Robotica (2012) volume 30, pp. 773–781. © Cambridge University Press 2011
doi:10.1017/S0263574711000993

Three-dimensional path planning for unmanned aerial vehicle based on linear programming

Yang Chen†‡§, Jianda Han†* and Xingang Zhao†

†*State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, P.R. China*
‡*School of Information Science and Engineering, Wuhan University of Science and Technology, Wuhan 430081, P.R. China*
§*Graduate School, Chinese Academy of Sciences, Beijing 100039, P.R. China*

(Accepted August 11, 2011. First published online: September 15, 2011)

SUMMARY
In this paper, an approach based on linear programming (LP) is proposed for path planning in three-dimensional space, in which an aerial vehicle is requested to pursue a target while avoiding static or dynamic obstacles. This problem is very meaningful for many aerial robots, such as unmanned aerial vehicles. First, the tasks of target-pursuit and obstacle-avoidance are modelled with linear constraints in relative coordination according to LP formulation. Then, two weighted cost functions, representing the optimal velocity resolution, are integrated into the final objective function. This resolution, defined to achieve the optimal velocity, deals with the optimization of a pair of orthogonal vectors. Some constraints, such as boundaries of the vehicle velocity, acceleration, sensor range, and flying height, are considered in this method. A number of simulations, under static and dynamic environments, are carried out to validate the performance of generating optimal trajectory in real time. Compared with ant colony optimization algorithm and genetic algorithm, our method has less parameters to tune and can achieve better performance in real-time application.

KEYWORDS: Path planning in three dimensions; Obstacle avoidance; Target pursuit; Uncertain environment.

1. Introduction

Path planning is fundamentally important for an autonomous vehicle. Generally, for an aerial vehicle A, a path is to be planned such that the target G can be reached and the obstacles O in sensor-effective range can be avoided. If real time is demanded, path planning plays an important role in application of navigation and surveillance.

A few methods have been proposed to model the task and generate a path. Artificial potential field[1,2] is one of the traditional methods, which is generally used with an octree in three-dimensional (3D) dynamic environment. The mechanism that the vehicle is driven by attractive and repulsive force in a cooperative way is simple but often works efficiently even in dynamic environment. Unfortunately, path-planning approach based on the potential function cannot obtain the optimal objective, which, in turn, cannot guarantee the desired optimal trajectories. Moreover, some additional measures are necessary for the sake of avoiding the undesired local minima. Potential panel method and harmonic potential function can also plan a path in space for an autonomous vehicle,[3–5] but the computing burdens is huge and the real-time performance hardly satisfies the practical requirement.

Inspired by biological intelligence, many approaches, such as ant colony optimization (ACO), particle swarm optimization (PSO), genetic algorithm (GA), evolution algorithm (EA), and their combinations,[6–11] are introduced to solve the path-planning problem in three dimensions. These mostly rely on the stochastic searching, known as non-deterministic algorithm. These methods will eventually find an optimal solution, but no estimate on the time of convergence can be given. Thus, it may take long, even infinite, time to find the best solution.[12] Furthermore, all of these methods have a great number of parameters to tune and that has never been an easy job, particularly when users are short of prior knowledge. Some comparisons[11,12] show that the expensive calculations limit their real application.

Another kind of method is mathematic programming. Based on the relative velocity obstacle method,[13] linear programming (LP), nonlinear programming (NLP), mixed integer linear programming (MILP), and other convex optimization methods have been employed for path planning problem.[14–16] For example, Wang, *et al.*,[14] converted the 2D path planning of an unmanned underwater vehicle to constrained optimization or semi-infinite constrained optimization problem. Zu *et al.*[15] discussed the path planning in two dimensions and an LP method was proposed for the problem of dynamic target pursuit and obstacle avoidance (TPOA). This method tries to plan the variables of linear and angular accelerations of a ground vehicle. Unfortunately, it cannot fit three dimensions because the extension needs to define cylindrical or spherical coordinate systems for the problem that is intractable. Schouwenaars *et al.*[17] proposed an MILP formulation with receding horizon strategy, where a minimum velocity and a limited turn rate of aircraft are constrained. However, their results still focused on two dimensions.

In this paper, we consider the same problem of TPOA, but in three dimensions. To our problem, the uncertain environment has one target G and many obstacles O that are all velocity-changeable with their moving actions. The aim of the vehicle A is to find an optimal path for pursuing

* Corresponding author. E-mail: jdhan@sia.cn

https://doi.org/10.1017/S0263574711000993 Published online by Cambridge University Press

Table I. Nomenclature (see Figs. 1 and 2).

A, G, O	Vehicle, target, obstacle
τ	Planning period
K	Time step
N	Sum of obstacles
$\mathbf{V}_A = (v_{Ax}, v_{Ay}, v_{Az})^{\mathrm{T}}$	Vehicle velocity
$\mathrm{V}_A = \|\mathbf{V}_A\|$	Magnitude of vehicle velocity
$\Delta\mathbf{V}_A = (\Delta v_{Ax}, \Delta v_{Ay}, \Delta v_{Az})^{\mathrm{T}}$	Vehicle acceleration
$\mathbf{V}_O$	Obstacle velocity
$\mathrm{V}_O = \|\mathbf{V}_O\|$	Magnitude of obstacle velocity
$\mathbf{V}_G$	Target velocity
$V_G = \|\mathbf{V}_G\|$	Magnitude of target velocity
$\mathbf{V}_{AO} = \mathbf{V}_A - \mathbf{V}_O = (v_x, v_y, v_z)^{\mathrm{T}}$	Vehicle's relative velocity to the obstacle
$V = \|\mathbf{V}_{AO}\|$	Magnitude of vehicle velocity relative to the obstacle
$\Delta\mathbf{V}_{AO}$	Vehicle acceleration relative to the obstacle
$\mathbf{V}_{AG} = \mathbf{V}_A - \mathbf{V}_G = (v_{Gx}, v_{Gy}, v_{Gz})^{\mathrm{T}}$	Vehicle velocity relative to target G
$V_{AG} = \|\mathbf{V}_{AG}\|$	Magnitude of vehicle's velocity relative to the target
$\Delta\mathbf{V}_{AG}$	Vehicle acceleration relative to the target
$\mathbf{L}_{AO} = (l_x, l_y, l_z)^{\mathrm{T}}$	Vehicle position relative to the obstacle
$L = \|\mathbf{L}_{AO}\|$	Magnitude of vehicle position relative to the obstacle
$\mathbf{L}_{AG} = (l_{Gx}, l_{Gy}, l_{Gz})^{\mathrm{T}}$	Vehicle position relative to target
$L_{AG} = \|\mathbf{L}_{AG}\|$	Magnitude of vehicle position relative to the target
$P = \mathbf{V}_{AO}^{\mathrm{T}}\mathbf{L}_{AO}$	Temporal variable
$P_{AG} = \mathbf{V}_{AG}^{\mathrm{T}}\mathbf{L}_{AG}$	Temporal variable

the target while, at the same time, avoiding collision with obstacles. The position and velocity of movers, including the target and obstacles, are assumed to be known or estimated at current time. To be more specific, we assume that the noise contained in the data can be eliminated with certain filters. But this paper has no intention to discuss the filtering algorithm for achieving the feasible data of on-board sensors. The trajectory from a start to a destination location typically needs to be computed gradually over time in the fashion of receding horizon[17] in which a new waypoint of the total path is computed at each time step by solving an LP problem. The target and obstacles, probably static or dynamic, are modelled by spheres having certain radius for their impact spectrum, and the vehicle is modelled by a mass point. In order to optimize the vehicle's acceleration online, we construct an LP model in Cartesian orthogonal coordinates based on relative velocity space.[15]

This paper is organized as follows. First, we formulate the tasks on object-avoidance and target-pursuit in Sections 2 and 3, respectively. Next, the constraints of dynamics, kinematics, and sensor's limitation are discussed for optimization in Section 4. In Section 5, some simulation and experiment results validate the real-time capability of this novel method comparing with the ACO and GA methods. A brief conclusion is drawn in Section 6.

2. Formulation of Obstacle Avoidance

First, some assumptions and definitions are given, based on which a proposition is derived by formulating the problem of obstacle avoidance. $\mathbf{V}_A$ denotes the vehicle velocity, and $\Delta\mathbf{V}_A$ denotes the vehicle's acceleration. Other symbols and the variables are listed in Table I.

Hypothesis: All obstacles and the target keep their velocities unchanged during the planning period, τ, except that the vehicle can adjust its velocity. In fact, this hypothesis is usually satisfied because τ is short enough for a real spatial vehicle. This treatment is acceptable in a numerical approximation method, because it forms the basis of the derivations. We get $\Delta\mathbf{V}_{AO} = \Delta\mathbf{V}_{AG} = \Delta\mathbf{V}_A$. Here, the symbol Δ denotes the changing quantities during one planning period τ. $\Delta\mathbf{V}_{AG}$ is vehicle's velocity relative to the target. In detail, it is

$$\begin{aligned}(\Delta v_x, \Delta v_y, \Delta v_z) &= (\Delta v_{Gx}, \Delta v_{Gy}, \Delta v_{Gz}) \\ &= (\Delta v_{Ax}, \Delta v_{Ay}, \Delta v_{Az}).\end{aligned} \tag{1}$$

In the remainder of the paper, we use $(\Delta v_x, \Delta v_y, \Delta v_z)$ to represent the other two vectors in Eq. (1).

Definition: The angle between vectors $\mathbf{V}_{AO}$ and $\mathbf{L}_{AO}$, denoted by γ_{AO}, is defined as *collision angle*. *Collision cone* is defined as a cone of relative velocity $\mathbf{V}_{AO}$, which will lead the vehicle to collide with any obstacle, and it is denoted by CC_{AO} (see Fig. 1). Generally, CC_{AO} is in the cone AEF. Referring to the target, the angle between the relative velocity $\mathbf{V}_{AG}$ and relative distance $\mathbf{L}_{AG}$ is defined as *pursuit angle*, denoted by γ_{AG} (see Fig. 2). In both Figs. 1 and 2, the relative coordination is constructed on the mass point of vehicle A. It shows relative parameters between vehicle and other movers.

Proposition: For each obstacle O, under the assumption that its velocity is constant in τ, it will be avoided if the *collision angle*, γ_{AO}, is big enough to make $\mathbf{V}_{AO}$ out of CC_{AO} over the interval τ when the vehicle moves from time step k to $k+1$. See Fig. 1, which suggests that the obstacle hold the following constraints:

$$\gamma_{AOC(k)} \leq \gamma_{AO(k+1)} \leq \pi, \tag{2}$$

https://doi.org/10.1017/S0263574711000993 Published online by Cambridge University Press

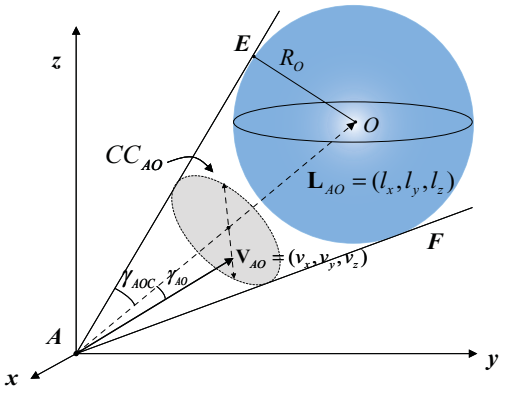


Fig. 1. (Colour online) Collision avoidance in relative space.

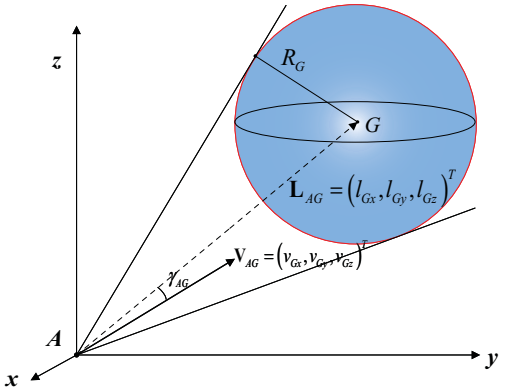


Fig. 2. (Colour online) Target pursuit in relative space.

where $\gamma_{AOC(k)} = \arcsin(R_O/L_{(k)})$, defined as *collision region angle* in time step k.

By geometric calculation, we obtain

$$\gamma_{AO} = \arccos\left(\frac{\mathbf{V}_{AO}\cdot\mathbf{L}_{AO}}{\|\mathbf{V}_{AO}\|\cdot\|\mathbf{L}_{AO}\|}\right) = \arccos\left(\frac{P}{VL}\right), \quad (3)$$

where $V = \|\mathbf{V}_{AO}\| = \sqrt{v_x^2 + v_y^2 + v_z^2}$, $L = \|\mathbf{L}_{AO}\| = \sqrt{l_x^2 + l_y^2 + l_z^2}$, and $P = \mathbf{V}_{AO}^{\mathrm{T}}\mathbf{L}_{AO}$. Here, V, L, and P are the compact versions of V_{AO}, L_{AO}, and P_{AO}. We just discard the subscripts to obtain concise derivations in the coming mathematics. Using Taylor's theorem to linearize the function, we obtain

$$\begin{aligned}\gamma_{AO_{(k+1)}} &= \gamma_{AO_{(k)}} + \Delta\gamma_{AO_{(k)}} \\ &= \gamma_{AO_{(k)}} + \tau\left\{\left(\frac{\partial\gamma_{AO}}{\partial v_x}\right)_{(k)}\frac{dv_x}{dt} + \left(\frac{\partial\gamma_{AO}}{\partial v_y}\right)_{(k)}\frac{dv_y}{dt} \right. \\ &\quad \left. + \left(\frac{\partial\gamma_{AO}}{\partial v_z}\right)_{(k)}\frac{dv_z}{dt}\right\}. \end{aligned} \quad (4)$$

The $\gamma_{AO(k+1)}$ represents the collision angle in step $k+1$ after the movement of the vehicle. The second term in Eq. (4), i.e., increment of *collision angle* in one planning period, can be computed with the following equation:

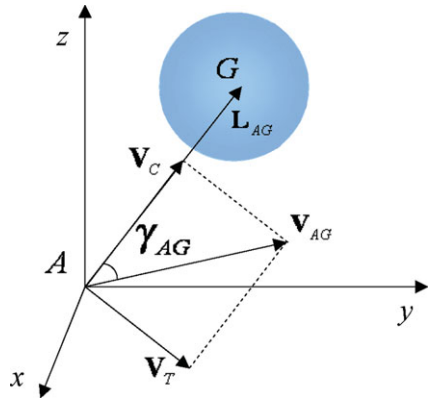


Fig. 3. (Colour online) The relative velocity of the vehicle is resolved into two orthogonal components, $\mathbf{V}_C$ and $\mathbf{V}_T$.

$$\begin{aligned}\Delta\gamma_{AO_{(k)}} = -\frac{\tau}{\sqrt{V^2L^2 - P^2}|_{(k)}}&\left\{\left(l_x - \frac{P}{V^2}v_x\right)_{(k)}\Delta v_x \right. \\ &\left. + \left(l_y - \frac{P}{V^2}v_y\right)_{(k)}\Delta v_y + \left(l_z - \frac{P}{V^2}v_z\right)_{(k)}\Delta v_z\right\}. \end{aligned} \quad (5)$$

In Eq. (4), Δv_x, Δv_y, and Δv_z are the variables we are trying to plan in step k for step $k+1$. Let i denotes the label of the obstacle, then Eq. (2) changes to Eq. (6). $i = 1, 2 \ldots N$, where N stands for the number of obstacles.

$$\gamma_{AOCi(k)} \le \gamma_{AOi(k)} + \Delta\gamma_{AOi(k)} \le \pi, \quad (6)$$

where

$$\begin{aligned}\Delta\gamma_{AOi_{(k)}} = -\frac{\tau}{\sqrt{V^2L^2 - P^2}|_{i,(k)}}&\left\{\left(l_x - \frac{P}{V^2}v_x\right)_{i,(k)}\Delta v_x \right. \\ &\left. + \left(l_y - \frac{P}{V^2}v_y\right)_{i,(k)}\Delta v_y + \left(l_z - \frac{P}{V^2}v_z\right)_{i,(k)}\Delta v_z.\right. \end{aligned} \quad (7)$$

3. Formulation of Target Pursuit

The vehicle's relative velocity to the target can be resolved into a pair of orthogonal components, as shown in Fig. 3. One component, $\mathbf{V}_{\mathrm{C}}$, is in the direction $\mathbf{L}_{AG}$ and is pointed to the target. The other $\mathbf{V}_{\mathrm{T}}$ is orthogonal to $\mathbf{L}_{AG}$. The vehicle is expected to tune its velocity to the optimum. Here, $\mathbf{V}_{\mathrm{C}}$ represents the velocity pointing to the target. If the magnitude of $\mathbf{V}_{\mathrm{C}}$ is positive, the vehicle moves to the target, and if negative, then the vehicle moves apart from the target. $\mathbf{V}_{\mathrm{T}}$ represents the velocity that enables the vehicle moving around the target. Then the policy for the optimal velocity is obvious that $\mathbf{V}_{\mathrm{T}}$ should be minimized, while $\mathbf{V}_{\mathrm{C}}$ is maximized. We formulate them in Eqs. (8) and (9), and will discuss them separately:

$$\min : V_{\mathrm{T}} = \|\mathbf{V}_{AG}\|\sin\gamma_{AG}, \quad (8)$$

$$\max : V_C = \|\mathbf{V}_{AG}\|\cos\gamma_{AG}. \quad (9)$$

https://doi.org/10.1017/S0263574711000993 Published online by Cambridge University Press

3.1. Optimization of V_{T}

The statement of Eq. (8) is equivalent to $\min : V_{\mathrm{T}} = \sqrt{V_{AG}^2 - P_{AG}^2/L_{AG}^2} \geq 0$, where $P_{AG} = \mathbf{V}_{AG}^{\mathrm{T}}\mathbf{L}_{AG}$. Let $V_{\mathrm{T}(k+1)}$ represents the optimal index after the vehicle moves from step k to $k+1$. The first objective function is derived as $\min : V_{\mathrm{T}(k+1)}$. By introducing an auxiliary variable, this statement is equivalent to $\min : q_1$, subject to

$$0 \leq V_{\mathrm{T}(k+1)} \leq q_1, \tag{10}$$

$$0 \leq q_1 \leq V_{AG\max}. \tag{11}$$

$V_{AG\max}$ is the upper bound of q_1 and denotes the maximum of relative velocity between the vehicle and the target. It can be estimated by $V_{AG\max} = V_{A\max} + V_{G\max}$, which occurs when both are at maximum velocity as well as on the opposite directions. Function $V_{\mathrm{T}}(\cdot)$ is calculated with the discrete formal, represented as

$$\begin{aligned} &V_{\mathrm{T}(k+1)} \\ &= V_{\mathrm{T}(k)} + \tau\left(\frac{\partial V_{\mathrm{T}}}{\partial v_{Gx}}\frac{dv_{Gx}}{dt} + \frac{\partial V_{\mathrm{T}}}{\partial v_{Gy}}\frac{dv_{Gy}}{dt} + \frac{\partial V_{\mathrm{T}}}{\partial v_{Gz}}\frac{dv_{Gz}}{dt}\right)_{(k)} \\ &= V_{\mathrm{T}(k)} + \frac{\tau}{V_{\mathrm{T}(k)}}\left\{\left(v_{Gx} - \frac{P_{AG}l_{Gx}}{L_{AG}^2}\right)_k \Delta v_x \right. \\ &\quad \left. + \left(v_{Gy} - \frac{P_{AG}l_{Gy}}{L_{AG}^2}\right)_k \Delta v_y + \left(v_{Gz} - \frac{P_{AG}l_{Gz}}{L_{AG}^2}\right)_k \Delta v_z\right\}. \end{aligned}$$

3.2. Optimization of V_{C}

V_{C} in Eq. (9) can be computed as $V_{\mathrm{C}} = P_{AG}/L_{AG}$. From Eq. (9), we obtain the other objective function, which is $\max : V_{\mathrm{C}(k+1)}$, where $V_{\mathrm{C}(k+1)} \in [-V_{AG\max},\ V_{AG\max}]$. This can be transformed to a tractable formula with the expression of $\min : -V_{\mathrm{C}(k+1)}$. Through an auxiliary variable, we obtain $\min : q_2$, subject to

$$-V_{\mathrm{C}(k+1)} \leq q_2, \tag{12}$$

$$-V_{A\max} - V_{G\max} \leq q_2 \leq V_{A\max} + V_{G\max}. \tag{13}$$

This criterion at time step $k+1$ is calculated by

$$\begin{aligned} V_{\mathrm{C}(k+1)} &= V_{\mathrm{C}(k)} + \tau\left(\frac{\partial V_{\mathrm{C}}}{\partial v_{Gx}}\frac{dv_{Gx}}{dt} + \frac{\partial V_{\mathrm{C}}}{\partial v_{Gy}}\frac{dv_{Gy}}{dt} \right. \\ &\quad \left. + \frac{\partial V_{\mathrm{C}}}{\partial v_{Gz}}\frac{dv_{Gz}}{dt}\right)_{(k)} \\ &= \left(\frac{P_{AG}}{L_{AG}}\right)_{(k)} + \tau\left\{\left(\frac{l_{Gx}}{L_{AG}}\right)_{(k)} \Delta v_{Gx} \right. \\ &\quad \left. + \left(\frac{l_{Gy}}{L_{AG}}\right)_{(k)} \Delta v_{Gy} + \left(\frac{l_{Gz}}{L_{AG}}\right)_{(k)} \Delta v_{Gz}\right\}. \end{aligned}$$

3.3. The final objective function

The two variables have been introduced to the objective function of the LP model, that is

$$\min : \omega q_1 + (1-\omega)q_2, \tag{14}$$

where $\omega \in [0, 1]$ is the weight factor. Generally, the parameter ω can be tuned. Through our experience, Eq. (14) did not depend sensitively on the value of ω. In practice, the value $\omega = 0.5$ works well.

4. LP Model

4.1. Constraints on kinematics and dynamics

The velocity and acceleration of the vehicle should be limited by its specific actuators, drives, and dynamics. We describe these magnitude constraints as $-V_{A\max} \leq v_{Aj} \leq V_{A\max}$, $j = \{x, y, z\}$, and $-\Delta_{\max} \leq \Delta v_j \leq \Delta_{\max}$. Here, $V_{A\max}$ stands for the upper bound of the vehicle's velocity, and v_{Aj} denotes one of the three components of $\mathbf{V}_A$. $\Delta_{\max}$ represents the upper bound of acceleration. Also, the velocity in step $k+1$ is restricted to the bound: $-V_{A\max} \leq v_{Aj} + \Delta v_j \tau \leq V_{A\max}$. The constraints on acceleration components are generated as follows:

$$\begin{aligned} &\max\left\{-\Delta_{\max}, -\frac{1}{\tau}(V_{A\max} + v_{Ax(k)})\right\} \leq \Delta v_x \\ &\quad \leq \min\left\{\Delta_{\max}, \frac{1}{\tau}(V_{A\max} - v_{Ax(k)})\right\}, \\ &\max\left\{-\Delta_{\max}, -\frac{1}{\tau}(V_{A\max} + v_{Ay(k)})\right\} \leq \Delta v_y \\ &\quad \leq \min\left\{\Delta_{\max}, \frac{1}{\tau}(V_{A\max} - v_{Ay(k)})\right\}, \\ &\max\left\{-\Delta_{\max}, -\frac{1}{\tau}(V_{A\max} + v_{Az(k)})\right\} \leq \Delta v_z \\ &\quad \leq \min\left\{\Delta_{\max}, \frac{1}{\tau}(V_{A\max} - v_{Az(k)})\right\}. \end{aligned}$$

These functions, $\max\{\}$ and $\min\{\}$, return the largest and the smallest elements in the brace, respectively.

4.2. Constraints on sensor range and flying height

The distance between the vehicle and any obstacle is detected by certain sensors such as radar, visual tracker, and laser scanner. Each of these sensors has its own range limitation and neither can detect the obstacle out of its range. As a result, an obstacle possibly disappears after being passed by it. Meanwhile, a new obstacle may enter the sensor's horizon randomly. The parameter, λ_i, is assigned to evaluate the threat imposed by an obstacle on the vehicle.

If the relative distance between the vehicle and an obstacle is larger than a threshold, the obstacle O_i is considered as no threat, namely if $L_{AOi} \geq L_{\min i}$, then $\lambda_i = 0$. On the contrary, if $L_{AOi} < L_{\min i}$, then $\lambda_i = 1$. The threshold $L_{\min i}$ is a parameter associated with radius (e.g., a distance of five radiuses). The wider the radius, the larger the threshold.

The aerial vehicle has the flying height restriction, which briefly we indicate with

$$h_{\min} \leq z_{A(k+1)} \leq h_{\max}, \tag{15}$$

where $z_{A(k+1)} = z_{A(k)} + v_{Az(k)}\tau + \Delta v_z \tau^2$.

https://doi.org/10.1017/S0263574711000993 Published online by Cambridge University Press

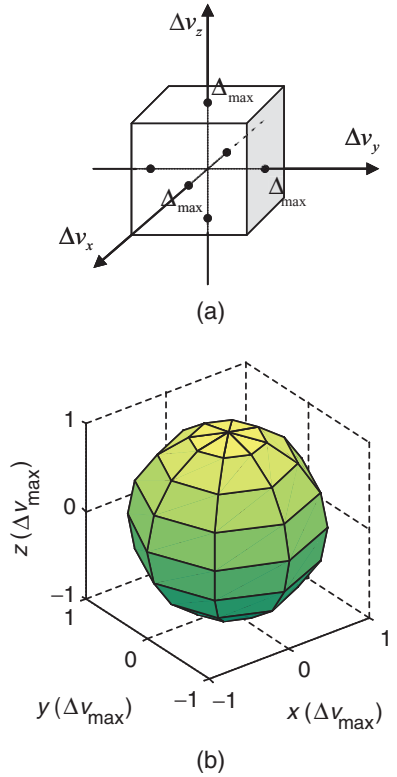


Fig. 4. (Colour online) The searching region. (a) A poor approximation for searching region using a cube; (b) an elaborate approximation with multi-planes ($M = 8$). It illustrates the effect when 64 planes are used during approximation.

4.3. Constraints on feasible region

The feasible region of a vehicle's acceleration is a sphere in Cartesian space for the nearly equivalent length of Δv_x, Δv_y, and Δv_z. This suggests that the constraints of Δv_x, Δv_y, and Δv_z are nonlinear. However, one simple way to represent these constraints in a linear formal way is shown in Eq. (16). The profile of these constraints, a cube, is shown in Fig. 4(a).

$$\begin{cases} -\Delta_{\max} \le \Delta v_x \le \Delta_{\max} \\ -\Delta_{\max} \le \Delta v_y \le \Delta_{\max} \\ -\Delta_{\max} \le \Delta v_z \le \Delta_{\max} \end{cases}. \tag{16}$$

This cube is obviously a poor approximation to the sphere. However, a number of constraints can be used to achieve better approximations. In Fig. 4(b), multi-planes are used to approximate the nonlinear space as shown in Eq. (17). In fact, it is a modified version of 2D,[16] which needs to approximate a circle,

$$\sin\theta_m \cos\phi_n \Delta v_x + \sin\theta_m \sin\phi_n \Delta v_y + \cos\theta_m \Delta v_z \le \Delta_{\max}, \tag{17}$$

where $\phi_n = 2n\pi/M$, $\theta_m = 2m\pi/M$, $m, n = 0, 1, \ldots M-1$, where M denotes the number of planes used for approximation.

4.4. LP method formulation

At this point the LP model is formulized. Then objective function is Eq. (14). All constraints are recapitulated as follows. A block diagram is shown in Fig. 5.

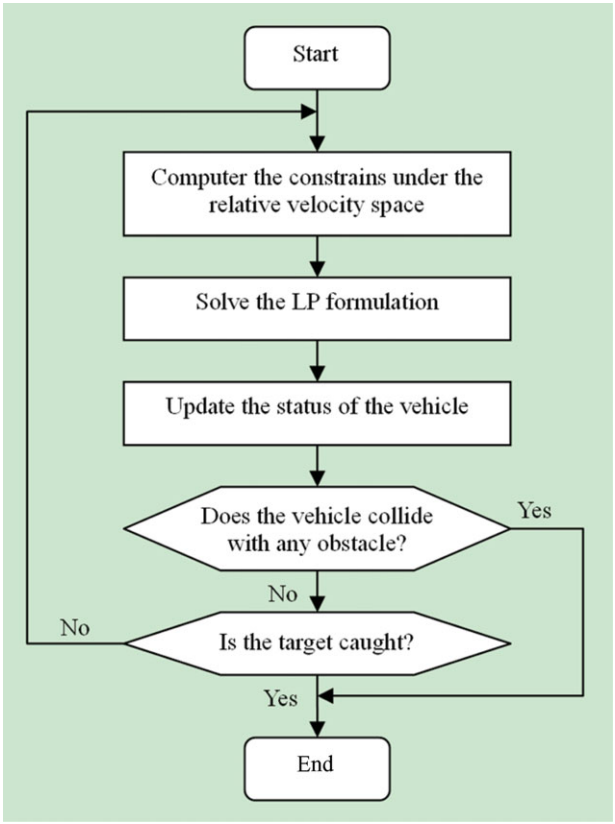


Fig. 5. (Colour online) Block diagram of the experiments.

$$\begin{cases} \lambda_i \cdot \gamma_{AOCi(k)} \le \lambda_i \cdot \gamma_{AOi(k+1)} \le \pi \\ 0 \le V_{T(k+1)} \le q_1 \\ -V_{C(k+1)} \le q_2 \\ 0 \le q_1 \le V_{AG\max} \\ -V_{A\max} - V_{G\max} \le q_2 \le V_{A\max} + V_{G\max} \\ \max\left\{-\Delta_{\max}, -\frac{1}{\tau}(V_{A\max} + v_{Ax(k)})\right\} \le \Delta v_x \\ \quad \le \min\left\{\Delta_{\max}, \frac{1}{\tau}(V_{A\max} - v_{Ax(k)})\right\} \\ \max\left\{-\Delta_{\max}, -\frac{1}{\tau}(V_{A\max} + v_{Ay(k)})\right\} \le \Delta v_y \\ \quad \le \min\left\{\Delta_{\max}, \frac{1}{\tau}(V_{A\max} - v_{Ay(k)})\right\} \\ \max\left\{-\Delta_{\max}, -\frac{1}{\tau}(V_{A\max} + v_{Az(k)})\right\} \le \Delta v_z \\ \quad \le \min\left\{\Delta_{\max}, \frac{1}{\tau}(V_{A\max} - v_{Az(k)})\right\} \\ h_{\min} \le z_{A(k+1)} \le h_{\max} \\ \sin\theta_m \cos\phi_n \Delta v_x + \sin\theta_m \sin\phi_n \Delta v_y + \cos\theta_m \Delta v_z \le \Delta_{\max} \end{cases}.$$

5. Simulation and Experimental Results

The algorithm is programmed with C++ programming language and runs on a PC with WinXP/PentiumIV/2.53GHz/4GB. LP is solved by open library named Qsopt.[18] For a typical environment of three obstacles, the simulation results are given in Figs. 6–8. We list some initial constraints and parameters of the vehicle as shown in Table II.

https://doi.org/10.1017/S0263574711000993 Published online by Cambridge University Press

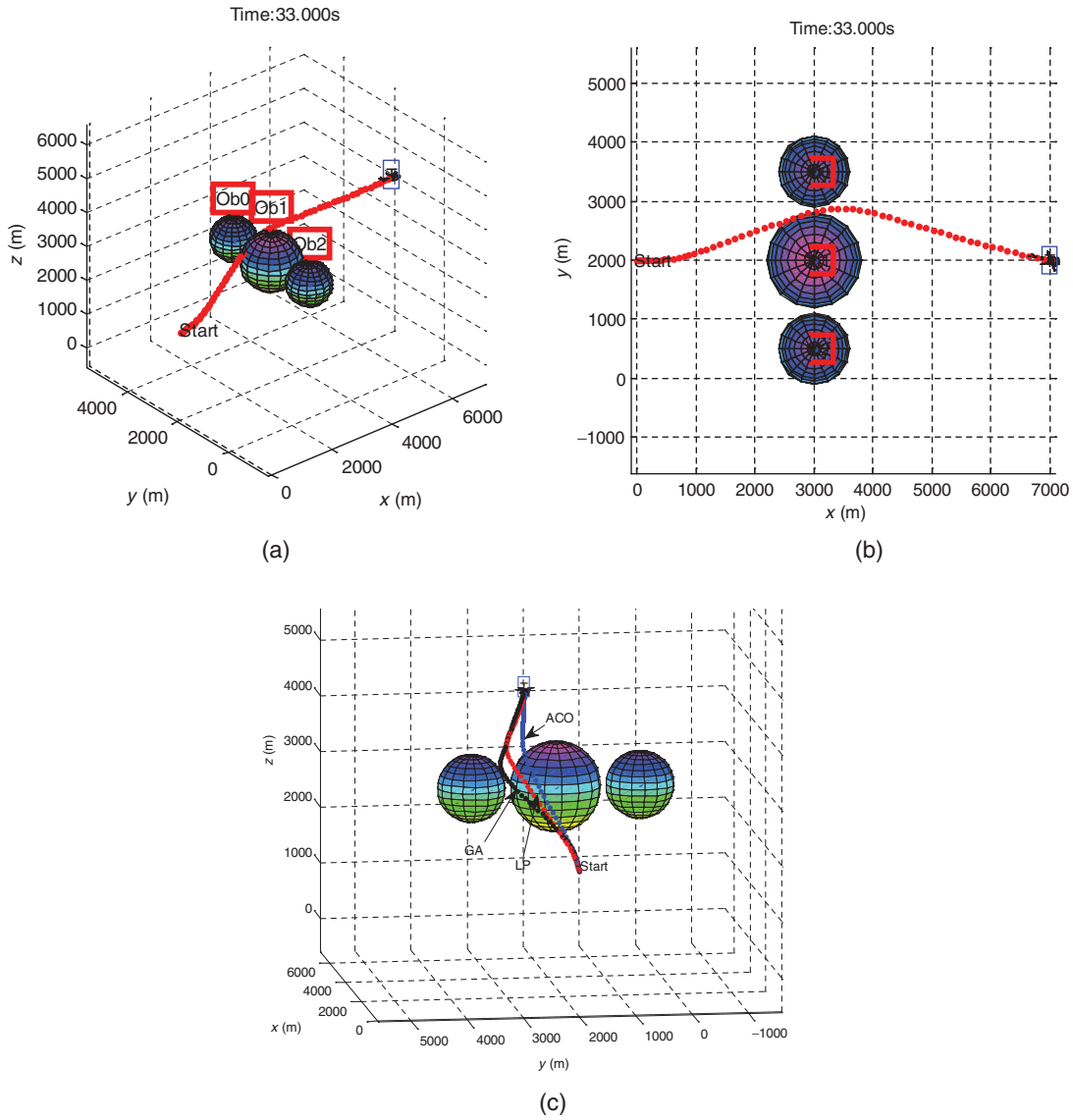


Fig. 6. (Colour online) Path planning in static environment. Spheres denote the obstacles. Dash-dot curve represents the trajectory of the vehicle. (a) Complete path computed by LP; (b) vertical view of LP path; (c) comparing LP path with that from GA and ACO algorithm. LP path: red line. GA path: dark line. ACO path: blue line. We generally run GA and ACO 10 times and draw only the best one as denoted in the picture.

Table II. Some initial constraints and parameters of the vehicle.

$V_{A\max}$	$\Delta v_{\max}$	$h_{\min}$	$h_{\max}$	τ
250 m/s	30 m/s^2	0 m	1.0 e5 m	0.5 s

In the following, we give some simulation results in static and dynamic environments. Meanwhile, comparisons and contrasts result among the algorithms we propose and the ACO and GA methods that are provided in the following tables. Simulation results and comparisons with other algorithms will help justify our contributions. Then a human-in-the-loop simulation is demonstrated, in which a joystick is used to control the target moving in any direction and speed so the LP method will be validated.

5.1. Static environment

This simulation shows that the new method works effectively in a static environment. Table III lists all initial states used by the simulation. Three obstacles are initially placed on the line of the vehicle to the target. The task of the aerial vehicle is moving to the fixed target, which is hidden behind the obstacles. All the obstacles are static. The trajectory planned by our method is shown in Figs. 6(a) and (b). We re-plan this path by the existing algorithms, ACO and GA, as published in the literature.[5,11] Three trajectories are drawn in Fig. 6(c) for contrast purpose. It is obvious that the vehicle can avoid each static obstacle automatically.

Figure 5 shows that the vehicle can evade the static obstacles and find a velocity-optimal path to the target point. When comparing with the GA and ACO algorithms, our method shows better performance in several aspects. Not only the trajectory length is shorter than others, but the algorithm complexity is also inferior over the other two. By using LP planner, the vehicle consumed 32.8 s to catch the target. However, it needs 37.7 s for GA and 42.11 s for ACO. Time complexity of our method is 3.84 ms, while the complexity of GA and ACO is 4.06 ms and 4.43 ms, respectively. The

https://doi.org/10.1017/S0263574711000993 Published online by Cambridge University Press

Table III. Initial parameters for simulation in static environment.

	Vehicle	T (target)	Ob 1 (static)	Ob 2 (static)	Ob 3 (static)
Initial position (m)	(0, 2000, 2000)	(7000, 2000, 4000)	(3000, 3500, 3000)	(3000, 2000, 3000)	(3000, 500, 3000)
Initial velocity (m/s)	(10, 0, 0)	–	–	–	–
Radius (m)	–	40	600	800	600

computation time of our algorithm is short enough for real-time application.

All the comparing data are listed in Table IV. It should be noted that the data in Table IV come from the average level of 10 runs. All of them run successfully except ACO, which fails once.

We extend our simulation to the constant dynamic environment where the target and obstacles move with constant velocity. Our simulations also show that the LP method can avoid moving obstacles autonomously without losing its target. Moreover, the vehicle is expected to select an optimal or near optimal velocity as the action.

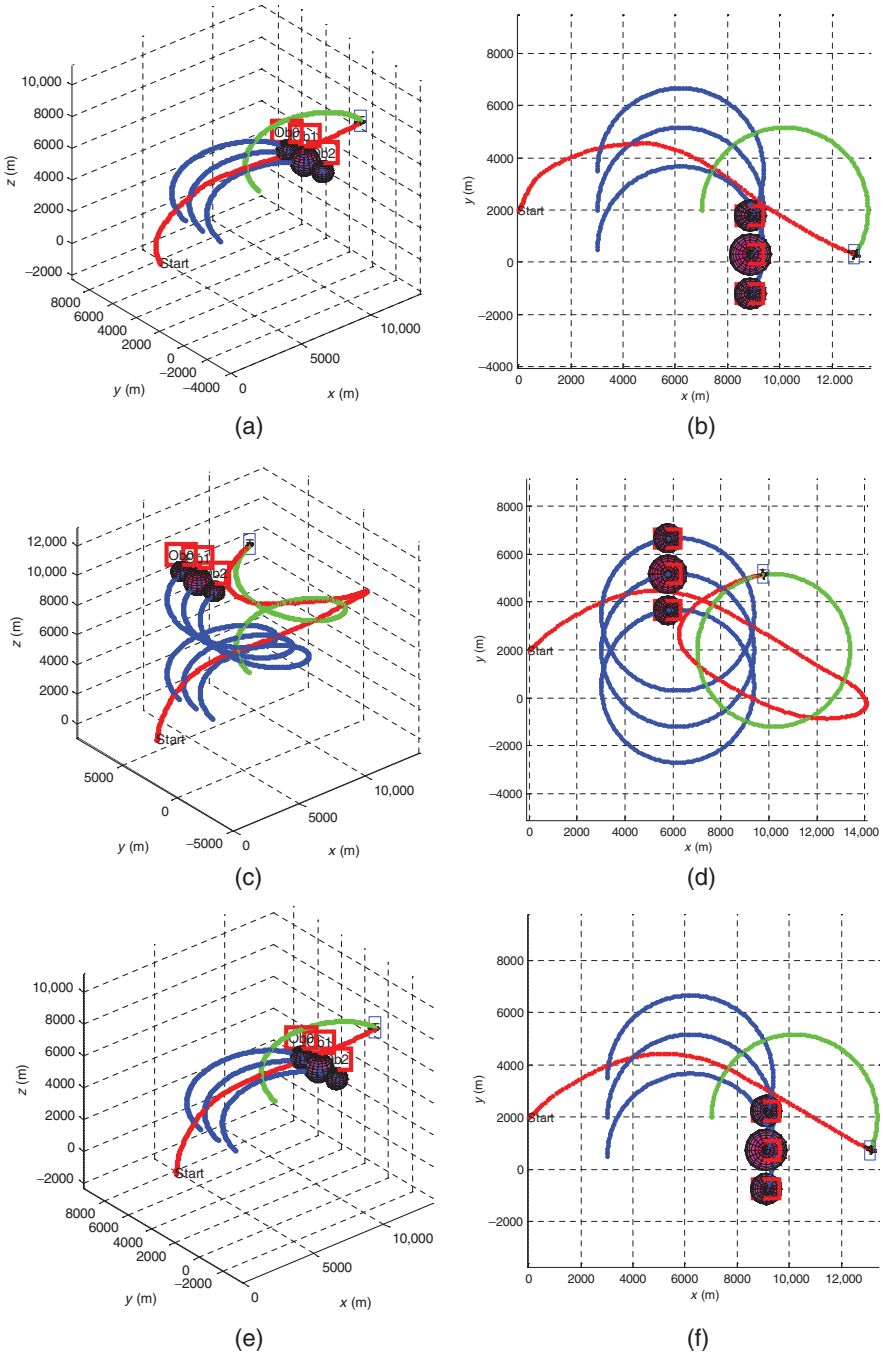


Fig. 7. (Colour online) Path planning in dynamic environment where the target and obstacles fly circular trajectories. (a) and (b): Path produced by LP and its vertical view; (c) and (d): Path produced by GA and its vertical view; (e) and (f): Path produced by ACO and its vertical view.

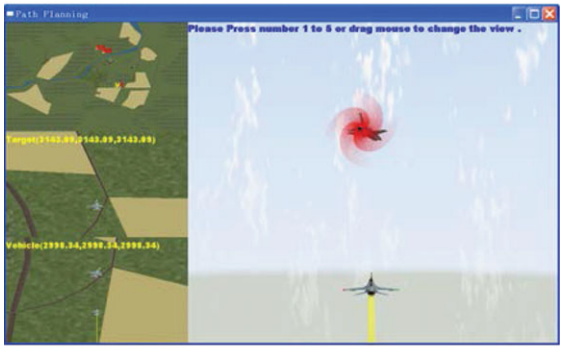
https://doi.org/10.1017/S0263574711000993 Published online by Cambridge University Press

Table IV. Comparing LP method with GA and ACO in static environment.

Critical parameters	LP $\omega = 0.5$ $M = 8$	GA Colony = 60 Generations = 40	ACO Colony = 50 Generations = 20 Elite ants = 20
Failed/run	0/10	0/10	1/10
Time consumed (s)	32.8	37.7	42.11
Trajectories length (m)	7542.87	8077.82	8611.74
Algorithm complexity (ms)	3.84	4.06	4.43



(a)



(b)

Fig. 8. (Colour online) Human-in-the-loop simulation. (a) A joystick is used to control the target flying; (b) the vehicle catching the target. There are three spatial obstacles in the simulation. As shown in the left-up window, which is a mini-map from the vertical view, red letters denote obstacles in the air.

5.2. Random dynamic environment

This time we deal with the challenging situation where all the movers are set to fly circular trajectories in three dimensions. The obstacles are so hazardous that it is very difficult for the vehicle to fulfill the task. The LP algorithm, as well as GA and ACO, is equipped on the vehicle to pursue an uncertain target that is also flying with circular trajectory. The path and its contrast with GA and ACO are shown in Fig. 7 and Table V, respectively.

From Table V, the LP method yields lower computed complexity than ACO, although it produces a slight more length. Compared to GA, the LP method shows distinct strengthen in total length and consumed time. This simulation shows that the vehicle evades dynamic obstacles and produces a sophisticated trajectory, which is a very smooth path and almost optimal to the moving target. During the whole process, the vehicle inherently realizes the threat of moving obstacles that may cause a potential collision. So, the path planning is optimized online.

Table V. Comparing LP method with GA and ACO in dynamic environment where target and objects move along circular trajectories.

	LP	GA	ACO
Failed/run	0/10	0/10	0/10
Time consumed (s)	59.5	123.5	57
Trajectories length (m)	15,887.25	31,528.48	15,548.08
Algorithm complexity (ms)	4.44	4.03	4.49

5.3. Human in the loop simulation with a joystick

In order to validate our algorithm, we take a human-in-the-loop simulation based on virtual reality software, Vega Prime 2.0,[20] where our LP method runs as a background library. A joystick is used to control the target moving in the space (see Fig. 8). We also place three static or dynamic obstacles in the air. The experiment results show that our algorithm enables to find a velocity-optimal path for pursuit task.

6. Discussion and Conclusions

An LP-based method is proposed for vehicle path planning in three dimensions. Two assignments, obstacle-avoiding and target-tracing, are linearized to generate an LP model in relative velocity space. We have considered the dynamics and kinematics of the vehicle in the model, as well as the limitation of sensor range. Several simulation examples are supplied in this paper and we can derive three conclusions from them. First, the algorithm is successful in 3D path planning, and it is easy to get the solution of the result model. Second, it is fast enough for real-time application. In most scenes it is superior to GA and ACO algorithms in the aspect of time complexity. Third, it has good performance on optimization regardless of static obstacles or moving obstacles environment, even the random moving environment.

The LP method is based on the mathematical programming and is one of the convex optimization technologies[19] that can guarantee the convergence and numerical accuracy. The future work will relax the linearization restriction and consider the quadric programming, which is another mature convex optimization technology and has higher accuracy. For the generic nature of mathematical programming, the

https://doi.org/10.1017/S0263574711000993 Published online by Cambridge University Press

local minimum exists only when the configuration and the relative parameters of all movers are extremely symmetrical. In terms of our experience, the case happens rarely in the real-world application. Even if encountered, we will remove it by introducing a little perturbation for the vehicle velocity in static environment. It remains an open problem to completely remove the local minima during path planning.

Acknowledgments

This work was partially supported by Natural Science Foundation of China (NSFC) under grant nos. 61035005 and 61075087, Hubei Provincial Education Department Foundation of China under grant no. Q20111105, and Hubei Provincial Science and Technology Department Foundation of China under grant no. 2010CDA005. Thanks are due to Qichuan Ding and Yulong Ma of Shenyang Institute of Automation for valuable discussion.

References

1. C. L. Shih, T. T. Lee and W. A. Gruver, "A unified approach for robot motion planning with moving polyhedral obstacles," *IEEE Trans. Syst. Man Cybern.* **20**(4), 903–915 (1990).
2. Y. Kitamura, T. Tanaka, F. Kishino and M. Yachida, "3-D path planning in a dynamic environment using an octree and an artificial potential field," In: *Proceeding of IEEE/RSJ International Conference on Intelligent Robots & Systems (IROS)*, (Oct. 11–15, 1995) pp. 474–481.
3. Y. Zhang and K. P. Valavanis, "A 3-D potential panel method for robot motion planning," *Robotica* **15**(4), 421–434 (1997).
4. S. S. Ge and Y. J. Cui, "New potential functions for mobile robot path planning," *IEEE Trans. Robot. Autom.* **16**(5), 615–620 (2000).
5. F. Fahimi, *Autonomous Robots Modeling, Path Planning, and Control* (Springer Science + Business Media, Dordrecht, Netherlands, 2009).
6. C. Mou, W. Qing-xian and J. Chang-sheng, "A modified ant optimization algorithm for path planning of UCAV," *Appl. Soft Comput.* **8**(4), 1712–1718 (2008).
7. Y. Hao, W. Zu and Y. Zhao, "Real-time obstacle avoidance method based on polar coordination particle swarm optimization in dynamic environment," In: *2nd IEEE Conference on Industrial Electronics and Applications (ICIEA 2007)*, Harbin, China (May 23–25, 2007) pp. 1612–1617.
8. J. Foo, J. Knutzon, J. Oliver and E. Winer, "Three-dimensional path planning of unmanned aerial vehicles using particle swarm optimization," *Proceedings of the 11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference (AIAA 2006–6995)* (Sep. 6–8, 2006).
9. S. Mittal and K. Deb, "Three-dimensional offline path planning for UAVs using multiobjective evolutionary algorithms," In: *Proceedings of the IEEE Congress on Evolutionary Computation*, Singapore (Sep. 25–28, 2007) pp. 3195–3202.
10. L. Zhao and V. R. Murthy, "Optimal flight path planner for an unmanned helicopter by evolutionary algorithms," *AIAA Guidance, Navigation and Control Conference and Exhibit, AIAA 2007–6741*, South Carolina, USA (Aug. 20–23, 2007).
11. F. C. J. Allaire, M. Tarbouchi, G. Labonté and G. Fusina, "FPGA implementation of genetic algorithm for UAV real-time path planning," *J. Intell. Robot. Syst.* **54**, 495–510 (2009).
12. T. Krenzke, "Ant Colony Optimization for Agile Motion Planning," *Master's Thesis* (MIT, Cambridge, MA, 2006).
13. P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *Int. J. Robot. Res.* **17**(7), 760–772 (1998).
14. Y. Wang, D. M. Lane and G. J. Falconer, "Two novel approaches for unmanned underwater vehicle path planning: constrained optimisation and semi-infinite constrained optimization," *Robotica* **18**, 123–142 (2000).
15. D. Zu, J. Han and D. Tan, "Acceleration space LP for the path planning of dynamic target pursuit and obstacle avoidance," In: *Proceedings of the 6th World Congress on Intelligent Control and Automation*, Dalian, China (Jun. 21–23, 2006) pp. 9084–9088.
16. A. Richards and J. P. How, "Aircraft trajectory planning with collision avoidance using mixed integer linear programming," In: *Proceedings of the American Control Conference*, Alaska, USA (May 8–10, 2002) pp. 1936–1941.
17. T. Schouwenaars, J. How and E. Feron, "Receding horizon path planning with implicit safety guarantees," In: *Proceedings of the American Control Conference*, Boston, MA, USA (Jun 30–Jul 2, 2004) pp. 5576–5581.
18. D. Applegate, W. Cook, S. Dash and M. Mevenkamp, QSopt Callable Library [online]. Available at: http://www2.isye.gatech.edu/∼wcook/qsopt/ Accessed May 4, 2011.
19. S. Boyd and L. Vandenberghe, *Convex Optimization* (Cambridge University Press, Cambridge, UK, 2004).
20. MultiGen-Paradigm. *Vega Prime Getting Started Guide, Version 2.0* (MultiGen-Paradigm, Inc, San Jose, CA, 2005).

https://doi.org/10.1017/S0263574711000993 Published online by Cambridge University Press