidal decomposition is useful for point vehicle path planning, rigid vehicles with freedom to rotate require a more complex approach. In this algorithm, free space is divided into critical and non-critical regions. The boundaries of these regions are piecewise polynomial curves. The various regions formed by the decomposition process are connected by a graph and this graph is searched for a path. The algorithm is for two-dimensional problems, is complete but not optimal, and runs in $O(N^2 \log N)$ time.

#### *3.2.3 Cylindrical Algebraic Decomposition*

This more complex decomposition extends the critical-curve decomposition to three-dimensional problems. It bisects parts of the free space using critical surfaces. It is complete but not optimal, and runs in double exponential time.

#### *3.2.4 Connected Balls in Free Space*

This approach is designed to deal with un-structured obstacle fields, and operates by filling free space with overlapping balls (for instance, spheres are balls in three-dimensional Euclidian space) that are totally in free space.

Vandapel et al. [117] introduces unions of free-space balls as a roadmap in multi-dimensional space.

## 3.3 Approximate Cell Decomposition

### *3.3.1 Rectanguloid Cell Decomposition*

This divides the entire configuration space into rectanguloid regions, and labels each rectanguloid as being completely filled (black), partially filled (grey), or completely empty (white). It is proven to be resolution-complete.

The most common example is that of the A* or D* search over a square or cubic grid of occupied or unoccupied cells. Ferguson et al. [33] reviews this type of approach, with a focus on dynamic problems.

### *3.3.2 $2^m$ Tree Decomposition (Quadtree or Octree Decomposition)*

This decomposition is designed to reduce the number of points needed to represent obstacles as compared to a full grid representation.

This type of representation is becoming increasingly more common, and several new papers using tree decompositions have been published. Sinopoli et al. [102] uses wavelet transform processing for path planning purposes. Behnke [5] proposes a quadtree algorithm with weights put in to avoid obstacles by a longer distance. Soucy and Payeur [105] compares a fixed resolution vs. quadtree characterization for similar problems. Tsenkov et al. [113] describes a real-time implementation of planning over a quadtree representation of obstacles, demonstrated on an unmanned helicopter.

## 3.4 Approximate and Decompose

This decomposition is similar to the trapezoidal decomposition, but replaces the triangular end regions with rectangular mixed regions. This approach reduces the proportion of mixed area in comparison with a grid decomposition with mixed cells.

## 3.5 Potential Field Methods

Potential field methods are based on the idea of assigning a potential function to the free space, and simulating the vehicle as a particle reacting to forces due to the potential field. The goal point has the lowest potential, and attracts the vehicle, while obstacles repel the vehicle. Since their initial publication Khatib and Mampey [59], potential field methods have been generally known for being of low computational complexity but incomplete. However, a potential field which has the properties of a navigation function [92] makes a complete path planner. There are two classes of potential fields known to satisfy properties of a navigation function: those based on harmonic functions [27] and those based on solving the optimal distance-to-go function [42]. These methods require, however, discretizing the configuration space into a grid with $M$ points, and this discretization scales as $O(M^D)$ with the dimension $D$ of the configuration space. The added advantage of a navigation function is that it can be used to provide direct feedback control, rather then relying on feed-forward control, as traditional trajectory planners do. A single navigation function produces a trajectory for every possible starting point in the configuration space.

### *3.5.1 Potential Field with Gradient Descent (Virtual Force Field—VFF)*

This is the original potential field approach, and is designed to run quickly. It assigns a decaying function to the goal point with a negative minimum value, and a decaying function to each of the obstacles with a positive maximum value, and sums the functions from the goal and all obstacles to get the total potential.

Since the VFF algorithm is sometimes used directly for trajectory generation and is valid for an arbitrary number of dimensions, some references are included here. Khatib and Mampey [59] is cited as the first use of the potential field method, and for many years the term "potential field" applied strictly to this algorithm. Borenstein and Koren [12] and Borenstein and Koren [13] couple sensing with planning in an evidence grid, using the VFF method to provide motion planning inputs for ground robots. A reactive method, based heavily on localized sensor input and immediate response, uses a grid with VFF. Yoram Koren and Johann Borenstein [63] discusses inherent limitations of potential methods, but the contents of this article apply only to the VFF algorithm. The limitations mentioned in this article, trap situations due to local minima and no passage between closely spaced obstacles, do not apply to potential fields that are navigation functions in the sense of Rimon-Koditschek. The oscillations cited here result from poor control system design, and can be eliminated by applying classical control theory. Ahuja and Chuang [2] publish a generalized potential field based on summed Poisson solutions of simple cases.

### *3.5.2 Potential Field Guided Search (Depth-First, Best-First, Variational Planning—Arbitrary Potential Field)*

This approach is designed for potential fields that have local minima. Rather then use gradient descent, which is easily trapped in local minima, a search that is complete in the resolution or probabilistic sense is used. This can be considered as being similar to an A* search with the simple heuristic replaced by a potential field, although this approach is somewhat more general since it admits depth-first searches as well. The variational planning approach uses the potential as a cost functional, and attempts to find a path to the goal point that minimizes this cost.

### *3.5.3 Harmonic Potential Functions*

This class of functions is based on solving a partial differential equation with a Laplacian term. These equations include Laplace's equation, Poisson's equation, the conduction heat flow equation, approximations to Navier–Stoke's equation, and other partial differential equations of this type. While not producing an optimal path, these equations generate functions that are true navigation functions in the sense of Rimon and Koditschek, meaning they are smooth, have only one local minimum that occurs at the goal point, potential obtaining a constant maximal value at the boundaries of obstacles, and has a non-degenerate Hessian at each critical point of the function. This implies that these functions may be used to produce a complete planner using a simple gradient descent search. This type of function needs to be solved by a numerical method with global information, and is generally solved on a grid.

The first publications of this method are in Akishita et al. [93] and Connolly et al. [27]. Connolly et al. [26] provides further theory into the dynamic aspects of the approach, including a Hamiltonian framework to model inertial effects, and allowing orbits around a goal point. Zelek [124] makes use of the harmonic potential field, and [107] use solutions the diffusion equation as potential fields. Kazhdan et al. [83] describes Poisson surface reconstruction: while not a path planning paper, it describes numerical methods useful for solving the three-dimensional partial differential equation. Scherer et al. [94] describes the results of this method actually flown on an unmanned rotorcraft, and combines a harmonic potential planner with reactive planner.

#### *3.5.4 Continuous Dijkstra (Optimal Navigation Function Using Visibility Polygons)*

The exact optimal navigation function can be generated using this method for two-dimensional problems, and runs in $O(N^{5/3} \log N)$ time. This method divides the space into visibility polygons, each encoding "wavelets" rooted at a polygon vertex, and searches them. However, it cannot be extended to higher-dimensional problems.

#### *3.5.5 Wavefront Expansion (Dynamic Programming)*

This is the grid-sampled version of the Continuous Dijkstra method, and can be used in multiple dimensions, running in $O(M \log M)$ time. It is also closely related to the complete search over a grid using dynamic programming. It is able to produce a path that is complete and optimal in the resolution sense.

#### *3.5.6 Wavefront Expansion with a Skeleton (NF2)*

This is an obstacle avoiding navigation function that plans paths along the medial axis rather then nearly optimal paths, and also runs in $O(M \log M)$ time. The advantage of this is that the paths do not graze obstacle, but rather avoid the nearest obstacles by a maximal amount. Thus it is more suitable for practical applications, where the position of the vehicle or obstacles may not be know with perfect certainty.

### 3.6 Probabilistic Approaches

#### *3.6.1 Randomized Search in a Potential Field to Avoid Being Trapped in Local Minima*

This method is designed to help a gradient descent search routine escape from local minima that are found in the original potential based method. There are various approaches to randomizing the search, such as adding random components to subsequent steps or changing the values of obstacle potentials at random. Completeness of these algorithms has been difficult to prove.

Caselli et al. [20] uses a heuristic stochastic search (instead of a random walk) to escape local minima.

#### *3.6.2 Potential Field with a Global Optimization Search or Learning Algorithm*

The field of global optimization has several algorithms that are able to solve an optimization problem with local minima. Any of these may be applied to a potential field with local minima in order to find the true solution.

#### *3.6.3 Probabilistic Roadmap (PRM)*

This approach is important because it admits a solution to problems of arbitrary complexity and dimension that is convergent in the probabilistic sense. It is generally used in manipulator problems, where the configuration space is typically of high dimension and complex. However, the rate of convergence is slow, and the paths it produces are not optimal. In the case of two- and three-dimensional configuration space planning, the resultant paths are required to be smoothed significantly before a trajectory can be formed. It has particular difficulty in converging in problems that have long passageways.

Kavraki et al. [57] initially describes this algorithm. Mazer and Ahuactzin [76] combines random landmarks with a local search. Kavraki et al. [56] discusses theoretical properties of PRM algorithm. Bohlin and Kavraki [11] outlines the Lazy PRM, a modified method that overcomes some of the simple PRM's limitations.

### 3.7 Weighted Region Problem

In this problem, the configuration space has an associated weight function. The path planner has to minimize to weight integrated over a path.

#### *3.7.1 Exact Algorithm for Polygonal Weighted Regions*

This algorithm uses results of variational calculus used to derive Snell's law of refraction. It is capable of finding the exact path over weighted polygonal regions. While operating in polynomial time, it is expensive, requiring $O\,(N^8 \log N)$ operations [82].

#### *3.7.2 Approximation Algorithm for Polygonal Weighted Regions*

The approximation algorithm for this problem is resolution optimal, and can run in $O\,(MN \log MN)$ operations [88]. It divides the edges of the polygons into shorter segments, and finds a path connecting these vertices.

#### *3.7.3 Exact Algorithm for Weighted Grids*

This is a simpler problem then general polygonal weighted regions. It is simply a discrete search over a grid, and can be accomplished in $O(M \log M)$ time. It is complete and optimal in the resolution sense.

Recently, Ikonen and Toivanen [51] discuss grey-level distance transforms for shortest paths on curved surfaces and their approximation using dynamic programming, and Ikonen [50] discusses the priority queue implementation and compares computational complexity with a previous iterative approach.

## 4 Trajectory Planning with Differential Constraints

Most trajectory planning problems relevant to today's UAV applications have to be considered as dynamics-constrained problems. The behavior of aerial vehicles is often not sufficiently well approximated by their kinematics (as is more often the case in ground vehicles). Taking into account the equations of motion is directly relevant to guaranteeing the soundness of the planner, since approximating the

dynamics solely through a kinematic model with constraints will lead to overly conservative models. The equations of motion are also relevant in details of the vehicle maneuvering affecting energy or duration of the trajectory. For example Kong and Mettler [62] show that performance criteria have a significant effect on the resulting trajectories and heuristic methods (e.g. minimizing distance) are not able meet specific performance requirements.

This class of planning problems is substantially more difficult to solve due to the dependency between time and the state-space introduced by the differential constraints. Even in the trivial case of connecting two states in a configuration space without obstacles, an exact solution is generally not possible. An exact solution is available for two-dimensional problems only, solvable in exponential time and polynomial space [54]. However, this approach cannot be extended to three-dimensional problems. For applications requiring a vehicle to navigate among obstacles or complex terrain, algorithms that exploit some form of approximation or heuristic are necessary, not only for the merit of finding a feasible or sub-optimal trajectory but also for the need to negotiate with hardware capacity. Solutions to this class of problem represent a newer research area where very few approximation bounds or benchmarking results have been proposed.

A tabulated overview of these algorithms is given in Table 2. Blank table cells correspond to properties of algorithms that cannot be determined from the literature. For a more detailed breakdown of potential-based methods, refer to Table 1.

## 4.1 Sampling-Based Trajectory Planning

### *4.1.1 Grid-Based State Space Search*

This method played an important role because it establish proof for the completeness and optimality (in the resolution sense) for the case of a polynomial approximation to the problem with dynamic constraints, which are represented in the form of velocity and acceleration bounds. Furthermore, it defines an arbitrary speed-varying safety corridor, making this particular algorithm one of very few trajectory planning algorithms with a proven explicit safety guarantee. The way this algorithm works is that it discretizes the entire state space of the vehicle onto a lattice, and searches the lattice for the time-optimal path that satisfies the safety corridor. Although the algorithm converges as a polynomial of the number of obstacles, it is a high-order polynomial that is exponential with the number of dimensions, making practical real-time implementation difficult due to high dimensionality of the state space. It also has difficulty in solving planning problems for the case of under-actuated vehicles, which are quite common in application.

Donald [31] is the first publication of this algorithm. Donald and Xavier [32] gives a proof of a lower complexity bound for the same algorithm. Reif et al. [89] uses nonuniform discretization to reduce the algorithm's complexity.

### *4.1.2 State-Space Navigation Function with Interpolation*

Much like the grid-based state space search, this method approximates the time-optimal path to a goal. Instead of returning only a single trajectory, it returns a navigation function over the state space. This navigation function can be computed by either value iteration or control policy iteration, although value iteration is more

**Table 2** Comparison of algorithms for trajectory planning with differential constraints

| Algorithm name | Completeness | Optimality | Soundness | Proven time complexity |
|---|---|---|---|---|
| **Canonical two-dimensional solution** | Complete | Optimal | Sound | $O(\exp(N))$ |
| **State-space sampling** | | | | |
| State space lattice search | Resolution complete | Resolution optimal | Resolution sound | $O(C^{\mathrm{d}} N \mathrm{err}^{(-6\mathrm{d})})$ |
| Dynamic programming with interpolation | Resolution complete | Resolution optimal | Resolution sound | $O(C^{\mathrm{d}} N \mathrm{err}^{(-3\mathrm{d})})$ |
| Rapidly-expanding random tree (RRT) | Probabilistically complete | Non-optimal | Resolution sound | |
| Reachability graph, rapidly-expanding dense tree (RDT) | Resolution complete | Resolution optimal | Resolution sound | |
| Pruned reachability graph | Resolution complete | Resolution optimal | Resolution sound | |
| **Minimum distance path followed by trajectory forming** | | | | |
| Linked boundary point solvers | Resolution complete | | | |
| Maneuver automation | Resolution complete when complete search used | | Resolution sound if constraints are met | |
| Decoupled approach | | Non-optimal | | $O(\text{polynomial}(N))$ |
| Plan and transform (spline smoothing) | | Non-optimal | | |
| Path-constrained trajectory planning | Resolution complete when complete search used | Non-optimal | Resolution sound | |
| Plane slicing | Resolution complete | Non-optimal | Resolution sound | |
| **Mathematical programming** | | | | |
| Gradient-based trajectory optimization | Resolution complete when complete search used | Resolution optimal when complete search used | Resolution sound (for infinite horizon) | $O(\exp(M))$ |
| Model predictive control (receding horizon) | | | Resolution sound if constraints are met | |
| **Potential-based** | | | | |
| Path-constrained trajectory planning | Resolution complete when complete navigation function is used | Non-optimal | Resolution sound if space varying speed limit constraints are met | $O(M \log M)$ |
| Randomized potential fields | | Non-optimal | | |

popular. Bertsekas [6] gives details in a more general sense. For any given state, performing gradient descent on this navigation function will produce an approximately time-optimal trajectory to the goal. Interpolation between lattice points allows a continuous function which can be used for feedback. The algorithm takes on the same order of complexity as the grid-based state space search. A recent paper which employs this method can be found in LaValle and Konkimalla [68].

#### *4.1.3 Rapidly-Expanding Random Tree (RRT)*

This works by using a stochastic search over the body-centered frame of reference, and expanding a tree through a random sampling of the configuration space. This algorithm is proven to be complete in the probabilistic sense, and to produce a trajectory that is feasible given the dynamic constraints of the vehicle. However, there is no proof of the convergence rate or of optimality. In fact, it is known that certain types of problems, such as those with long passageways, may have very slow convergence rates. However, in these cases gradient based optimization methods can be applied to reach locally optimal solutions.

LaValle [67] is the first publication describing the RRT, followed by a more detailed report [71]. In these two papers, the vehicle is considered holonomic; neither dynamics nor kinematic constraints are considered. Frazzoli et al. [36] extends the RRT to systems with dynamics by reducing the whole system to finite states possible using a finite automaton. LaValle [69] compares dynamic programming and RRT algorithms. Frazzoli et al. [37] gives a more extensive description to the RRT method, and Redding et al. [87] provides an application of the RRT: this approach uses the RRT in combination with a Dijkstra search based path refinement step.

#### *4.1.4 Reachability Graph*

This approach also uses a body-centered frame of reference: for each state, the tree explores variety of states including the maximum deflection states. The combinatorial complexity of such a process is often prohibitive (in fact, exponential with $M$), and the tree quickly fills the space close to the initialization point. The basic approach has been employed for curvature-constrained path problem in two dimensions [70].

Another way to make this approach tractable is to use cell-based pruning: the configuration space is divided into cells, and the reachability graph is set up to be a tree that has no more then one leaf ending in each cell [70].

### 4.2 Decoupled Trajectory Planning

#### *4.2.1 Minimum Distance Discrete Path Followed by Trajectory Forming (Two-Step Approach)*

The algorithms in this section follow the same general approach: first a discrete path through the configuration space is found by one of the algorithms, and then the resulting path is used as the basis for the generation of a trajectory that is feasible for the dynamics-constrained vehicle. For the first stage, a well-known algorithm such as A* is applied over a grid, the PRM, or the Voronoi approach is typically used. Complexity in this approach is typically dominated by the path planning phase computations.

This decomposition-based approach allows efficient computation of approximate solutions, but makes proofs of completeness, optimality, or even of soundness, difficult. There is no safety corridor built in and soundness requires checking whether the trajectory intersects with obstacle space, and re-computing or rejecting trajectories that do not pass. Generally, practical implementations of these algorithms will use a conservative safety corridor to prevent collisions.

#### *4.2.2 Discrete C-Space Search Connected by Two-Point Free-Space Boundary Value Solver*

In this approach, a set of waypoints is first selected (generally by a grid-based search), a velocity is assigned to each one, and a boundary-value problem connecting each waypoint to the next point is solved. Although these boundary-value problems don't have to deal with obstacles, a general simple solution is not possible, so the solution is generally approximated using numerical methods. As with the previous method, an explicit collision check on the trajectory is needed to ensure soundness, possibly at the expense of completeness.

#### *4.2.3 Hierarchical Decoupled Planning and Control*

This is a general strategy that is used in most practical applications. Since feedback control is typically required for reliable operation of air vehicles and most traditional algorithms provide only feed-forward solutions, the hierarchical decoupled planning and control approach provides a straightforward way to integrate a waypoint planning algorithm and the vehicle control system. The overall hierarchic system involves open- and closed-loop controllers operating at a variety of rates, linked together from top to bottom. The outer, open loop consists of a discrete search that produces a set of waypoints leading to the goal while avoiding obstacles. The second open loop level smoothes this set so that the waypoints are feasible given the vehicle's velocity and acceleration limits. The third open loop level generates a timing function along the trajectory, and creates a set point (or "rabbit") that moves through space, and last, the inner loop is a closed-loop tracking controller that attempts to minimize the error between the vehicle and the rabbit.

This approach has been popular for UAVs because it is relatively easy to implement since it requires no sophisticated solvers and is made of a hierarchy of modules with well defined functions. Since many unmanned aircraft are often already equipped with an inner-loop tracking controller it can be easily adapted to a variety of vehicles. In addition, the majority of UAVs currently in production already have a waypoint following system (without obstacle avoidance) as the primary means of control—this makes the multi-level decoupled control approach easiest to integrate with the existing control scheme. However, there is no proof demonstrating in general that this method is sound, complete, nor that it produces near-optimal paths. Evidence of a sound planner needs to be produced on a case-to-case basis for safe use of this approach.

Boyle and Charnitoff [14] covers the closed-loop section of the planner, or the maneuver tracking controller. Judd and McLain [55] describes a Voronoi-based planner followed by spline smoothing for trajectory formation. Yang and Zhao [122] is based on A* coupled with higher order heuristics. Suzuki et al. [108] uses A*, followed by direct optimization of the trajectory, using an RTABU search. Scherer

et al. [94] uses an evidence grid with a Laplacian-based potential method as the outer loop, a reactive planner (dodger) to enforce soundness, a speed controller to convert the path into a trajectory, and an inner loop flight controller. Kim and Bhattacharya [60] is based on a modified visibility graph roadmap method that is followed by finite horizon optimal control. Takahashi et al. [110] covers the design and characterization of the inner-loop control law used in such a multi-level decoupled controller for an unmanned rotorcraft based on two types of path planners (quasi-3d implementations of an A* and a Voronoi-based planner). Howlett et al. [44] describes the implementation of the two path planner modules.

#### *4.2.4 Discrete C-Space Search Interpolated with Polynomial Arcs*

In this approach, an ordered set of waypoints produced by a discrete planner is fitted with a spline made up of polynomial arcs. This spline is set up so that the vehicle can follow it without violating acceleration constraints, and typically consists of circular arc segments with a minimum radius and straight segments. Other types of spline, such as the Pythagorean hodograph, have been proposed for the same purpose.

Yakimenko [120] uses optimization on a family of polynomials to approximate the two-point boundary value problem solution, which can be used for interpolating between states. Shanmugavel et al. [99] describe Pythagorean hodograph interpolation. Both approaches use optimization to determine the spline that gives best performance based on minimal time or minimal control energy.

#### *4.2.5 Curvature-Constrained Paths with Speed-Control Planning*

This is coupled with the previous approach: once a path is produced that is curvature constrained, it is possible to optimize the speed so that the vehicle will follow this path in minimal time. This path-constrained trajectory planning problem requires solving only a two-dimensional path-constrained state space (with time and velocity axes), and can be accomplished efficiently.

Slater [104] describes an approach for helicopter short trajectories based on guidelines for piloted flight. Yang and Kapila [121] considers curvature constrained paths based on the Canny's exact algorithm for two-dimensional problems.

#### *4.2.6 2-D Voronoi Solutions from Multiple Body-Based Planes*

In this approach, several planes containing both the initial and goal points are extracted as subsets of the three-dimensional configuration space. These planes that are discriminated from each other by a single angle—this angle is discretized so there are a finite number of planes (in experiments, it was found that four to six planes suffices). These planes are then searched with a two-dimensional path planner, such as a Voronoi roadmap planner, and then rated against each other according to optimality criteria. The best of the set is chosen.

Howlett et al. [45] provides a basis by describing the 2-d Voronoi approach, and Whalley et al. [119] describes the Voronoi-based plane slicing method with multi-level control for an unmanned rotorcraft with multiple-target surveillance ability.

### 4.3 Finite-State Motion Model: The Maneuver Automaton (MA)

The general idea of finite state models is to reduce the optimization or search problem from an infinite-dimensional space to a finite one. It helps to significantly reduce the computational complexity of a trajectory optimization.

There are two primary type of finite-state models for dynamics systems: the first is a discrete-time model with quantized states (quantization); another choice is to relax the restrictions on control and time and instead use operations over discrete stages with fixed start and end states. These stages, which are feasible time-parameterized curves in state space, are called motion primitives.

In the context of vehicle trajectory planning, this model is called a maneuver automaton (MA). The concept of MA for vehicle is based in part on the observation that human pilots achieve agile control using a combination of trim trajectories and maneuvers (non-equilibrium transitions between trims).

While Agarwal and Wang [1] is not exactly a MA, but is based on Canny's exact algorithm with fixed-radius segments, it includes some ideas similar to the MA. Yakimenko [120] stores a set of solutions to the two-endpoint boundary value problem as a motion primitive set (but does not deal with obstacles). Piedmonte and Feron [85] and Gavrilets et al. [40] investigate the concept of maneuver automaton for human piloted acrobatic flight. Frazzoli et al. [35] provides a rigorous definition of the concept of MA within the context of autonomous guidance. In the basic MA form the set of trim and maneuvers are used to pre-compute a cost-to-go map. This map can be used online with a greedy guidance policy. States falling between the pre-computed values are obtained via interpolation. Mettler et al. [78] provide a simulation example of this MA based guidance policy for online rotorcraft guidance. In this form of guidance policy, the vehicle behavior is constrained to the set of primitives in the MA. Hence, for agile vehicles, it can be an issue to achieve a sufficiently expressive MA due to the "curse of dimensionality". To relax the vehicle behavior and provide flexibility in an obstacle-rich environment, Schouwenaars et al. [96] use the concept of the MA within a receding horizon optimization framework. Instead of fixed trim trajectories, the trims are replaced by controllable linear modes. This model is a more faithful emulation of the human control strategy. The maneuvers are still open-loop trajectories used to transition between modes or for a tactical purpose (e.g. split-S maneuver can be used to reverse direction of flight without lateral displacement). Dever et al. [30] extends this framework to allow interpolation between maneuver boundary conditions within a class of maneuver. This provides additional flexibility for the initiation and completion of maneuvers. In the meantime, the MA has also been used to generate a state-dependent cost-to-go map for a receding horizon planning [81]. This will be discussed in Section 4.5.

Parallel to the maneuver automaton concept, a similar idea called "control quanta" is introduced for driftless systems with a symmetry property [72]. For this special class of systems, by employing control quanta, the reachable set can be restricted to a lattice. And by choosing a suitable set of control quanta, the reachable set can be everywhere dense in the limit when $M$ approaches infinity. The difference is that for the control quanta method, the control policy is chosen from a collection of control library policies, while for motion primitive method the trajectory is chosen from a library of maneuvers that can result from a various control strategies.

## 4.4 Mathematical Programming

Mathematical programming methods treat the trajectory planning problem as a numerical optimization problem. Some popular methods include Mixed Integer Linear Programming (MILP), nonlinear programming, and other constrained optimization approaches. These methods are also known as trajectory optimization methods, since they find a trajectory to a goal point that is optimal in the resolution sense. However, the cost functions typically have a number of local minima, thus finding the global solution strongly depends on the initial guess (the general formulation is NP-hard, although given an initial guess sufficiently close to the global solution, the optimization converges in polynomial time).

For this type of problem, one standard strategy it to enforce the equations of motion as constraints. An earlier review of this method can be found in Betts [10]. Another strategy is to discretize the variational principles underlying the systems dynamics, such as Hamilton's principle or Lagrange–D'Alembert principles, and then these discrete equations can serve as constraints. This kind of strategy is called Discrete Mechanics and Optimal Control (DMOC) and can be found at Marsden and West [75] and Junge et al. [53]. Kobilarov et al. [61] extends the framework to deal with obstacles. Several approaches have been used to break this into simpler problems.

### *4.4.1 Initialization of Mathematical Programming Methods (Infinite Horizon Control)*

An initial trajectory for the mathematic programming methods, such as a constant-speed trajectory, can formed using a discrete search over the configuration space. These waypoints are then used as an initial point in the mathematical programming search. If this initial point falls within the basin of attraction of the global solution, then the mathematical programming approach can find the optimal solution in polynomial time. However, unless care is taken in finding proper initial points, the solution could fall into a local minimum, and general global optimization approaches guaranteed to find the global minimum are prohibitively expensive.

Toussaint et al. [112] describes this approach and Richards and How [91] covers both single and multiple-vehicle planning in two-dimensional cases. Milam [74] produces trajectories for a constrained ducted fan setup with two degrees of freedom and provides flight test results. Carlyle and Wood [19] describes the Lagrangian relaxation solution. Menon et al. [77] give an application of direct optimization used in aerospace: they produce fuel-optimal periodic cruise trajectories involving high Mach numbers and allowing periodic large changes of altitude. Carlyle and Wood [19] describes again the Lagrangian relaxation approach, with the addition of risk avoidance (UAVs are required to avoid SAM sites). Keith et al. [58] describes how a trajectory planning problem within complex terrain can be converted into a MILP problem by representing terrain as a piece-wise affine function.

## 4.5 Receding Horizon Control (Model Predictive Control)

Receding horizon control (RHC) or model predictive control (MPC) solves the numerical optimization problem over a reduced time horizon. In this approach, an open-loop control policy is designed to control the vehicle until the end of

the time horizon. Optimization over a finite horizon requires reduced computation time, however, it will not converge to a globally optimal solution without using an appropriate cost-to-go function to capture the discarded portion of the trajectory. Except in trivial cases, optimality and completeness are difficult to prove. In UAV guidance applications, this approach has often been used with a MILP solution, however, there is no reason to restrict the receding horizon control method with this type of numerical solver.

Jadbabaie [52] introduced receding horizon control to solve trajectory planning problems for general nonlinear systems in an obstacle-free environments. The paper also provides necessary conditions for the stability of the RH scheme by employing concept of control Lyapunov function. The RHC-based approach is used in Singh and Fuller [103] to navigate a vehicle with nonlinear dynamics through a toy urban environment with a vector of known way-points in a decoupled manner: first convert the problem into a convex optimization problem by linearizing the vehicle model and obtain a nominal trajectory, then generate a series of overlapping convex regions around the trajectory and finally within these feasible convex regions open-loop trajectory is updated by RHC as time evolves.

The features of RHC makes it a suitable trajectory planning technique for many UAV applications. Sensory information can be incorporated into on-line computation thus it can deal with uncertainty; at the same time, only local information is integrated thus it can reduce computational effort. However, properly designed terminal cost function needs to be provided to the on-line planner to guarantee completeness and near-optimality. For instance, Schouwenaars et al. [97] uses a cost function based on a visibility graph and [7] estimates the cost function by searching a graph representation of the environment with Dijkstra's algorithm. Schouwenaars et al. [97] also investigates the effect of the length of planning horizon on the computation time and optimality. Schouwenaars et al. [96] and Shim and Sastry [100] investigate hardware and software implementation details and also provide experimental flight-test results for a fixed-wing aircraft and a rotorcraft vehicle in various 2D guidance missions. Mettler and Bachelder [79] describe a receding horizon scheme in 3D with visibility constraints. Frew [38] describes the way to apply the Fisher information matrix to integrate passive, non-cooperative sensory information into the RHC framework. Prazenica et al. [86] estimates the obstacle map from local visual data by using an adaptive learning algorithm so as to avoid unknown obstacles in an urban environment and uses RHC to generate a trajectory and control strategy.

The value function captures the relationship between the vehicle dynamics (state), the environment and the cost.

Without a cost-to-go function which provides an sufficiently good approximation of the value function associated with the trajectory optimization, important aspects pertaining to performance can be lost. This may be acceptable for vehicle with simpler dynamics but will cause a gap in performance for highly agile vehicles. For example, techniques like the visibility graph do not take the vehicle state into account and therefore cannot capture the spatio-dynamic relationship. Toupet and Mettler [80] approximate the state-dependency using a multi-scale environment decomposition. Toupet and Mettler [111] implement and flight test the receding horizon planner with an improved CTG on a surrogate A-160 rotorcraft. Mettler and Kong [81] use the concept of MA to compute state-dependent cost-to-go map for a receding horizon planning. Dadkhah et al. [28] describes the implementation

and experimental evaluation of this planner and highlight the shortcomings of approximate CTG functions that do not take into account state information.

## 4.6 Other Methods and Issues

### *4.6.1 Ad Hoc Receding Horizon Planning*

The advent of more agile robotic platforms has highlighted some limitations of classic robot motion planning techniques. In the Dynamic Window Approach (DWA) [34], a robot's translational and rotational velocity is computed periodically by optimizing a measure of distance to the goal. The velocities are selected from a finite set of admissible velocities, which is determined based on the proximity of immediate obstacles.

The concept was extended to prevent local minima [15] by combining the DWA with a global path planner based on an occupancy grid. Further extensions were implemented to guarantee the stability of the system [84]. Finally, in Hwangbo et al. [49], a similar technique was applied to aerial vehicle motion planning. These latest techniques are conceptually similar to receding horizon optimization but are ad-hoc in their formulation and implementation. These examples can be viewed as a testimony of the convergence of the robotics planning and control concepts.

### *4.6.2 Potential Field Methods and the Navigation Function*

Just as in solving a problem with unconstrained dynamics, the potential field can be used to serve as a controller. The same properties apply here: the function is generally simple to compute, but may result in an incomplete planner, and is non-optimal in general. If the potential field is designed properly, it may be used directly as part of a feedback controller. However, if used in that way, care needs to be taken so that the feedback controller is stable. Furthermore, to use such a method, there usually exist certain constraints on the form of the vehicle dynamics.

Such a method is first proposed by Conner et al. [23]: the free configuration space is first decomposed into convex cells and then local control policies are designed for each cell to respect dynamic constraints. The convergence is proved for a double integrator. Belta et al. [9] uses the same idea to solve planning problem of a vehicle whose dynamics can be represented as an affine system within a polyhedral environment. Such configuration space division techniques also enable a marriage of control method and powerful logic-based AI methods as shown in Belta et al. [8]. Conner et al. [24] further extend the idea to convex-bodied nonholonomic wheeled vehicles.

### *4.6.3 Planning in the Presence of Uncertainties*

The general problem of planning with uncertainty can be phrased as follows: Given a vehicle with uncertain position information, uncertain environment knowledge (e.g. obstacle locations), and having limited precision in tracking commands, find the best path to the goal.

While the idea of uncertainty in planning has been around for a long time, it has traditionally been considered in the domain of compliant control, where the robot is allowed, or even required, to touch obstacles. The type of situations considered are relevant to problems involving manipulators and gripping, but are not of much use to

vehicle motion planning, where contact with an obstacle is to be avoided at all costs (with the exception of landing or perching). In most of the real world UAV planning problems, the issue of uncertainty in sensing and control is unavoidable.

Connolly [25] presents the Laplacian potential field as a solution to the problem of minimizing collisions of a random walk with an obstacle field. Lazanas [3] solves the problem exactly given regions where there is no uncertainty. Schouwenaars et al. [95] describes a robust Maneuver Automaton which guides the vehicle by taking explicitly into account the uncertainty in the maneuver outcome. Zengin and Dogan [125], which solves the problem of approaching a goal while avoiding SAM sites, uses a state-space search. The RHC framework described earlier can also be employed to deal with uncertainties: the off-line, pre-planned trajectory or approximate cost-to-go function accounts for global convergence based on known knowledge and the online RHC can used to negotiate with mid-flight uncertainties. For instance, in Kuwata et al. [65], the RHC is used to generate trajectories for a vehicle operating in an environment with atmospheric turbulence.

The field of Simultaneous Localization and Mapping (SLAM) is important in problems of planning with uncertainty. Although the SLAM problem is closely linked with the problem of motion planning with uncertainty, it doesn't immediately address the motion planning issue, so it is not covered here in any detail. Thrun [114] gives a survey of SLAM techniques [115], describes a SLAM example for helicopter flight, and Thrun et al. [116] shows what part SLAM played in their DARPA Grand Challenge entry.

### *4.6.4 Reactive Planning*

The term "reactive planning" refers in general to a broad class of algorithms that use only local knowledge of the obstacle field to plan the trajectory. Reactive algorithms are important in dealing with uncertainty, and run very quickly since no elaborate couplings are involved. In the case where a global obstacle map is not available and obstacle positions are known only within a small radius, a reactive algorithm prevents last-minute collisions by stopping or swerving the vehicle when an obstacle is known to be in the trajectory, which has been planned by a different algorithm. This type of approach is important in many existing practical implementations in order to "patch" an unsound algorithm to ensure that it is sound, as well as to deal with obstacle fields that may change suddenly. However, reactive planners, due to their inability to take the global planning problem into consideration, are seldom used as the sole trajectory generation process. In other words, if only the reactive planner is used, the vehicle may never find a trajectory that will lead to the goal, let alone an optimal one.

The Motion Description Language (MDL), described in Brockett [16], and its extension MDLe [73] can be used to define and design the reactive algorithms. In this framework, a sensor based interrupt (i.e. obstacle detected) will cause the vehicle to switch to another behavior. Hristu-Varsakelis et al. [46] proves that MDLe is formal language.

Zavlangas et al. [123] uses fuzzy logic as a basis for a reactive algorithm. Hui-Zhong et al. [47] describes a different approach, and [43] describes an algorithm that uses learning from the examples of human operators to improve the reactive planner. Call et al. [17] describes visual flow based reactive planning, and Spenko et al. [106] uses a planner based on a high fidelity ground vehicle dynamic model. Geyer and Johnson [41] uses a reactive algorithm in based on laser scanner data in

local coordinates. There are numerous ongoing works on this topic, especially bio-inspired reactive planning algorithms. This can be explained due to the deficiencies of existing planning algorithms to involve sensory information in a principled way so that a complete planning framework results.

## 5 Discussion

In this paper we provided a survey of published motion planning techniques. This survey emphasizes practical methods and provides a general perspective on the particular problems arising with UAVs. UAVs belong to a class of vehicles for which velocity and acceleration constraints are both significant. More agile UAVs even require taking into account the higher order differential constraints associated with the equations of motion, or accounting for aerodynamic effects.

While efficient algorithms exist that are well-characterized for simpler sub-problems, such as the problem of motion planning for vehicles not bound by dynamic constraints, an exact solution to a typical UAV problem is often not trivial. Algorithms that can solve these planning problems are frequently too expensive to be used in real time, and when they are tractable, they are not proven to be complete, sound, or optimal.

Given these difficulties, one conclusion of this survey is that a solutions must be chosen to specifically fit the characteristics of the particular planning problem. In some cases, aspects that at first may seem challenging like partial knowledge of the environment or disturbances, can in reality provide opportunities, or at least rationales, for certain forms of approximations.

From the reviewed literature, we find that the issue of uncertainties and robustness in general has not been studied much. Therefore understanding these effects represents a fundamental aspect of determining practical algorithms that are simultaneously computationally efficient, optimal and robust. If we consider feedback control systems, we get a general ideas on how uncertainties or partial knowledge can be addressed by an algorithm. We also see that often relaxing the complexity of the controller can contribute to robustness.

When reviewing the different papers, we find that the majority of the methods surveyed here do not include much discussion about practical implementation. When they do, they often only provide simulation results based on idealized vehicles and operational conditions. So it is inevitable that theoretical work often leaves out important issues. In particular, some of the more expensive, complete algorithms have, to date, never been implemented in code. Among the papers surveyed, most of the practical implementations of UAV guidance have been of the hierarchic decoupled control type, or in some cases focus only on the reactive planning needed to avoid obstacles. With the advent of smaller and faster microcontrollers, more sophisticated planners are beginning to be implemented for real-time guidance of vehicles.

The reality is that it is often going to be impossible to generate provably complete or optimal algorithms for problems with differential constraints, therefore more work needs to take place on the development of benchmarks needed to compare algorithms and provide some design standards. Benchmarking will give researchers a way of determining the ability of an algorithm to approximate an optimal solution as well

as an evaluation of its computational complexity. They also provide opportunities to develop more knowledge about the characteristics of the planning problem itself, which ultimately will be most important in the optimality and complexity and the choice of method.

Finally, not all tasks are simple goal-directed guidance problems, but the overall guidance behavior results from the interaction of the vehicle with the environment. In many cases, the information gathered by the UAV about the environment (and the mission) will be useful to improve the trajectory. In robotics these interactive problems are most often found in the context of simultaneous localization and mapping (SLAM). For UAVs there is a stronger emphasis on the system dynamics, thus simultaneous mapping and planning (SMAP) may be more appropriate. Sensors and perception may play as much of a role as the control system or the dynamics in dictating the behavior and performance of the autonomous guidance system. Therefore, future work will have to encompass all of these aspects to achieve a realistic and comprehensive understanding of the algorithms and the development of design principles that will achieve reliable performance and robustness.

**Acknowledgements** This work owes its existence to the funding from the Army Aeroflightdynamics Directorate, and collaboration with the ongoing Autonomous Rotorcraft Project.

## References

1. Agarwal, P.K., Wang, H.: Approximation algorithms for curvature-constrained shortest paths. SIAM J. Comput. **30**(6), 1739–1772 (1996)
2. Ahuja, N., Chuang, J.: Shape representation using a generalized potential field model. IEEE Trans. Pattern Anal. Mach. Intell. **19**, 169–176 (1997)
3. Lazanas, A., Latombe, J.C.: Motion planning with uncertainty: a landmark approach. Artif. Intell. **76**(1&2), 287–317 (1995)
4. Barto, A.G., Bradtke, S.J., Singh, S.P.: Learning to act using real-time dynamic programming. Artif. Intell. **72**(1), 81–138 (1995)
5. Behnke, S.: Local multiresolution path planning. In: Proceedings of 7th RoboCup International Symposium, pp. 332–343 (2004)
6. Bertsekas, D.: Dynamic Programming and Optimal Control, vols. I and II. Athena Scientific, Nashua, NH (2007)
7. Bellingham, J., Richards, A., How, J.P.: Receding horizon control of autonomous aerial vehicles. In: Proceedings of the American Controls Conference, pp. 3741–3746 (2002)
8. Belta, C., Bicci, A., Egerstedt, M., Frazzoli, E., Klavins, E., Pappas, G.: Symbolic control and planning of robotic motion. Proc. IEEE Robot. Autom. Mag. **14**, 61–70 (2007)
9. Belta, C., Isler, V., Pappas, G.: Discrete abstractions for robot motion planning and control in polygonal environments. IEEE Trans. Robot. **21**(5), 864–874 (2005)
10. Betts, J.T.: Survey of numerical methods for trajectory optimization. J. Guid. Control Dyn. **21**, 193–207 (1998)
11. Bohlin, R., Kavraki, L.E.: Path planning using lazy PRM. IEEE Int. Conf. Robot. Autom. **1**, 521–528 (2000)
12. Borenstein, J., Koren, Y.: Real-time obstacle avoidance for fast mobile robots. IEEE Trans. Syst. Man Cybern. **19**(5), 1179–1187 (1989)
13. Borenstein, J., Koren, Y.: The vector field histogram—fast obstacle avoidance for mobile robots. IEEE J. Robot. Autom. **7**, 278–288 (1991)
14. Boyle, D.P., Charnitoff, G.E.: Autonomous maneuver tracking for self-piloted vehicles. J. Guid. Control Dyn. **22**(1), 58–67 (1999)
15. Brock, O., Khatib, O.: High-speed navigation using the global dynamic window approach. In: IEEE International Conference on Robotics and Automation, pp. 341–346 (1999)
16. Brockett, R.W.: Formal languages for motion description and map making. Proc. Symp. Appl. Math. **41**, 181–193 (1990)

17. Call, B., Beard, R., Taylor, C., Barber, B.: Obstacle avoidance for unmanned air vehicles using image feature tracking. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, Keystone, CO (2006)
18. Canny, J.F.: The Complexity of Robot Motion Planning. MIT, Cambridge (1988)
19. Carlyle, W.M., Wood, R.K.: Lagrangian relaxation and enumeration for solving constrained shortest-path problems. In: Proceedings of the 38th Annual ORSNZ Conference (2007)
20. Caselli, S., Reggiani, M., Rocchi, R.: Heuristic methods for randomized path planning in potential fields. In: Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation, pp. 426–431 (2001)
21. Choset, H., Burdick, J.: Sensor-based exploration: the hierarchical generalized Voronoi graph. Int. J. Rob. Res. **19**(2), 96–125 (2000)
22. Choset, H., Walker, S., Eiamsa-Ard, K., Burdick, J.: Sensor-based exploration: incremental construction of the hierarchical generalized Voronoi graph. Int. J. Rob. Res. **19**(2), 126–148 (2000)
23. Conner, D.C., Rizzi, A.A., Choset, H.: Composition of local potential functions for global robot control and navigation. In: Proceedings of 2003 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems, pp. 3546–3551 (2003)
24. Conner, D.C., Choset, H., Rizzi, A.A.: Integrated planning and control for convex-bodied nonholonomic systems using local feedback control policies. In: Robotics: Science and Systems II, Philadelphia, PA (2006)
25. Connolly, C.I.:Harmonic functions and collision probabilities. Int. J. Rob. Res. **16**(4), 497 (1997)
26. Connolly, C.I., Souccar, K., Grupen, R.A.: A Hamiltonian framework for kinodynamic planning and control. Proc. IEEE Conf. Robot. Autom. **3**, 2746–2752 (1995)
27. Connolly, C.I., Burns, J.B., Weiss, R.: Path planning using Laplace's equation. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 2102–2106 (1990)
28. Dadkhah, N., Kong, Z., Korukanti, V., Mettler, B.: Experimental demonstration of an online trajectory optimization scheme using approximate spatial value functions. In: Proceedings of the Conference on Decision and Control, Shanghai, China (2009)
29. Dechter, R., Pearl, J.: Generalized best-first search strategies and the optimality of A*. J. ACM **32**, 505–536 (1985)
30. Dever, C., Mettler, B., Feron, E., Papovic, J., McConley, M.: Trajectory interpolation for parametrized maneuvering and flexible motion planning of autonomous vehicles. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, Providence, RI (2004)
31. Donald, B., Xavier, P.: Kinodynamic motion planning. J. ACM **40**, 1048–1066 (1993)
32. Donald, B.R., Xavier, P.G.: Provably good approximation algorithms for optimal kinodynamic planning for Cartesian robots and open chain manipulators. Algorithmica **14**, 958–963 (1995)
33. Ferguson, D., Likhachev, M., Stentz, A.: A Guide to heuristic-based path planning. In: Proceedings of ICAPS Workshop on Planning under Uncertainty for Autonomous Systems, AAAI (2005)
34. Fox, D., Burgard, W., Thrun, S.: The dynamic window approach to collision avoidance. IEEE Robot. Autom. Mag. **4**(1), 23–33 (1997)
35. Frazzoli, E., Dahleh, M.A., Feron, E.: A hybrid control architecture for aggressive maneuvering of autonomous helicopters. In: Proceedings of the Conference on Decision and Control, Phoenix, AZ (1999)
36. Frazzoli, E., Dahleh, M.A., Feron, E.: Real-time motion planning for agile autonomous vehicles. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, Denver, CO (2000)
37. Frazzoli, E., Dahleh, M.A., Feron, E.: Real-time motion planning for agile autonomous vehicles. J. Guid. Control Dyn. **25**(1), 116–129 (2002)
38. Frew, E.: Receding horizon control using random search for UAV navigation with passive, noncooperative sensing. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, San Francisco, CA (2005)
39. Fujimura, K., Tunii, T.L.: Motion Planning in Dynamic Environments. Springer, Secaucus (1992)
40. Gavrilets, V., Frazzoli, E., Mettler, B., Piedmonte, M., Feron, E.: Aggressive maneuvering of small autonomous helicopters: a human-centered approach. Int. J. Rob. Res. **20**(10), 795 (2001)
41. Geyer, M.S., Johnson, E.N.: 3D obstacle avoidance in adversarial environments for unmanned aerial vehicles. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, Keystone, CO (2006)

42. Guillemin, V., Pollack, A.: Differential Topology. Prentice-Hall, New York (1974)
43. Hamner, B., Singh, S., Scherer, S.: Learning obstacle avoidance parameters from operator behavior. In: Special Issue on Machine Learning Based Robotics in Unstructured Environments, vol. 23, pp. 1037–1058. Wiley InterScience (2006)
44. Howlett, J., Tsenkov, P., Whalley, M., Schulein, G., Takahashi, M.: Flight evaluation of a system for unmanned rotorcraft reactive navigation in uncertain urban environments. Presented at the 63rd Annual Forum of the American Helicopter Society, Virginia Beach, VA (2008)
45. Howlett, J.K., Schulein, G., Mansur, M.H.: A practical approach to obstacle field route planning for unmanned rotorcraft. In: American Helicopter Society 60th Annual Forum Proceedings, Baltimore, MD (2004)
46. Hristu-Varsakelis, D., Egerstedt, M., Krishnaprasad, P.S.: On the structural complexity of the motion description language MDLe. In: Proceedings of the 42nd IEEE Conference on Decision and Control, pp. 3360–3365 (2003)
47. Hui-Zhong Zhuang, D.S., Wu, T.: Real-time path planning for mobile robots. In: Proceedings of International Conference on Machine Learning and Cybernetics, vol. 1, pp. 526–531 (2005)
48. Hwang, Y.K., Ahuja, N.: Gross motion planning—a survey. ACM Comput. Surv. **24**, 219–291 (1992)
49. Hwangbo, M., Kuffner, J., Kanade, T.: Efficient two-phase 3D motion planning for small fixed-wing UAVs. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 1035–1041 (2007)
50. Ikonen, L.: Pixel queue algorithm for geodesic distance transforms. In: Discrete Geometry for Computer Imagery, pp. 228–239 (2005)
51. Ikonen, L., Toivanen, P.: Shortest routes on varying height surfaces using gray-level distance transforms. Image Vis. Comput. **23**, 133–140 (2005)
52. Jadbabaie, A.: Receding horizon control of nonlinear systems: a control Lyapunov function approach. PhD, California Institute of Technology (2000)
53. Junge, O., Marsden, J.E., Ober-blöbaum, S.: Discrete mechanics and optimal control. In: The 16th IFAC World Congress, Prague, Czech (2005)
54. John Canny, A.R.: An exact algorithm for kinodynamic planning in the plane. In: Proceedings of the Sixth Annual Symposium on Computational Geometry, pp. 271–280 (1990)
55. Judd, K.B., McLain, T.W.: Spline based path planning for unmanned air vehicles. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, Montreal, Canada (2000)
56. Kavraki, L., Nolountzakis, M., Latombe, J.: Analysis of probabilistic roadmaps for path planning. IEEE Trans. Robot. Autom. **14**(1), 166–171 (1998)
57. Kavraki, L., Svestka, P., Latombe, J., Overmars, M.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Robot. Autom. **12**, 566–580 (1996)
58. Keith, G., Tait, J., Richards, A.G.: Efficient path optimization with terrain avoidance. In: AIAA Navigation, Guidance and Control Conference, Hilton Head, SC (2007)
59. Khatib, O., Mampey, L.M.: Fonction Decision-Commande d'un Robot Manipulateur, Rep. 2/7156. DERA/CERT, Toulouse, France (1978)
60. Kim, S.H., Bhattacharya, R.: Multi-layer approach for motion planning in obstacle rich environments. In: AIAA Navigation, Guidance and Control Conference, Hilton Head, SC (2007)
61. Kobilarov, M., Desbrun, M., Marsden, J., Sukhatme, G.S.: A discrete geometric optimal control framework for systems with symmetries. In: Robotics: Science and Systems, vol. 3, pp. 1–8 (2007)
62. Kong, Z., Korukanti, V., Mettler B.: Mapping 3D guidance performance using approximate optimal cost-to-go function. In: AIAA Navigation, Guidance and Control Conference, Chicago, IL (2009)
63. Koren, Y., Borenstein, J.: Potential field methods and their inherent limitations for mobile robot navigation. In: Proceedings of the IEEE Conference on Robotics and Automation, vol. 2, pp. 1398–1404 (1991)
64. Tarabanis, K., Allen, P.K., Tsai, R.Y.: A survey of sensor planning in computer vision. IEEE Trans. Robot. Autom. **11**, 86–104 (1995)
65. Kuwata, Y., Schouwenaars, T., Richards, A., How J.: Robust constrained receding horizon control for trajectory planning. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, San Francisco, CA (2005)
66. Latombe, J.: Robot Motion Planning. Kluwer Academic, Boston (1991)
67. LaValle, S.M.: Rapidly-exploring random trees: a new tool for path planning. Tech. Rep (TR 98-11), Computer Science Dept, Iowa State University (1998)

68. LaValle, S.M., Konkimalla, P.: Algorithms for computing numerical optimal feedback motion strategies. Int. J. Rob. Res. **20**, 729–752 (2001)
69. LaValle, S.M.: From dynamic programming to RRTs: algorithmic design of feasible trajectories. In: Control Problems in Robotics, pp. 19–37 (2002)
70. LaValle, S.: Planning Algorithms. Cambridge University Press, Cambridge (2006)
71. LaValle, S.M., Kuffner, J.J.: Randomized kinodynamic planning. In: Proceedings of IEEE International Conference on Robotics and Automation, vol. 20, no. 5, p. 378 (1999)
72. Marigo, A., Bicchi, A.: Steering driftless nonholonomic systems by control Quanta. In: Proceedings of IEEE International Conference on Decision and Control, pp. 4164–4169 (1998)
73. Manikonda, V., Krishnaprasad, P.S., Hendler, J.: Languages, behaviors, hybrid architectures and motion control. In: Mathematical Control Theory, pp. 199–226 (1998)
74. Milam, M.B., Franz, R, Murray R.M.: Real-time constrained trajectory generation applied to a flight control experiment. In: Proceedings of the IFAC World Congress (2002)
75. Marsden, J.E., West, M.: Discrete mechanics and variational integrators. Acta Numer. **10**, 357–514 (2001)
76. Mazer, E., Ahuactzin, J.M.: The Ariadne's clew algorithm. J. Artif. Intell. Res. **9**, 1–32 (1998)
77. Menon, P.K., Sweriduk, G.D., Bowers, A.H.: Study of near-optimal endurance-maximizing periodic cruise trajectories. J. Aircr. **44**(2), 393–398 (2005)
78. Mettler, B., Valenti, M, Schouwenaars, T, Frazzoli, E., Feron, E.: Rotorcraft motion planning for Agile maneuvering using a maneuver automaton. In: 58th Forum of the American Helicopter Society, Montreal, Canada (2002)
79. Mettler, B., Bachelder, E.: Combining on-and offline optimization techniques for efficient autonomous vehicle's trajectory planning. In: AIAA Guidance, Navigation, and Control Conference, San Francisco, CA (2005)
80. Mettler, B., Toupet, O.: Receding horizon trajectory planning with an environment based cost-to-go function. In: Joint ECC-CDC Conference, Seville, Spain (2005)
81. Mettler, B., Kong, Z.: Receding horizon trajectory optimization with a finite-state value function approximation. In: American Control Conference (ACC), Seattle, WA (2008)
82. Mitchell, J., Papadimitriou, C.: The weighted region problem: finding shortest paths through a weighted planar subdivision. J. ACM **38**, 18–73 (1991)
83. Kazhdan, M., Bolitho, M., Hoppe H.: Poisson surface reconstruction. In: Proceedings of Eurographics Symposium on Geometry, pp. 61–70 (2006)
84. Ogren, P., Leonard, N.E.: A convergent dynamic window approach to obstacle avoidance. IEEE Trans. Robot. **21**, 188–195 (2005)
85. Piedmonte, M., Feron, E.: Aggressive maneuvering of autonomous aerial vehicles: a human-centered approach. In: Proceedings of the International Symposium on Robotics Research, vol. 9, pp. 413–420 (1999)
86. Prazenica, R.J., Kurdila, A.J., Sharpley, R.C., Evers, J.: Multiresolution and adaptive path planning for maneuver of Micro-Air-Vehicles in urban environments. In: AIAA Navigation, Guidance and Control Conference, San Francisco, CA (2005)
87. Redding, J., Amin, J.N., Boskovic, J.D., Kang, Y., Hedrick, K., Howlett, J., Poll, S.: A real-time obstacle detection and reactive path planning system for autonomous small-scale helicopters. In: AIAA Navigation, Guidance and Control Conference, Hilton Head, SC (2007)
88. Reif, J., Sun, Z.: An efficient approximation algorithm for weighted region shortest path problem. In: Proceedings of the 4th Workshop on Algorithmic Foundations of Robotics, p. 191 (2000)
89. Reif, J.H., Wang, H.: Nonuniform discretization for kinodynamic motion planning and its applications. SIAM J. Comput. **30**, 161–190 (2000)
90. Rhinehart, M.: Monte Carlo testing of 2- and 3-dimensional route planners for autonomous UAV navigation in urban environments. PhD thesis, University of Minnesota (2008)
91. Richards, A., How, J.: Aircraft trajectory planning with collision avoidance using mixed integer linear programming. In: American Control Conference, Anchorage, AK (2002)
92. Rimon, E., Koditschek, D.E.: Exact robot navigation using cost functions: the case of distinct spherical boundaries in Eˆn. Proc. IEEE Int. Conf. Robot. Autom. **3**, 1791–1796 (1988)
93. Akishita, S., Kawamura, S., Hayashi, K.: Laplace potential for moving obstacle avoidance and approach of a mobile robot. In: Proceedings of the Japan–USA Symposium on Flexible Automation (a Pacific Rim Conference), pp. 139–142 (1990)
94. Scherer, S., Singh, S., Chamberlain, L.J., Saripalli, S.: Flying fast and low among obstacles. In: Proceedings International Conference on Robotics and Automation, pp. 2023–2029 (2007)

95. Schouwenaars, T., Mettler, B., Feron, E., How, J.: Robust motion planning using a maneuver automaton with built-in uncertainties. Proc. Am. Control Conf. **3**, 2211–2216 (2003)
96. Schouwenaars, T., Mettler, B., Feron, E., How, J.: Hybrid model for trajectory planning of agile autonomous vehicles. J. Aero. Comput. Inform. Commun. **1**(12), 466–478 (2004)
97. Schouwenaars, T., Moor, B.D., Feron, E., How, J.: Mixed integer programming for multi-vehicle path planning. In: European Control Conference (2001)
98. Schwartz, J.T., Yap, C.K.: Advances in Robotics: Algorithmic and Geometric Aspects of Robotics, vol. 1. Lawrence Erlbaum, Philadelphia (1986)
99. Shanmugavel, M., Tsourdos, A., Zbikowski, R., White, B.A.: 3d path planning for multiple UAVs using Pythagorean Hodograph curves. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, Hilton Head, South Carolina (2007)
100. Shim, D., Sastry, S.: A situation-aware flight control system design using real-time model predictive control for unmanned autonomous helicopters. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, Keystone, CO (2006)
101. Shkel, A., Lumelsky, V.: The jogger's problem: accounting for body dynamics in real-time motion planning. In: Intelligent Robots and Systems, vol. 95, pp. 441–447 (1995)
102. Sinopoli, B., Micheli, M., Donato, G., Koo, T.J.: Vision based navigation for an unmanned aerial vehicle. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 1757–1765 (2001)
103. Singh, L., Fuller, J.: Trajectory generation for a UAV in urban terrain, using non-linear MPC. In: Proceedings of American Control Conference, pp. 2301–2308 (2001)
104. Slater, G.L.: Guidance on maneuvering flight paths for rotary wing aircraft. In: AIAA Guidance, Navigation and Control Conference, vol. 1, pp. 793–803. Monterey, CA (1987)
105. Soucy, M., Payeur, P.: Robot path planning with multiresolution probabilistic representations: a comparative study. In: Proceedings of the Canadian Conference on Electrical and Computer Engineering, Niagara Falls, Canada (2004)
106. Spenko, K., Overholt, J., Iagnemma, K.: High speed hazard avoidance for unmanned ground vehicles in emergency situations. In: 25th Army Science Conference, Orlando, FL (2006)
107. Stopp, A., Riethmuller, T.: Fast reactive path planning by 2D and 3D multi-layer spatial gridsfor mobile robot navigation. In: Proceedings of the 1995 IEEE International Symposium on Intelligent Control, pp. 545–550 (1995)
108. Suzuki, S., Komatsu, Y., Yonezawa, S., Masui, K., Tomita, H.: Online four-dimensional flight trajectory search and its flight testing. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, San Francisco, CA (2005)
109. Švestka, P., Overmars, M.H.: Coordinated motion planning for multiple car-like robots using probabilistic roadmaps. In: Proceedings of the IEEE Conference on Robotics and Automation, Nagoya Japan (1995)
110. Takahashi, M.D., Schulein, G., Whalley, M.:. Flight Control Law Design and Development for an Autonomous Rotorcraft. American Helicopter Society, Virginia Beach, VA (2008)
111. Toupet, O., Mettler, B.: Design and flight test evaluation of guidance system for autonomous rotorcraft. In: Proceedings of the AIAA Guidance, Navigation and Control Conference, Keystone, CO (2006)
112. Toussaint, G.J., Basar, T., Bullo, F.: Motion planning for nonlinear underactuated vehicles using H-infinity techniques. In: Proceedings of the 2001 American Control Conference, vol. 5, pp. 4097–4102 (2001)
113. Tsenkov, P., Howlett, J.K., Whalley, M., Schulein, G., Takahashi, M., Rhinehart, M.H., Mettler, B.: A system for 3d autonomous rotorcraft navigation in urban environments. In: AIAA Guidance, Navigation and Control Conference and Exhibit,Honolulu, HI (2008)
114. Thrun, S.: Robotic mapping: a survey. In: Exploring Artificial Intelligence in the New Millennium. Morgan Kaufmann, San Francisco (2002)
115. Thrun, S., Diel, M., Hahnel, D.: Scan alignment and 3-D surface modeling with a helicopter platform. In: International Conference on Field and Service Robotics, Japan (2003)
116. Thrun, S., Montemerlo, M.: The graph slam algorithm with applications to large-scale mapping of urban structures. Int. J. Rob. Res. **25**(5–6), 403 (2006)
117. Vandapel, N., Kuffner, J., Amidi, O.: Planning 3-D path networks in unstructured environments. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 4624–4629 (2005)
118. Wei, S., Zefran, M.: Smooth path planning and control for mobile robots. In: Proceedings of IEEE International Conference on Networking, Sensing and Control, pp. 894–899 (2005)

119. Whalley, M., Freed, M., Harris, R., Takahashi, M., Schulein, G., Howlett, J.K.: Design, integration, and flight test results for an autonomous surveillance helicopter. In: Proceedings of the AHS International Specialists Meeting on Unmanned Rotorcraft, Candler, AZ (2005)
120. Yakimenko, O.A.: Direct method for rapid prototyping of near-optimal aircraft trajectories. J. Guid. Control Dyn. **23**(5), 865 (2000)
121. Yang, G., Kapila, V.: Optimal path planning for unmanned air vehicles with kinematic and tactical constraints. In: IEEE Conference on Decision and Control, Las Vegas, NV (2002)
122. Yang, H.I., Zhao, Y.J.: Trajectory planning for autonomous aerospace vehicles amid known obstacles and conflicts. J. Guid. Control Dyn. **27**(6), 997–1008 (2004)
123. Zavlangas, P.G., Tzafestas, P.S.G., Althoefer, D.K.: Fuzzy obstacle avoidance and navigation for omnidirectional mobile robots. In: European Symposium on Intelligent Techniques, Aachen, Germany (2000)
124. Zelek, J.S.: Dynamic path planning. In: IEEE International Conference on Systems, Man and Cybernetics, Vancouver, Canada (1995)
125. Zengin, U., Dogan, A.: Probabilistic trajectory planning for UAVs in dynamic environments. In: AIAA 3rd "Unmanned Unlimited" Technical Conference, Workshop and Exhibit, Chicago, IL (2004)