JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 27, No. 6, November–December 2004

Trajectory Planning for Autonomous Aerospace Vehicles amid Known Obstacles and Conflicts

Hong Iris Yang* and Yiyuan J. Zhao†
University of Minnesota, Minneapolis, Minnesota 55455

A discrete search strategy is presented for potential real-time generations of four-dimensional trajectories for a single autonomous aerospace vehicle amid known obstacles and conflicts. A model of autonomous operation is first developed. The problem of and requirements on real-time trajectory generations for autonomous aerospace vehicles are discussed. After an overview of various potential solution frameworks, a discrete search strategy is developed. In this strategy, a four-dimensional search space is defined and discretized. Potential obstacles and conflicts are represented by several basic geometric shapes and their combinations. Mathematical conditions are developed for a trajectory segment to be outside of an obstacle or conflict. Then, the A^* search technique is used to obtain trajectory solutions, in which successor points are selected that avoid obstacles and conflicts and that satisfy dynamic motion constraints of the vehicle. A linear combination of flight distance and flight time is optimized in the trajectory generation process. A heuristic function that approximates this performance index is developed for the A^* search procedure. Examples are provided that illustrate the application of the proposed method.

I. Introduction

AUTONOMOUS aerospace vehicles have become increasingly attractive for missions where human presence is dangerous or difficult.[1] However, an unpiloted vehicle does not automatically become an autonomous vehicle unless it has some intelligence. Specifically, an autonomous vehicle should be capable of accomplishing missions on its own. In particular, it should have the ability to plan its motion trajectories in the lack of timely communications with base control stations.

Real-time trajectory planning is a key enabling technology for autonomous operations. Feasible trajectories should enable an autonomous vehicle to reach a desired final destination within a specified time and to avoid obstacles and potential conflicts with other vehicles. These trajectories should respect limitations of onboard control system capabilities and vehicle performances. In addition, it is desirable that these trajectories can optimize some performance index.

Developing practical methods for real-time trajectory generation can be quite challenging. Real-time trajectory generations onboard autonomous vehicles must be sufficiently fast and always reliable. In addition, practical methods for real-time trajectory generations must be able to make do with limited computer time and storage onboard an autonomous vehicle and limited, even poor, information available from onboard sensing equipments and communication devices. These challenges are further compounded by the fact that autonomous aerospace vehicles may often be expected to operate in dynamic, varying, and sometimes unknown environments and to engage in aggressive maneuvers near their performance boundaries.

Efforts have been made to develop trajectory generation methods for various types of autonomous vehicle operations. Krozel[2] studied two-dimensional path planning for a vehicle flying at constant altitude in mountainous terrain. The vehicle is considered as a point mass, and search graphs are constructed to model paths in free space. In Ref. 3, Chandler et al. studied two-dimensional path planning and coordination of multiple unmanned air vehicles. In particular, a two-stage solution concept was proposed in which a coarse polygonal path would first be found and would then be refined to obtain a flyable trajectory. In Ref. 4, McLain and Beard studied the coordination of multiple unmanned aerospace vehicle (UAV) rendezvous at a predetermined target location in the horizontal plane. Further studies of these approaches were made in Ref. 5. In Ref. 6, Frazzoli et al. presented a randomized motion-planning algorithm by employing obstacle-free guidance systems as local planners in a probabilistic roadmap framework for robot motion planning.[7] A family of trim autonomous helicopter trajectories in level flight was used to construct optimal maneuvers. In Ref. 8, Faiz et al. proposed a trajectory-planning scheme for differentially flat dynamic systems. For controls of autonomous vehicles, Mettler et al. & Kanade developed a parameterized model for a small-scale unmanned helicopter using a frequency-domain identification technique,[9] and in Refs. 10 and 11 feedback control design techniques were examined. Despite all of these results, significant research efforts are still needed to advance the state of the art of trajectory planning for autonomous aerospace vehicles.

In Ref. 12, Sasiadek and Duleba gave a clear review of research efforts in robotics literature and presented an effective method of local trajectory planning for a generic autonomous vehicle. In this method, a motion path is first planned and trajectories as functions of time are then generated. In addition, the presence of navigational uncertainties is considered.

This paper presents a global discrete search strategy that can be potentially used for real-time onboard generations of four-dimensional trajectories for autonomous aerospace vehicles. Specifically, a general model of autonomous operation is first introduced. Available solution frameworks for trajectory generations are reviewed and compared, and a discrete search paradigm is proposed. In this paradigm, a four-dimensional search space is first defined and discretized. Potential obstacles are represented by several basic geometric shapes and/or their combinations. Mathematical conditions are developed that ensure a trajectory segment to be outside of an obstacle. Then, the A^* search scheme in artificial intelligence is used to determine trajectory solutions. In this search scheme, successor points are generated that both avoid all specified obstacles and stay within limits of dynamic vehicle motion constraints. A heuristic function is developed that approximates a linear combination of minimum distance-to-go and minimum time-to-go. Numerical examples are presented to illustrate the applications of the proposed discrete search strategy.

Received 10 July 2002; accepted for publication 1 January 2004.

*Ph.D. Candidate, Aerospace Engineering and Mechanics. Student Member AIAA.

†Associate Professor, Aerospace Engineering and Mechanics. Associate Fellow AIAA.

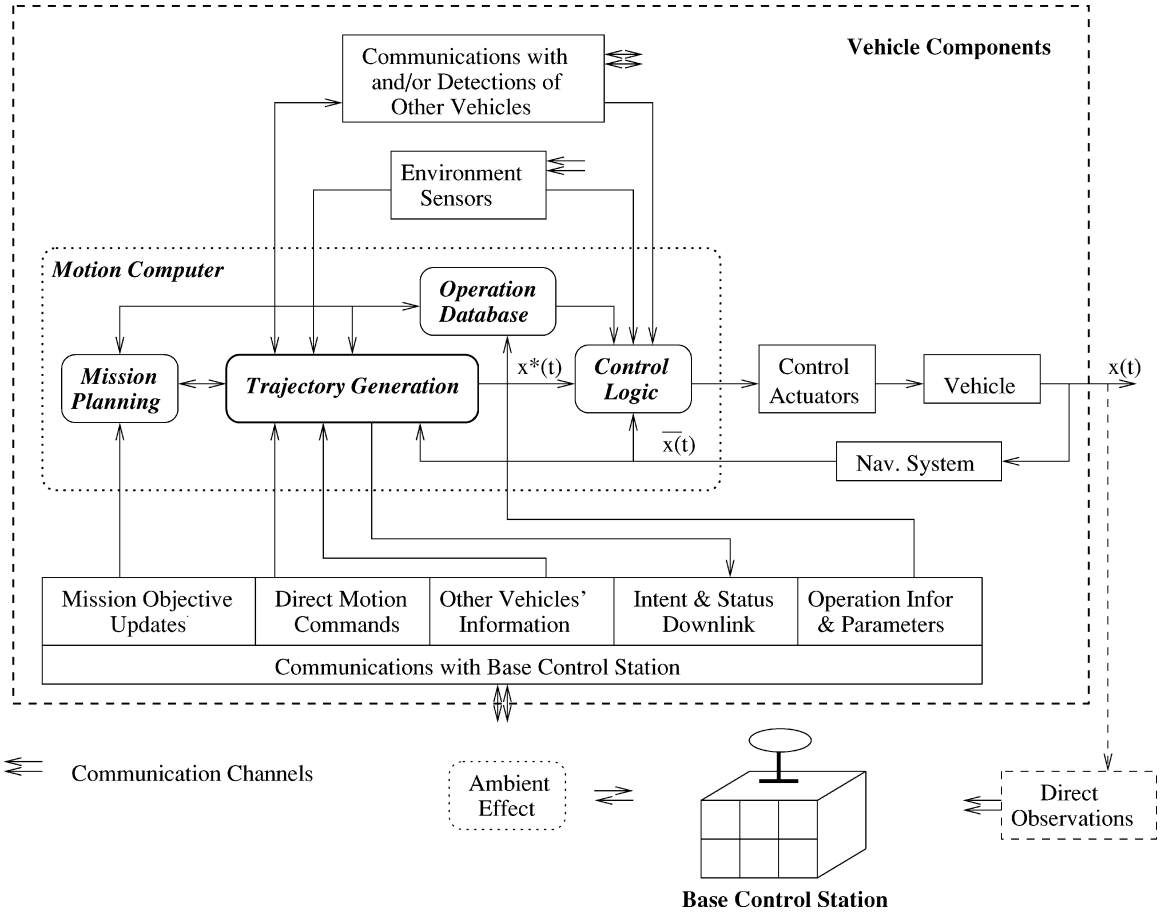

Fig. 1 Autonomous operation model.

II. General Model of Autonomous Operation

All human-made vehicles can eventually be made autonomous to a certain degree through the use of automation and intelligence. Whereas different autonomous vehicles may operate in different environments (air, space, water, or land) and have different requirements on onboard trajectory generations, it is helpful to develop a common model of autonomous operations.

Figure 1 attempts to provide an integrated description of various physical as well as software components in the operation of a generic autonomous vehicle and the environment for trajectory generations. For different types of autonomous vehicles, the various components in Fig. 1 operate under different principles and have different characteristics. Still, they contain some basic functionalities that are the same for different autonomous operations. In Fig. 1, vehicle refers to the basic vehicle construction, excluding add-on hardware or software systems. Communication devices and datalinks are needed to send and/or receive information to/from the base control station and other vehicles. Here, communication devices represent channels that are used when needed, whereas datalinks transmit signals automatically at a fixed rate. Environment sensors are used to detect obstacles in the operational environment and to determine ambient conditions. Navigation systems contain sensors and measurement equipment that are used to determine motion states of the autonomous vehicle itself. Operation database contains information on control system and vehicle performance limits, preknowledge of the operational environments, possible in-flight parameter tuning schedules, and any intelligence for autonomy.

In particular, a three-level hierarchy is proposed for the functions of mission planning, trajectory generation, and control, all housed in the onboard computer system. This modular division is by no means unique, but is convenient for system design and is flexible for future growth. The main function of the control logic is to maintain vehicle stability and to follow the motion trajectories produced by the trajectory generation function. In comparison, the trajectory generation function calculates future motion trajectories that achieve specified mission objectives, satisfy vehicle performance constraints, stay away from obstacles in the environment and potential conflicts with other vehicles, and respect other practical constraints. Finally, mission planning determines high-level characteristics of maneuvers to achieve mission objectives while safeguarding the vehicle integrity. It translates mission objectives into mathematical formulations for trajectory generations. It can also determine when to replan motion trajectories.

III. Problem Statement

In trajectory generations, it is convenient to distinguish among obstacles, potential conflicts, and threats. Obstacles are stationary or slowly moving objects and can be avoided by flying barely outside of them. Potential conflicts refer to collisions or near misses between the own vehicle and other moving objects that are not intentionally threatening. In comparison, threats, either stationary or moving, intend to harm the vehicle. Predictable threats may be avoided either by flying as far as possible or at a safe distance from them. If a safe distance can be defined around a threat, it can be treated similarly as obstacles or potential conflicts in the trajectory generation process. In this case, obstacles, potential conflicts, and threats become motion constraints to trajectory generations. On the other hand, a threat that is not fully predictable or may move intelligently needs to be treated as an adversary threat in trajectory generations.

A key characterization of motion constraints for the purpose of trajectory generation is when and how well their presences are known to the onboard system. If the presences of all motion constraints are known well in advance for trajectory generations, methods with well-understood convergence properties can be developed. In comparison, motion constraints whose presences can only be

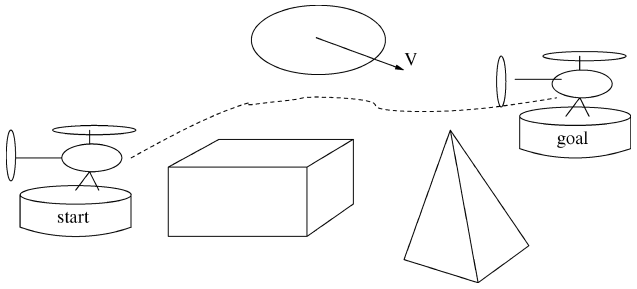

Fig. 2 Basic problem of UAV trajectory planning.

detected when the vehicle is close to them are called pop-up constraints. It is in general more difficult to develop solution methods with well-understood convergence properties for trajectory generations containing pop-up constraints.

Accordingly, the operational environment for real-time onboard trajectory generations may be classified into the following types, ranging from the simplest to the most challenging: 1) The environment contains only static obstacles, and their locations and properties are well known in advance. 2) The environment may contain obstacles, conflicts, and threats but no adversary threats. Their locations and properties are well known in advance. 3) The environment may contain obstacles, conflicts, and threats but no adversary threats. However, their locations and properties are not always accurately known. 4) In addition to those specified in type 3, the environment may contain pop-up constraints but no adversary threats. 5) In addition to those specified in type 4, the environment may contain adversary threats.

Trajectory generation for a type 2 environment is fundamental to autonomous aerospace vehicle operation. Solution methods for type 2 are automatically applicable to type 1 environments and can lay the foundation to the development of methods for other more complicated types of environments. In particular, these methods can be used to provide benchmarks for trajectory generations in other types of environments.

This paper seeks to address two problems: The first is to develop a trajectory generation framework that promises to be applicable to all five types of environments described earlier and to possess properties listed in the next section. The second is to develop a fast and reliable solution strategy for type 2 environments. Specifically, this strategy would be able to plan safe and flyable four-dimensional trajectories (position and time) for an autonomous aerospace vehicle starting from a given initial state and reaching a specified final state while optimizing a certain performance index. In a type 2 environment, all obstacles and potential conflicts are known a priori and can be time varying (Fig. 2). The following performance index is optimized:

$$\min I = K_d d_f + K_t t_f \tag{1}$$

where d_f is the path length of the trajectory, t_f is the flight time of the trajectory, and $K_d \geq 0$ and $K_t \geq 0$ are appropriate weighting factors.

IV. Proposed Solution Framework for Trajectory Generation

Planned flight trajectories for a single autonomous vehicle must be safe and flyable and should desirably optimize a certain performance index. Safety means that these trajectories must avoid all obstacles, potential conflicts, and threats. Flyability means that they must be physically possible for the vehicle to follow with acceptable margins of error and within limitations of onboard control system capabilities and performance limitations. In this paper, feasible trajectories are both safe and flyable.

An acceptable solution method for real-time onboard trajectory generations must be able to produce feasible solutions sufficiently fast with limited computer resources and to make do with limited quantity and quality of vehicle and environment information from onboard communication devices and sensors. In addition, a desirable solution method should be easily expandable to allow for multiple vehicle coordinations and to handle cases with adversary threats. In general, a solution method may be evaluated according to some or all of the following criteria:

The first criterion is completeness. A solution method is complete if it can always find a feasible trajectory solution when there is one. In other words, it may be concluded that there is no feasible solution if a complete method does not find one.

The second is time and space complexity. Time complexity measures the amount of computational time required to produce a trajectory solution, whereas space complexity measures the computer memory storage needs.

The third criterion is robustness. A solution method is considered to be robust if it can produce reliable solutions when measurements of vehicle states and the environment contain errors.

The fourth is optimality. Optimality measures the degree with which trajectory solutions produced by a method optimize a given performance index.

The last criterion is flexibility. A solution method is said to be flexible if it can be used to solve a wide range of trajectory generation problems with small modifications.

We first define a general solution framework for onboard trajectory generations that can potentially satisfy most or all of the five criteria. A systematic overview of available solution frameworks is highly desirable in this endeavor.

Many theoretical frameworks and their variations may be used for trajectory generations. These frameworks include optimal control, parameter optimization, dynamic programming, potential field method, artificial intelligent methods, graphic theories, geometric methods, generalizations of robot motion planning methods,[7,12–14] etc. Although various frameworks differ significantly in form, they can be roughly divided into two groups: continuous optimization and discrete search.

In a continuous optimization framework, solution variables take on continuous values and, thus, can admit an infinite number of possible solutions. Presence of uncertainties may be treated in a stochastic framework or as worst-case scenarios in a deterministic framework. The presence of adversary objects can be considered in a game-theoretic approach. A main advantage of the continuous optimization approach is that it can produce smooth and flyable optimal trajectories. On the other hand, the continuous optimization approach has some significant disadvantages. Numerical solution methods for a continuous optimization problem typically rely on good initial guesses for convergence. There can be no a priori guarantee on their convergences. In addition, the more constraints, the more difficult it is in general to find a solution. Furthermore, continuous game problems can be very difficult to solve. These disadvantages make the direct or exclusive use of a continuous optimization approach difficult for real-time trajectory generations.

For example, the optimal control theory is often used for aerospace system trajectory generations. It explicitly allows for the optimization of a performance index and can incorporate vehicle and actuators dynamics as part of dynamics equations. However, it is inherently difficult to handle various kinds of constraints, uncertainties, and/or adversaries using the optimal control theory. In addition, all numerical solution methods can only yield locally optimal solutions, and their convergence heavily depends on the proper selection of initial guesses.

In comparison, discrete search schemes determine an optimal solution among a finite, albeit large, number of choices. They can have systematic starting procedures and theoretical guarantees on their convergences to globally optimal solutions. In general, these methods are inherently capable of handling constraint; uncertainties, for example, stochastic dynamic programming; and presences of adversary objects, for example, minimax formulations. Actually, the larger the number of constraints, the easier it is in general to find solutions. On the other hand, the use of discrete search schemes for autonomous vehicle trajectory generations requires a proper discretization of the search space, and solutions obtained within available computational time and computer memory may not be directly flyable or smooth. Furthermore, an applicable optimization

performance index may need to satisfy some requirements such as being additive.

Clearly, a two-step procedure that properly combines these two solution frameworks would work well for real-time trajectory generations of autonomous aerospace vehicles. In such a procedure, a search space for potential trajectory maneuvers can be first defined and discretized. An appropriate discrete search scheme may then be used to yield a coarse optimal trajectory solution. Next, a continuous optimization method can be used to refine the coarse trajectory solution. In addition to some apparent advantages, this procedure can be easily expanded to multivehicle trajectory generations through coordinations over specified waypoints at specified times.

A discrete search strategy is presented as the first part of this two-step procedure. Key elements of this strategy include proper representations of obstacles and conflicts, discretization of the search space, and application of a discrete search scheme. It generates discrete globally optimal trajectories within a specified search space.

V. Elements of a Discrete Search Strategy

A basic coordinate system, x, y, and h, is used throughout the paper, where the x axis points to the east, the y axis points to the north, and the h axis points upward. In general, the transformation between two coordinate systems is defined through three orientation angles, ϕ, θ, and ψ, where ψ is defined through a negative rotation about the third axis (the h axis), θ is defined through a positive rotation about the first axis, and ϕ is defined through a positive rotation about the second axis. For a vehicle, ψ represents the heading angle measured clockwise from the north (y axis).

A. Representations of Obstacles and Conflicts

Actual obstacles in an operational environment may come in different shapes. In this paper, four geometric shapes are used to represent basic obstacle elements: ellipsoid, cuboid, cylinder, and pyramid. Obstacles in practice can be adequately described by proper combinations of these basic shapes or approximated by the closure of a series of planes. In addition, location and dimension parameters of these obstacle elements can be defined as functions of time to represent potential conflicts caused by other moving objects.

A three-dimensional ellipsoid obstacle element (Fig. 3) is defined by nine parameters: coordinates of the center location, x_c, y_c, and h_c; three semi-axes, a, b, and c; and three orientation angles, ϕ, θ, and ψ, that relate the basic coordinate system to the principal system, X, Y, and H, of the ellipsoid,

$$\begin{bmatrix} X \\ Y \\ H \end{bmatrix} = T(\phi) \cdot T(\theta) \cdot T(\psi) \cdot \begin{bmatrix} x - x_c \\ y - y_c \\ h - h_c \end{bmatrix} \tag{2}$$

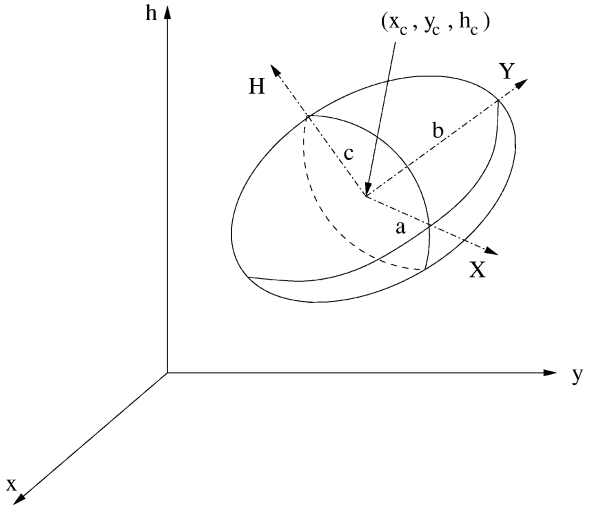

Fig. 3 Ellipsoid obstacle element.

where

$$T(\phi) = \begin{bmatrix} \cos\phi & 0 & -\sin\phi \\ 0 & 1 & 0 \\ \sin\phi & 0 & \cos\phi \end{bmatrix} \tag{3}$$

$$T(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix} \tag{4}$$

$$T(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

If a given point (x_p, y_p, h_p) after the transformation to the principal system, satisfies

$$X_p^2/a^2 + Y_p^2/b^2 + H_p^2/c^2 > 1 \tag{6}$$

it is outside of the ellipsoid obstacle.

A cuboid obstacle element (Fig. 4) is also defined by nine parameters: coordinates of the center location, x_c, y_c, and z_c; three dimensions, a, b, and c; and three orientation angles, ϕ, θ, and ψ. Transformation to the principal coordinate system is also achieved by Eq. (2). A given point (x_p, y_p, h_p) would be outside of the cuboid if, after the transformation, one of the following conditions is met:

$$|X_p| > a/2, \qquad |Y_p| > b/2, \qquad \text{or} \qquad |H_p| > c/2 \tag{7}$$

A cylinder obstacle element (Fig. 5) is fully described by eight parameters, which include coordinates of the center of the bottom

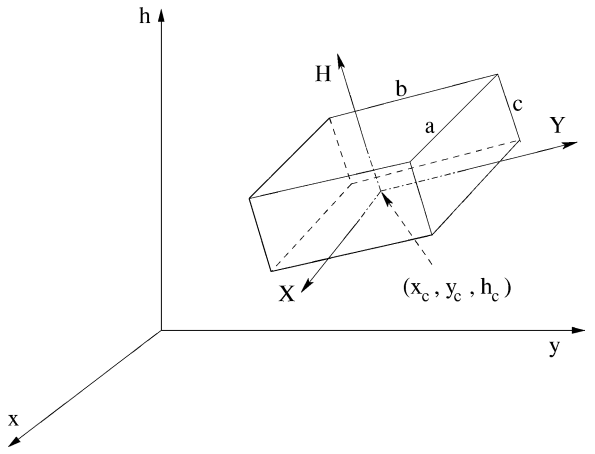

Fig. 4 Cuboid obstacle element.

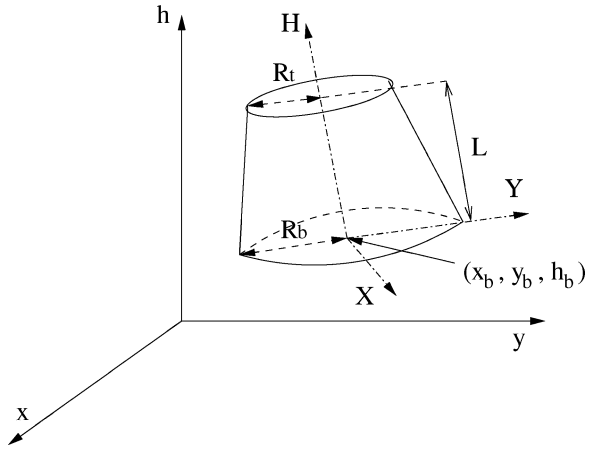

Fig. 5 Cylinder obstacle element.

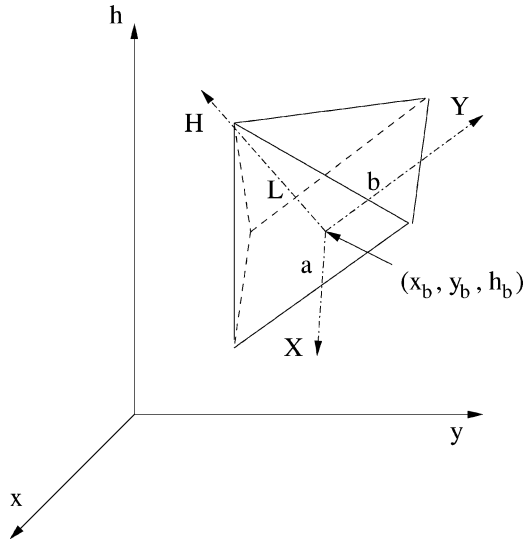

Fig. 6 Pyramid obstacle element.

surface, x_b, y_b, and h_b; radii of the top and bottom surfaces, R_t and R_b; the height L; and two orientation angles, ϕ and θ, where the axis of the cylinder is initially aligned up with the h axis. Transformation to a principal body axis is given by

$$\begin{bmatrix} X \\ Y \\ H \end{bmatrix} = T(\phi) \cdot T(\theta) \cdot \begin{bmatrix} x - x_b \\ y - y_b \\ h - h_b \end{bmatrix} \tag{8}$$

A given point (x_p, y_p, h_p) is outside of the cylinder if, after the transformation, one of the following conditions is met:

$$X_p^2 + Y_p^2 > [R_b - (R_b - R_t)(H_p/L)]^2, \qquad H_p < 0, \quad \text{or} \quad H_p > L \tag{9}$$

In general, an obstacle of arbitrary shape may be well approximated by the closure of a series of planes, where a direction can be defined for each plane that points inward to the obstacle. Plane representation is advantageous because planes are relatively easy to describe mathematically. For example, a square pyramid obstacle element (Fig. 6) can be defined by the closure of five planes: four on the sides and one base. This representation can be described completely with nine parameters that include coordinates of the base center, x_b, y_b, and z_b; dimensions of the base, a and b; the height L; and three orientation angles, ϕ, θ, and ψ. Transformation to the principal coordinate system can be obtained as in Eq. (2). In the principal coordinate system, unit inward-pointing vectors on the base and the side surfaces of the pyramid can be defined as

$$\begin{aligned} \boldsymbol{n}_0 &= [0, 0, 1] \\ \boldsymbol{n}_1 &= \left[0, -L/\sqrt{b^2 + L^2}, -b/\sqrt{b^2 + L^2}\right] \\ \boldsymbol{n}_2 &= \left[L/\sqrt{a^2 + L^2}, 0, -a/\sqrt{a^2 + L^2}\right] \\ \boldsymbol{n}_3 &= \left[0, L/\sqrt{b^2 + L^2}, -b/\sqrt{b^2 + L^2}\right] \\ \boldsymbol{n}_4 &= \left[-L/\sqrt{a^2 + L^2}, 0, -a/\sqrt{a^2 + L^2}\right] \end{aligned} \tag{10}$$

Let $[n_{ix}, n_{iy}, n_{ih}]$ be the X, Y, and H components of the unit inward-pointing vectors in the principal coordinate system in Eq. (10), and (x_i, y_i, h_i) is an arbitrary point on the ith plane, where $i = 0, 1, 2, 3, 4$. A given point, x_p, y_p, or h_p, is outside of the pyramid, after the transformation, if there exists at least one plane $i = 0$, 1, 2, 3, or 4, such that

$$n_{ix}(X_p - X_i) + n_{iy}(Y_p - Y_i) + n_{ih}(H_p - H_i) < 0 \tag{11}$$

where

$$\begin{bmatrix} X_p \\ Y_p \\ H_p \end{bmatrix} = T(\phi) \cdot T(\theta) \cdot T(\psi) \cdot \begin{bmatrix} x_p - x_b \\ y_p - y_b \\ h_p - h_b \end{bmatrix} \tag{12}$$

and (X_i, Y_i, H_i) can be similarly obtained.

B. Feasible Trajectory Line Segments

Whereas the preceding conditions can be used to check if a given point on a trajectory solution is outside of all obstacles, the line segment between any two trajectory points must also be outside of all obstacles. Consider a line segment formed by two endpoints (x_1, y_1, h_1) and (x_2, y_2, h_2) that themselves are outside of all obstacles. Let (x_p, y_p, h_p) be a generic point on the line segment. We have

$$\begin{bmatrix} x_p(\alpha) \\ y_p(\alpha) \\ h_p(\alpha) \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \\ h_1 \end{bmatrix} + \alpha \cdot \begin{bmatrix} \triangle x \\ \triangle y \\ \triangle h \end{bmatrix} \tag{13}$$

and after transformation to the principal coordinate system of a given obstacle element,

$$\begin{bmatrix} X_p(\alpha) \\ Y_p(\alpha) \\ H_p(\alpha) \end{bmatrix} = \begin{bmatrix} X_1 \\ Y_1 \\ H_1 \end{bmatrix} + \alpha \cdot \begin{bmatrix} \triangle X \\ \triangle Y \\ \triangle H \end{bmatrix} \tag{14}$$

In the preceding equation, $0 \le \alpha \le 1$,

$$\begin{aligned} &\triangle x = x_2 - x_1, \qquad \triangle y = y_2 - y_1, \qquad \triangle h = h_2 - h_1 \\ \text{and} \quad &\triangle X = X_2 - X_1, \qquad \triangle Y = Y_2 - Y_1, \qquad \triangle H = H_2 - H_1 \end{aligned} \tag{15}$$

We first consider the ellipsoid obstacle element. Define

$$\begin{aligned} D(\alpha) &= X_p^2(\alpha)/a^2 + Y_p^2(\alpha)/b^2 + H_p^2(\alpha)/c^2 - 1 \\ &= a_0\alpha^2 + a_1\alpha + a_2 \end{aligned} \tag{16}$$

where

$$\begin{aligned} a_0 &= \triangle X^2/a^2 + \triangle Y^2/b^2 + \triangle H^2/c^2 \\ a_1 &= 2\left[\left(X_1 \cdot \triangle X\right)/a^2 + \left(Y_1 \cdot \triangle Y\right)/b^2 + \left(H_1 \cdot \triangle H\right)/c^2\right] \\ a_2 &= X_1^2/a^2 + Y_1^2/b^2 + H_1^2/c^2 - 1 \end{aligned}$$

Clearly, if $D(\alpha) > 0$ for all $\alpha \in [0, 1]$, the specified line segment is outside of the ellipsoid. Otherwise, if $D(\alpha) \le 0$ for some $\alpha \in [0, 1]$, a part of the line segment is inside of the ellipsoid. Solving for α in $D(\alpha) = 0$ leads to

$$\alpha_{1,2} = \frac{-a_1 \pm \sqrt{a_1^2 - 4a_0a_2}}{2a_0} \tag{17}$$

If at least one solution satisfies $0 \le \alpha \le 1$, the line segment intersects the ellipsoid. Otherwise, the entire line segment is outside of the ellipsoid.

Next, let us consider a cylinder obstacle. If

$$H_1 > L, \qquad H_2 > L \tag{18}$$

or

$$H_1 < 0, \qquad H_2 < 0 \tag{19}$$

the line segment is clearly outside of the cylinder obstacle. If neither of these two conditions is true, we need to examine if the line segment intersects the sides of the cylinder obstacle. Define

$$\begin{aligned} D(\alpha) &= X_p^2(\alpha) + Y_p^2(\alpha) - \{R_b - (R_b - R_t)[H_p(\alpha)/L]\}^2 \\ &= X_p^2(\alpha) + Y_p^2(\alpha) - (\{R_b - [(R_b - R_t)/L]H_1\} \\ &\quad - [(R_b - R_t)/L]\triangle H\alpha)^2 \\ &= a_0\alpha^2 + a_1\alpha + a_2 \end{aligned} \tag{20}$$

where

$$a_0 = \triangle X^2 + \triangle Y^2 - \left[(R_b - R_t)^2 / L^2\right]\triangle H^2$$

$$a_1 = 2 \cdot \left\{X_1 \cdot \triangle X + Y_1 \cdot \triangle Y - \left[(R_b - R_t)^2 / L^2\right]H_1\triangle H + [R_b(R_b - R_t)/L]\triangle H\right\}$$

$$a_2 = X_1^2 + Y_1^2 - \left[(R_b - R_t)^2 / L^2\right]H_1^2 - R_b^2 + 2R_bH_1(R_b - R_t)/L$$

Clearly, if $D(\alpha) > 0$ for all $\alpha \in [0, 1]$, the specified line segment is outside of the cylinder. Otherwise, if $D(\alpha) \le 0$ for some $\alpha \in [0, 1]$, a part of the line segment is inside of the cylinder. Solving for α in $D(\alpha) = 0$ leads to

$$\alpha_{1,2} = \frac{-a_1 \pm \sqrt{a_1^2 - 4a_0a_2}}{2a_0} \tag{21}$$

If at least one solution satisfies $0 \le \alpha \le 1$, the line segment intersects the cylinder.

Finally, for obstacles defined by a finite number of surface planes, we need to examine the relationship of the line segment with every surface plane. Consider a generic surface plane described by

$$\boldsymbol{n}_x(x - x_s) + \boldsymbol{n}_y(y - y_s) + \boldsymbol{n}_h(h - h_s) = 0 \tag{22}$$

where $\boldsymbol{n}_x$, $\boldsymbol{n}_y$, and $\boldsymbol{n}_h$ are unit vectors perpendicular to the plane and point to the interior of the obstacle and x_s, y_s, and h_s are arbitrarily selected point on the surface plane. If

$$\boldsymbol{n}_x\triangle x + \boldsymbol{n}_y\triangle y + \boldsymbol{n}_h\triangle h = 0 \tag{23}$$

the line segment is parallel to the given plane, and whether part of the line segment is inside the obstacle is determined by its relations with other surface planes. Otherwise, the line segment or its extension intersects the plane and the intersection point can be determined from

$$\boldsymbol{n}_x[x_p(\alpha) - x_s] + \boldsymbol{n}_y[y_p(\alpha) - y_s] + \boldsymbol{n}_h[h_p(\alpha) - h_s] = 0 \tag{24}$$

or

$$\alpha^* = -\frac{\boldsymbol{n}_x(x_1 - x_s) + \boldsymbol{n}_y(y_1 - y_s) + \boldsymbol{n}_h(h_1 - h_s)}{\boldsymbol{n}_x\triangle x + \boldsymbol{n}_y\triangle y + \boldsymbol{n}_h\triangle h} \tag{25}$$

If $\alpha^* < 0$ or $\alpha^* > 1$ for all surface planes of an obstacle, the entire line segment is outside of the obstacle. If $0 \le \alpha^* \le 1$ for a certain surface plane, the line segment intersects the plane. Because a surface plane extends into infinity whereas the obstacle has a finite volume defined by the closure of all of its surface planes, further verifications are needed. If all of the intersection points with different surface planes are feasible points, that is, outside of the obstacle, the line segment is actually outside of the obstacle. Otherwise, part of the line segment is inside of the obstacle.

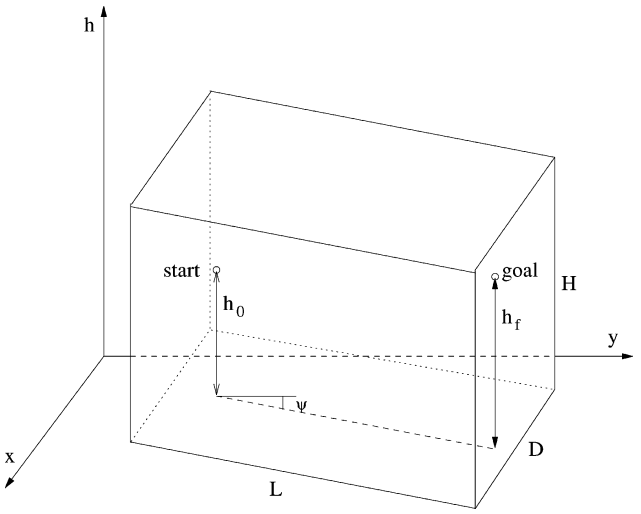

Fig. 7 Search space definition.

C. Definition of a Four-Dimensional Search Space

A search space defines the range in which an optimal trajectory solution is to be found. In this paper, the geometric search space is represented by a rectangular shape that contains both the starting location, x_0, y_0, and h_0, and the goal location, x_f, y_f, and h_f, of an autonomous vehicle. This rectangle can be fully defined by four parameters: the length L, the width D, the height H, and an orientation angle in the horizontal plane ψ (Fig. 7), which can be determined from

$$\psi = \cos^{-1}\left[\frac{y_f - y_0}{\sqrt{(x_f - x_0)^2 + (y_f - y_0)^2}}\right] \tag{26}$$

and the transformation to the principal coordinate system is given as

$$\begin{bmatrix} X \\ Y \\ H \end{bmatrix} = T(\psi) \cdot \begin{bmatrix} x - x_0 \\ y - y_0 \\ h \end{bmatrix} \tag{27}$$

The dimensional parameters need to satisfy

$$L \ge \sqrt{(x_f - x_0)^2 + (y_f - y_0)^2} \tag{28}$$

$$H \ge \max\{h_0, h_f\} \tag{29}$$

and the width D should be large enough to allow for sufficient lateral movement freedom. On the other hand, too large dimensions would increase the computational time in trajectory generations.

To employ a discrete search scheme, the search space needs to be discretized. Spatial discretizations can be made along the three dimensions by equal length divisions along each dimension with $\triangle L$, $\triangle D$, and $\triangle H$. Suppose the grid point G_{ijk} in the principal coordinate system refers to the point at ith interval in the X direction, jth interval in the Y direction, and kth interval in the H direction, where $i = 0, 1, \ldots, N_x = L/\triangle L$, $j = 0, 1, \ldots, N_y = D/\triangle D$, and $k = 0, 1, \ldots, N_h = H/\triangle H$. We have

$$\begin{bmatrix} X_{ijk} \\ Y_{ijk} \\ H_{ijk} \end{bmatrix} = \begin{bmatrix} i \cdot \triangle L \\ \left(j - \dfrac{D}{2 \cdot \triangle D}\right) \cdot \triangle D \\ k \cdot \triangle H \end{bmatrix} \tag{30}$$

and grid points in the original coordinate system are expressed as

$$\begin{bmatrix} x_{ijk} \\ y_{ijk} \\ h_{ijk} \end{bmatrix} = T^T(\psi) \cdot \begin{bmatrix} X_{ijk} \\ Y_{ijk} \\ H_{ijk} \end{bmatrix} + \begin{bmatrix} x_0 \\ y_0 \\ 0 \end{bmatrix} \tag{31}$$

In this discretization scheme, the ground is naturally defined by $h = 0$. However, it may not be possible to represent accurately both the starting altitude and the goal altitude by grid points, unless $h_0 = 0$. In this paper, the altitude discretization $\triangle H$ is selected such that the starting altitude is exactly on a grid point, whereas the goal point is approximated by the nearest vertical grid point.

Because of the dynamic nature of aerospace vehicle flight, the three-dimensional geometric search space needs to be augmented by including the time dimension. To discretize the time dimension, it is convenient to define a maximum range of flight time T, by which the actual UAV flight time is bounded. Consider a starting state (x_0, y_0, h_0, V_0) and goal state (x_g, y_g, h_g, V_g). If there is no obstacle and the vehicle changes speed linearly, the flight time would be given by

$$T^* = \frac{2\sqrt{(x_0 - x_g)^2 + (y_0 - y_g)^2 + (h_0 - h_g)^2}}{V_0 + V_g} \tag{32}$$

In case the final speed is not specified, one can set $V_g = V_0$ in Eq. (32) for estimating a reference flight time. On the other hand, there is a minimum feasible flight time $T_{\min}$ when obstacles are present in the flight environment. This minimum flight time may be estimated approximately or determined via a separate trajectory optimization problem. The maximum range of flight time T may be estimated as

$$T = \beta \max\{T^*, T_{\min}\} \tag{33}$$

where $\beta > 1$ is a scaling factor. Then, the time dimension can be discretized by

$$\triangle T = T/N_T \tag{34}$$

where N_T is the number of time grid points.

In general, vehicle flight speeds vary from grid point to grid point. For specified spatial movements, speed changes are directly related to flight times. In fact, the range of feasible speed variations at a given point defines the range of feasible arrival time variations or vice versa. As a result, the search space for trajectory generations in the current paper becomes four-dimensional,

$$G_{ijk,l} = (x_{ijk}, y_{ijk}, h_{ijk}, t_{ijk,l}; V_{ijk,l}) \tag{35}$$

where $l = 1, 2, \ldots, N_T$, $t_{ijk,l}$ are possible arrival times at the point, and $V_{ijk,l}$ are corresponding flight speeds at this point. The expression for the flight speed at a given successor point corresponding to a specified time of arrival is provided later in Eq. (52). The flight speed or flight time must be properly reflected in the optimization performance index or constraints for their inclusions in the search space to be meaningful.

VI. Application of the A^* Search Algorithm

Trajectory generation over a discretized search space may be solved by a number of different discrete search algorithms. The A^* search algorithm[15] has many good qualities and is used in this paper. The A^* search algorithm is basically a best-first search strategy. It begins from the starting grid point (initial vehicle state) and expands to other grid points, eventually reaching the goal state. In its core step, the algorithm generates a set of candidate successor points from the current point and determines which next point among these points to expand by optimizing the total cost of the next grid point.

The A^* search algorithm is optimally efficient in that no other optimal algorithm is guaranteed to expand fewer nodes than A^*. It is complete on locally finite graphs. It is an optimal algorithm. On the other hand, it could potentially take a considerable amount of computational time. Because it stores all generated nodes, it may run out of computer memory before it runs out of time. Research progress has been made that overcome the memory problem without sacrificing optimality or completeness.

A. Generation of Successor Points

To allow for sufficient freedom in trajectory generations, successor points are selected from neighboring grid points in all directions. Let $G^C_{ijk,l} = (x_{ijk}, y_{ijk}, h_{ijk}, t_{ijk,l}; V_{ijk,l})$ represent the current state; the set of neighboring points with depth N_s around $G^C_{ijk,l}$ are defined as

$$\Omega\left[G^C_{ijk,l}\right] = \{G_{i'j'k',l'}; -N_s \le (i' - i), (j' - j), (k' - k) \le N_s\} \tag{36}$$

where the depth N_s is an integer parameter that defines the range of the $G^C_{ijk,l}$ neighborhood. $\Omega[G^C_{ijk,l}]$ gives $(2N_s + 1)^3 - 1$ neighboring points, from which successor points are selected. For example, the set of neighboring points contains 26 points with depth one ($N_s = 1$) and 342 points with depth three ($N_s = 3$).

B. Dynamic Constraints

From the set of neighboring points around the current point, successor points are selected such that they are outside of all obstacles and potential conflicts, line segments from the current node to successor points are outside of all obstacles and conflicts, and they satisfy dynamic constraints of vehicle motions. We now examine the effects of dynamic constraints on choices of successor points.

In this paper, the flight vehicle is considered as a point mass with state variables of position x, y, altitude h, speed V, and heading Ψ. The following dynamic constraints are imposed on the average motion:

$$V_{\min} \le V \le V_{\max}, \qquad a_{\min} \le \dot{V} \le a_{\max}$$
$$|\dot{\Psi}| \le \dot{\Psi}_{\max}, \qquad |\dot{h}| \le \dot{h}_{\max} \tag{37}$$

where it is assumed $a_{\min} < 0$. In addition, the maximum altitude change and the maximum heading angle change from the current point to a successor point are bounded,

$$|\Delta h| \le \Delta H_{\max}, \qquad |\Delta\Psi| \le \Delta\Psi_{\max} \tag{38}$$

where all of the bounds can be defined as functions of locations, that is, $V_{\min}(x, y, h)$, for flexibility.

Let $P(x_p, y_p, h_p, t_p; V_p)$, $C(x_c, y_c, h_c, t_c; V_c)$, and $S(x_s, y_s, h_s, t_s; V_s)$ represent parent, current, and a potential successor node point, respectively. Altitudes of feasible successor points should satisfy

$$|h_s - h_c| \le \min\{\dot{h}_{\max}(\triangle d_{\text{cs}}/V_c), \Delta H_{\max}\} \tag{39}$$

where the distance from the current point to a successor point is given by

$$\triangle d_{\text{cs}} = \sqrt{(x_s - x_c)^2 + (y_s - y_c)^2 + (h_s - h_c)^2} \tag{40}$$

The heading orientations of the line segments PC and CS are represented by Ψ_{pc} and Ψ_{cs}, respectively, and can be determined from

$$\sin\Psi_{\text{pc}} = \frac{x_c - x_p}{\sqrt{(x_c - x_p)^2 + (y_c - y_p)^2}} \tag{41}$$

$$\sin\Psi_{\text{cs}} = \frac{x_s - x_c}{\sqrt{(x_s - x_c)^2 + (y_s - y_c)^2}} \tag{42}$$

Constraints on the heading angle change from the current point to a successor point as required by Eqs. (37) and (38) result in

$$|\Psi_{\text{cs}} - \Psi_{\text{pc}}| \le \min\{\dot{\Psi}_{\max}(\triangle d_{\text{cs}}/V_c), \Delta\Psi_{\max}\} \tag{43}$$

If $x_s = x_c$ and $y_s = y_c$, but $h_s \ne h_c$, there is no constraints on the heading angle change.

Finally, the speed constraints require that

$$V_s \ge \max\left\{\sqrt{V_c^2 + 2a_{\min}\triangle d_{\text{cs}}}, V_{\min}\right\} \tag{44}$$

$$V_s \le \min\left\{\sqrt{V_c^2 + 2a_{\max}\triangle d_{\text{cs}}}, V_{\max}\right\} \tag{45}$$

C. Feasible Range of Time Increments

For a given flight distance $\triangle d_{\text{cs}}$ and specified flight speeds at two endpoints, the time-of-flight between the two points is fixed. As a result, limits on the flight speed at a successor point define a feasible range for the time of arrival at the successor point,

$$t_s = t_c + \tau_s \tag{46}$$

where $\tau_s = \ell\triangle t, (\ell+1)\triangle t, \ldots, m\triangle t$ for appropriate integers ℓ and m such that

$$\tau_{\min} \le \tau_s \le \tau_{\max} \tag{47}$$

To achieve the shortest flight time between two grid points, the vehicle would increase its speed using the maximum acceleration possible and then stay at the maximum allowed speed. After some derivation, we have

$$\tau_{\min} = \begin{cases} \left(\sqrt{V_c^2 + 2a_{\max}\triangle d_{\text{cs}}} - V_c\right)\big/ a_{\max}, & \triangle d_{\text{cs}} \le d_{t1} \\ (V_{\max} - V_c)/a_{\max} + (\triangle d_{\text{cs}} - d_{t1})/V_{\max}, & \triangle d_{\text{cs}} > d_{t1} \end{cases} \tag{48}$$

where d_{t1} determines the distance required to reach the maximum speed $V_{\max}(x_s, y_s, h_s)$ at a successor point from the current speed V_c using the maximum acceleration $a_{\max}$,

$$d_{t1} = \frac{V_{\max}^2 - V_c^2}{2a_{\max}} \tag{49}$$

To use the longest flight time feasible between two grid points, the vehicle would do the opposite provided the minimum allowed speed is nonzero. If $V_{\min} = 0$, on the other hand, the vehicle can only decelerate gradually to reach the zero speed at the successor point.

$$\tau_{\max} = \begin{cases} \left(\sqrt{V_c^2 + 2a_{\min}\triangle d_{\text{cs}}} - V_c\right)\big/ a_{\min}, & \triangle d_{\text{cs}} \le d_{t2} \\ \min\{T - t_c, (V_{\min} - V_c)/a_{\min} \\ \quad + (\triangle d_{\text{cs}} - d_{t2})/V_{\min}\}, & \triangle d_{\text{cs}} > d_{t2} \end{cases} \tag{50}$$

where d_{t2} is the distance needed to reach the minimum speed $V_{\min}(x_s, y_s, h_s)$ at a successor point from the current speed V_c using the minimum acceleration $a_{\min}$ (or maximum deceleration),

$$d_{t2} = \frac{V_{\min}^2 - V_c^2}{2a_{\min}} \tag{51}$$

If t_s is the time of arrival at a successor point, the corresponding speed V_s at this successor point is given by

$$V_s = 2\triangle d_{\text{cs}}/\tau_s - V_c \tag{52}$$

truncated by the allowable bounds on the flight speed

$$V_s = \begin{cases} V_{\max}, & \text{if} \quad V_s > V_{\max} \\ V_{\min}, & \text{if} \quad V_s < V_{\min} \end{cases} \tag{53}$$

D. Total Cost and Past Cost

The total cost at a selected node is the sum of the past cost and a heuristic cost,

$$\text{Total cost} = P_{\text{cost}} + W \cdot H_{\text{cost}} \tag{54}$$

Past cost is the actual value of the cost in Eq. (1) from the starting point to reach the selected grid point, whereas the heuristic cost is an estimation of the cost in Eq. (1) (or the cost-to-go) from the selected point to the specified goal state. $W > 0$ is a weighting factor.

The past cost is accumulative along the path and, thus, depends on the specific path traversed. The incremental path distance from the parent point to the current point is given by

$$\triangle d_{\text{pc}} = \sqrt{(x_c - x_p)^2 + (y_c - y_p)^2 + (h_c - h_p)^2} \tag{55}$$

It is assumed that the autonomous vehicle changes speed linearly between the two points. The average speed between the two points is, therefore,

$$\bar{V} = (V_c + V_p)/2 \tag{56}$$

and the incremental flight time is given by

$$\triangle t_{\text{pc}} = \triangle d_{\text{pc}}/\bar{V} \tag{57}$$

As a result, the past cost of the current node is given by

$$d_c = d_p + \triangle d_{\text{pc}}, \qquad t_c = t_p + \triangle t_{\text{pc}} \tag{58}$$

and from Eq. (1),

$$P_{\text{cost}} = K_d \cdot d_c + K_t \cdot t_c \tag{59}$$

E. Choices of Heuristic Functions

The proper use of a heuristic function can effectively reduce the size of a huge search problem, and its choice is vitally important to the convergence rate of an A^* search algorithm. The heuristic cost takes the same form as the past cost. For the A^* search scheme to be optimal, the heuristic cost should never overestimate the actual cost and should be zero at the goal state. In this paper, the distance term of the heuristic function is selected to be the straight line distance from current location x_c, y_c, h_c to the goal location x_g, y_g, h_g,

$$d = \sqrt{(x_c - x_g)^2 + (y_c - y_g)^2 + (h_c - h_g)^2} \tag{60}$$

The time term is selected to the shortest feasible time for the vehicle to reach the goal location from current position. An estimate for the shortest feasible time is obtained by assuming that the autonomous vehicle would fly at the maximum possible speed profile from the current position to the goal location. Expressions for an estimate of the shortest time-to-go depend on whether there is a speed constraint at the goal state and how close the current location is to the goal location.

When there is no constraint on the goal speed, the speed profile that produces the shortest time-to-go heuristic and satisfies dynamic constraints of vehicle flight is shown in Fig. 8. Here we define a critical distance $d_{\text{cr},0}$, which is the distance required to reach the maximum speed $V_{\max}$ from the current speed V_c using the maximum acceleration $a_{\max}$,

$$d_{\text{cr},0} = \frac{V_{\max}^2 - V_c^2}{2a_{\max}} \tag{61}$$

If $d < d_{\text{cr},0}$, the autonomous vehicle would simply accelerate at $a_{\max}$ till it reaches the goal location, to arrive at the goal location as soon as possible. We have

$$V_g^2 = V_c^2 + 2a_{\max}d, \qquad t_f = (V_g - V_c)/a_{\max} \tag{62}$$

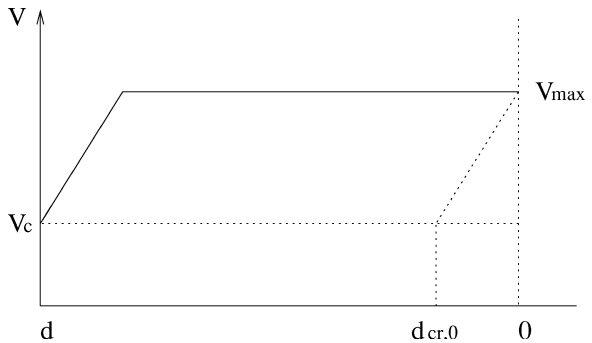

Fig. 8 Speed profile for shortest time-to-go without goal speed constraint.

If $d \geq d_{cr,0}$, on the other hand, there would be two phases in the shortest time-to-go speed profile. An estimate of the shortest time-to-go is given by

$$t_f = (V_{\max} - V_c)/a_{\max} + (d - d_{cr,0})/V_{\max} \tag{63}$$

where the first term represents the time needed for the autonomous vehicle to accelerate from V_c to $V_{\max}$ and the second term is the time needed for the autonomous vehicle to maintain the speed of $V_{\max}$ till it reaches the goal location. If the autonomous vehicle is at the goal location x_g, y_g, h_g, then $d = 0$ in Eq. (62), $V_g = V_c$, and, thus, $t_f = 0$. In other words, the heuristic time-to-go becomes zero at the goal state. The heuristic time-to-go obviously underestimates the actual time-to-go. Therefore, it satisfies requirements of the A^* algorithm.

When there is a specified speed at the goal state, the problem of developing a heuristic time-to-go becomes a little more complicated. Let V_g represent the specified speed at the goal state: $V_{\min} \leq V_g \leq V_{\max}$. Figure 9 shows speed profiles for the three possible cases of $V_g > V_c$, $V_g < V_c$, and $V_c = V_g$. We need to define two

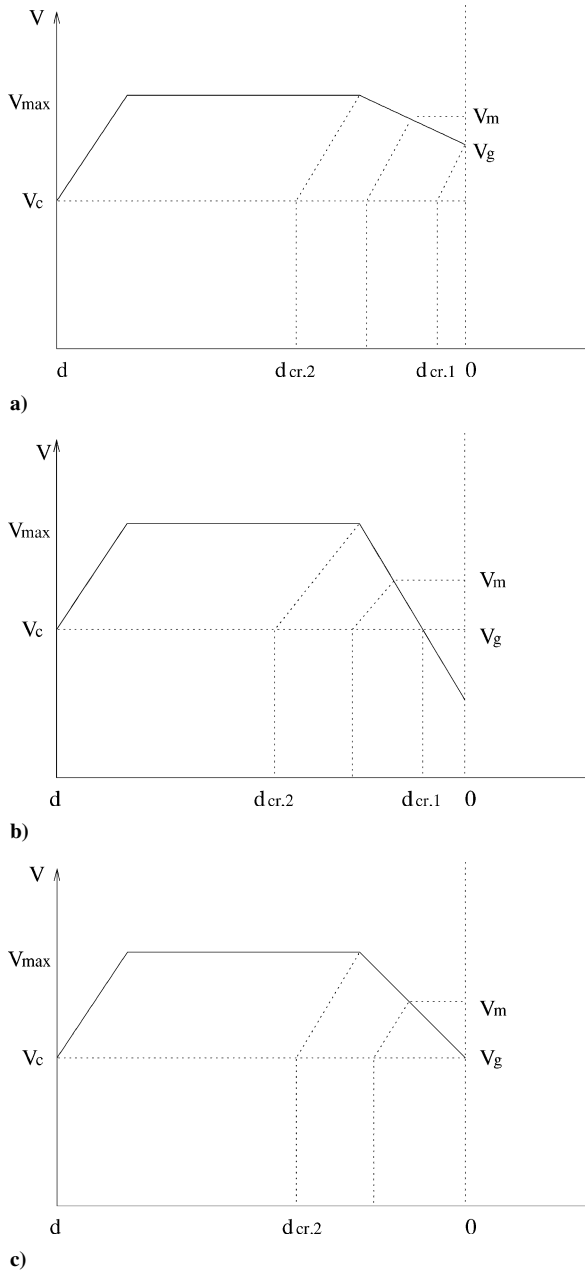

Fig. 9 Speed profiles for estimating shortest time-to-go with a specified goal speed: a) $V_g > V_c$, b) $V_g < V_c$, and c) $V_g = V_c$.

critical distances as shown in Fig. 9 for estimating the shortest time-to-go. Here, $d_{cr,1}$ is the distance required to reach V_g from V_c using either $a_{\max}$ or $a_{\min}$, depending on the relation between the values of V_g and V_c,

$$d_{cr,1} = \begin{cases} \left(V_g^2 - V_c^2\right)/2a_{\max}, & V_g > V_c \\ \left(V_g^2 - V_c^2\right)/2a_{\min}, & V_g < V_c \\ 0, & V_g = V_c \end{cases} \tag{64}$$

Also, $d_{cr,2}$ is the sum of the distance required to accelerate till $V_{\max}$ from current V_c and the distance used to decelerate from $V_{\max}$ to V_g. For all cases,

$$d_{cr,2} = \frac{V_{\max}^2 - V_c^2}{2a_{\max}} + \frac{V_g^2 - V_{\max}^2}{2a_{\min}} \tag{65}$$

If $V_g \neq V_c$ in Fig. 9, an estimated shortest time-to-go depends on the distance from the current point to the goal point. 1) If $d < d_{cr,1}$, there is just one phase and the estimated shortest time-to-go can be given as

$$t_f = \begin{cases} (V_g - V_c)/a_{\max}, & V_g > V_c \\ (V_g - V_c)/a_{\min}, & V_g < V_c \end{cases} \tag{66}$$

2) If $d_{cr,1} \leq d < d_{cr,2}$, on the other hand, there are two phases in the assumed speed profile. In the first phase, the autonomous vehicle would accelerate at $a_{\max}$ till it reaches some speed V_m, it would then decelerate at $a_{\min}$. The time at which when the acceleration would turn into a deceleration is determined from

$$t_1 = (V_m - V_c)/a_{\max} \tag{67}$$

$$d_1 = V_c \cdot t_1 + \tfrac{1}{2} \cdot a_{\max} \cdot t_1^2 \tag{68}$$

During the second phase, the autonomous vehicle would slow down at $a_{\min}$ till it achieves V_g at the goal location,

$$t_2 = (V_g - V_m)/a_{\min} \tag{69}$$

$$d_2 = V_m \cdot t_2 + \tfrac{1}{2} \cdot a_{\min} \cdot t_2^2 \tag{70}$$

Because $d_1 + d_2 = d$, we have

$$V_m = \sqrt{\frac{2da_{\max}a_{\min} + V_c^2 a_{\min} - V_g^2 a_{\max}}{a_{\min} - a_{\max}}} \tag{71}$$

and the estimated minimum time-to-go is given by

$$t_f = t_1 + t_2 \tag{72}$$

3) Finally, if $d \geq d_{cr,2}$, there are three phases in the assumed speed profile. During the first phase, the autonomous vehicle would accelerate from V_c to $V_{\max}$,

$$t_1 = (V_{\max} - V_c)/a_{\max} \tag{73}$$

$$d_1 = V_c \cdot t_1 + \tfrac{1}{2} \cdot a_{\max} \cdot t_1^2 \tag{74}$$

During the third phase of deceleration, the autonomous vehicle would slow down from $V_{\max}$ to V_g,

$$t_3 = (V_g - V_{\max})/a_{\min} \tag{75}$$

$$d_3 = V_{\max} \cdot t_3 + \tfrac{1}{2} \cdot a_{\min} \cdot t_3^2 \tag{76}$$

During the middle phase, the vehicle would move at the speed $V_{\max}$,

$$t_2 = d_2/V_{\max} \tag{77}$$

where

$$d_2 = d - d_1 - d_3 \tag{78}$$

As a result, the estimated shortest time-to-go is given by

$$t_f = t_1 + t_2 + t_3 \tag{79}$$

For the special case where $V_g = V_c$ in Fig. 9, $d_{\mathrm{cr},1} = 0$, and the speed profile for $d < d_{\mathrm{cr},2}$ and $d \geq d_{\mathrm{cr},2}$ are the same as in the preceding discussion. The preceding heuristic time-to-go also becomes zero at the goal state and underestimates the actual time-to-go. Thus, it satisfies requirements of the A^* algorithm.

VII. Numerical Results

Numerical examples are now presented to illustrate applications of the proposed discrete search strategy. The following performance parameters are assumed imitating those of an autonomous helicopter model: $V_{\min} = 0$, $V_{\max} = 100$ ft/s, $a_{\min} = -0.1\,g$, $a_{\max} = 0.1\,g$, $\dot{\Psi}_{\max} = 40$ deg/s, $\dot{h}_{\max} = 20$ ft/s, $(\mathrm{d}h/\mathrm{d}x)_{\max} = \tan 50$ deg, and $(\mathrm{d}\Psi/\mathrm{d}x)_{\max} = \tan 30$ deg. The longitudinal, lateral, and vertical dimension of the geometric search space are selected to be $L = 10^4$ ft, $D = 2500$ ft, and $H = 2500$ ft, to mimic a downtown environment in which the autonomous vehicle needs to fly from one end to the other. Buildings and other constructions are represented by basic obstacle elements and their combinations. The flight-time range is selected to be $T = 200$ s.

Choices of the discretization grid sizes affect the compromise between computational time and trajectory smoothness. In comparison, the choice of the successor generation depth N_s can have a nonlinear effect on the convergence of the trajectory generation process. A smaller N_s results in a smaller number of successor points for consideration at each step, but a small number of successor point choices may hinder the progress toward the goal state and, thus, slow down the overall problem convergence and vice versa. In the following examples, it is assumed that $N_x = 30$, $N_y = 30$, $N_h = 30$, $N_T = 200$, and $N_s = 3$. All computations were performed on a 2000-era Sun workstation.

Figure 10 shows the resulting optimal trajectories in the presence of six stationary obstacles, where four ellipsoid elements are specified with the following parameters:

$$E_1 : x_c = 2000, \quad y_c = 0, \quad h_c = 0$$

$$a = 200, \quad b = 1000, \quad c = 200$$

$$E_2 : x_c = 3000, \quad y_c = 100, \quad h_c = 250$$

$$a = 500, \quad b = 500, \quad c = 200$$

$$E_3 : x_c = 7000, \quad y_c = 0, \quad h_c = 0$$

$$a = 600, \quad b = 600, \quad c = 500$$

$$E_4 : x_c = 8000, \quad y_c = -1000, \quad h_c = 300$$

$$a = 400, \quad b = 400, \quad c = 400$$

one cuboid element is specified with

$$C_u : x_c = 5000, \quad y_c = -1000, \quad h_c = 250$$

$$a = 500, \quad b = 1200, \quad c = 1000$$

and one cylinder element is specified with

$$C_y : x_c = 3000, \quad y_c = 100, \quad h_c = 0$$

$$a = 500, \quad b = 500, \quad c = 250$$

where all location and dimension parameters are measured in feet and all orientation angles are assumed to be $\phi = 0$, $\theta = 0$, and $\psi = 0$. The combination of one ellipsoid element with the cylinder element is assumed in reference of a metrodome structure. In this example, the initial flight speed is assumed to be $V_0 = 60$ ft/s. The optimization performance index is selected to have $K_d = 0.5$ and $K_t = 0.5$ and $W = 100$ in Eq. (54). The trajectory generation process converged in around 21,824 ms. The final flight time is $t_f = 134$ s.

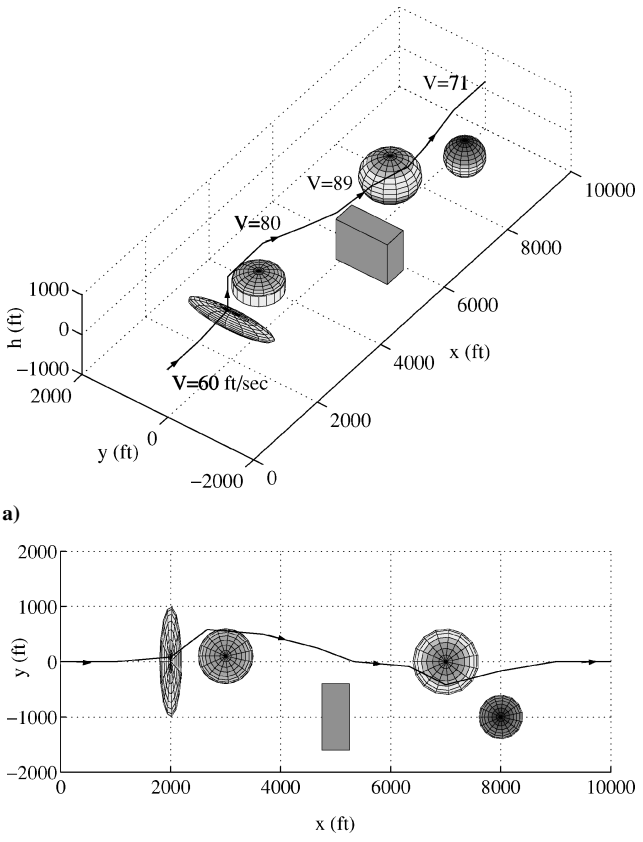

Fig. 10 Optimal flight trajectory in stationary obstacles: a) three-dimensional view and b) top view.

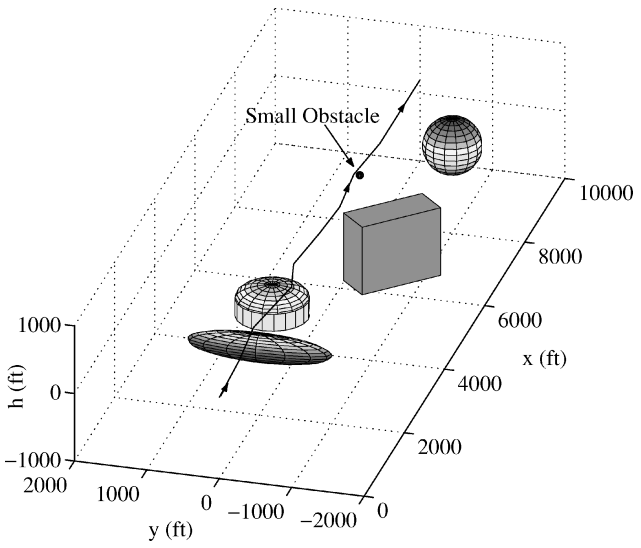

Fig. 11 Optimal trajectory avoiding a small obstacle.

Figure 11 shows that the proposed search algorithm can avoid obstacles smaller than the smallest grid size. In this example, the third ellipsoid obstacle in the preceding example is replaced by a small ellipsoid with

$$E_3' : x_c = 7000, \quad y_c = 0, \quad h_c = 200, \quad a = 50$$

$$b = 50, \quad c = 50, \quad \phi = 0, \quad \theta = 0, \quad \psi = 0$$

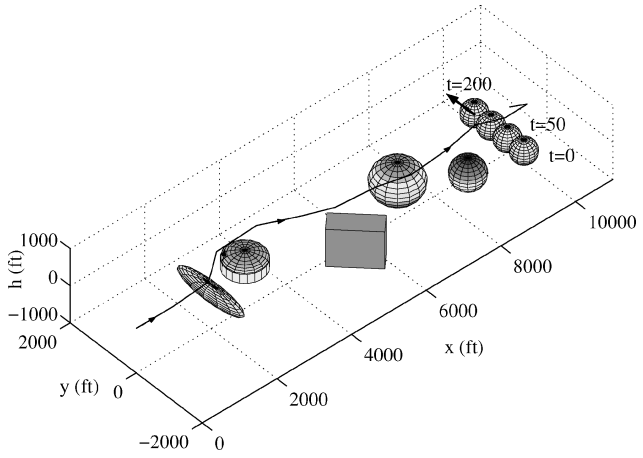

Fig. 12 Optimal trajectory avoiding a moving obstacle.

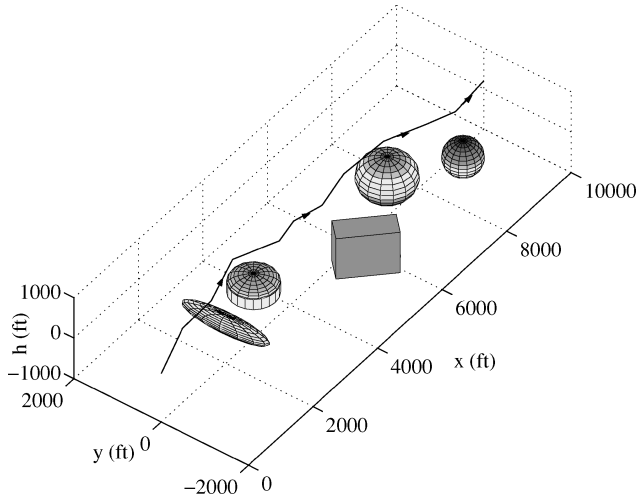

Fig. 13 Optimal trajectory with varying maximum speed fields: shortest time flight with $K_d = 0.01$ and $K_t = 0.9$.

In addition, the horizontal orientation of the cuboid obstacle is changed to (just to verify the algorithm)

$$C'_u : \psi = \pi/4$$

The solution process converged in around 22 s. The final flight time is $t_f = 125$ s.

In Fig. 12, trajectory planning in the presence of moving obstacles is studied. Parameters of the stationary obstacles are the same as in the first example, except that the orientation angle of the cuboid obstacle is changed to $C'_u : \psi = \pi/4$. A moving ellipsoid obstacle is introduced to the flight environment. The center location of the moving obstacle is assumed to be a function of time as $x_c(t) = 9500$, $y_c(t) = -1000 + 10t$, and $h_c(t) = 0$, and the dimensions are $a = 300$, $b = 300$, and $c = 300$. The moving obstacle together with stationary obstacles are avoided successfully, and the solution process converged in around 167 s. The final flight time is $t_f = 133$ s.

The fourth example illustrates the tradeoff between flight distance and time when the maximum allowed flight speeds are location dependent. In this example, it is assumed

$$V_{\max}(x, y, h) = 100 + 200(2y/D) \tag{80}$$

In other words, the maximum allowed flight speed is larger toward the outer sides of the specified flight space. If it is desirable to achieve a shorter flight time ($K_d = 0.01$ and $K_t = 0.9$), the vehicle tends to fly on the outer sides to take advantage of the larger allowed maximum speeds, as shown in Fig. 13, where the flight time is $t_f = 124$ s but the flight distance is larger. If it is desirable to follow a shorter flight path ($K_d = 0.9$ and $K_t = 0.01$), the solution trajectory gives a shorter flight distance with longer flight time: $t_f = 145$ s, as shown in Fig. 14.

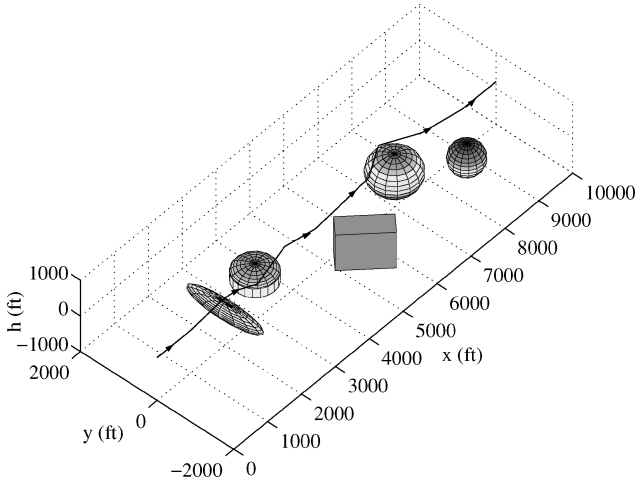

Fig. 14 Optimal trajectory with varying maximum speed fields: shortest distance flight with $K_d = 0.9$ and $K_t = 0.01$.

Numerous other examples with different combinations of obstacles were studied. These examples demonstrate that the proposed discrete search strategy can be used to generate feasible four-dimensional flight trajectories. Appropriate tradeoffs may need to be made between levels of global optimality and computational time through the choices of algorithm parameters such as the weighting in the total cost, grid sizes, and the successor generation depth, as well as specifications of the flight space.

The use of the weighting W in Eq. (54) makes it possible to examine different search strategies in the generation of flight trajectories. $W = 1$ corresponds to the standard A^* algorithm. When W is very large, for example, $W = 100$, the algorithm approaches the greedy search strategy, where only heuristic cost is used in determining which next node to consider. For the current four-dimensional trajectory generation problem, the greedy search algorithm tends to produce solutions fast at the expense of local optimality. When W is very small, on the other hand, the algorithm amounts to a uniform-cost search strategy. Experiences show that the uniform-cost search strategy can take a substantial amount of computational time. A variation of the A^* strategy with dynamic weighting was also studied, in which the weight W is varied as a function of location. Specifically in the current trajectory generations, $W = 100$ is used when the grid point under consideration is still far from the goal point, and $W = 1$ is used toward the goal point. This dynamic weighting has the advantage of encouraging grid point expansions toward the goal point and thus improving solution convergence speed. Overall, it is found that the A^* search strategy and its variations can be used effectively to generate feasible four-dimensional flight trajectories for UAV operations.

VIII. Discussion

In this paper, the issue of autonomous aerospace vehicle flight in an uncertain environment is not directly addressed. This is an important aspect of practical applications of any trajectory generation algorithm. In coping with uncertainties in a flight environment, the proposed discrete search method may be used periodically to regenerate new trajectory plans when knowledge of the environment is updated. This approach does not affect the application of the A^* algorithm. Alternatively, a stochastic search scheme may be used together with the proposed representations of obstacles and conflicts and the discretization of the search space. For example, a stochastic dynamic programming scheme may be used for the same search space discretization and obstacle representation. Further studies are needed on the proposed discrete search strategy to understand necessary modifications and its performances in the presence of uncertainties in the flight environment.

Real-time properties of the developed discrete search strategy are not yet studied. To facilitate real-time applications, bounds on

actual computational times must be developed. For a given onboard computer system, a sufficiently short computational time can be achieved at the expense of compromising the global optimality, resulting in locally optimal solutions. The proposed discrete search solution strategy is promising in delivering trajectory solutions within a specified period of time, but further studies are needed to establish relationships among algorithm parameter selections, computational times, and degree of optimality.

IX. Conclusions

This paper presents a discrete search optimization strategy for generations of four-dimensional trajectories onboard a single autonomous aerospace vehicle. In this strategy, a four-dimensional search space is defined and discretized in both space and time. Obstacles and potential conflicts are represented by several basic shapes and/or their combinations. Mathematical conditions are developed for a given point as well as a trajectory segment between two points to be outside of an obstacle. The A^* search technique is used to obtain trajectory solutions, in which successor trajectory points are selected that both avoid obstacles and satisfy dynamic motion constraints of the vehicle. A linear combination of flight distance and time is optimized in the trajectory generation process. At each search step of the A^* algorithm, the total of past cost and a heuristic function estimating the future cost-to-go is minimized. The heuristic distance function is selected to be the straight line distance from the current location to the goal location, whereas the heuristic time is selected to be the shortest flight time possible from the current location to the goal location.

Numerical examples show that the proposed discrete search strategy can handle a wide range of obstacle and conflict scenarios. It is able to generate feasible flight trajectories rapidly and has the potential to be used for real-time trajectory planning. In specific implementations, parameters of the algorithm need to be selected to achieve a desired level of tradeoff between computational speed and global optimality.

Acknowledgments

This research is supported by the Rotorcraft Division of NASA Ames Research Center under NGG 2-1428, monitored by M. Whalley. The authors thank anonymous reviewers for many helpful comments.

References

[1]Fahlstrom, P. G., and Gleason, T. J., *Introduction to UAV Systems*, UAV Systems, Columbia, MD, June 1998, pp. I-1–I-11.

[2]Krozel, J. A., "Search Problems in Mission Planning and Navigation of Autonomous Aircraft," M.S. Thesis, School of Aeronautics, Purdue Univ., Layfayette, IN, May 1988.

[3]Chandler, P. R., Rasmussen, S., and Pachter, M., "UAV Cooperative Path Planning," AIAA Paper 2000-4370, Aug. 2000.

[4]McLain, T. W., and Beard, R. W., "Trajectory Planning for Coordinated Rendezvous of Unmanned Air Vehicles," AIAA Paper 2000-4369, Aug. 2000.

[5]Judd, K. B., and McLain, T. W., "Spline-Based Path Planning for Unmanned Air Vehicles," AIAA Paper 2001-4238, Aug. 2001.

[6]Frazzoli, E., Dahleh, M. A., and Feron, E., "Real-Time Motion Planning for Agile Autonomous Vehicles," *Journal of Guidance, Control, and Dynamics*, Vol. 25, No. 1, 2002, pp. 116–129.

[7]Latombe, J.-C., *Robot Motion Planning*, Kluwer Academic, Norwell, MA, 1991.

[8]Faiz, N., Agrawal, S. K., and Murray, R. M., "Trajectory Planning of Differentially Flat Systems with Dynamics and Inequalities," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 2, 2001, pp. 219–227.

[9]Mettler, B., Tischler, M. B., and Kanade, T., "System Identification Modeling of a Small-Scale Unmanned Rotorcraft for Flight Control Design," *Journal of the American Helicopter Society*, Vol. 47, No. 1, 2002, pp. 50–63.

[10]Mettler, B., Kanade, T., Tischler, M. B., and Messner, W., "Attitude Control Optimization for a Small-Scale Unmanned Helicopter," AIAA Paper 2000-4059, Aug. 2000.

[11]Shim, D. H., Kim, H. J., and Sastry, S., "Hierarchical Control System Synthesis for Rotorcraft-Based Unmanned Aerial Vehicles," AIAA Paper 2000-4057, Aug. 2000.

[12]Sasiadek, J. Z., and Duleba, I., "3D Local Trajectory Planner for UAV," *Journal of Intelligent and Robotic Systems*, Vol. 29, No. 2, 2000, pp. 191–210.

[13]Gupta, K., and Del Pobil, A. P. (eds.), *Practical Motion Planning in Robotics*, Wiley, Chichester, U.K., 1998, Pt. 1.

[14]Kant, K., and Zucker, S., "Toward Efficient Trajectory Planning: the Path-Velocity Decomposition," *International Journal of Robotics Research*, Vol. 5, No. 3, 1986, pp. 72–89.

[15]Russell, S., and Norvig, P., *Artificial Intelligence, A Modern Approach*, Prentice–Hall, Upper Saddle River, NJ, 1995, pp. 96–101.