J Intell Robot Syst (2011) 61:203–219
DOI 10.1007/s10846-010-9493-9

A Minimum Risk Approach for Path Planning of UAVs

Luca De Filippis · Giorgio Guglieri · Fulvia Quagliotti

Received: 1 February 2010 / Accepted: 1 September 2010 / Published online: 12 November 2010
© Springer Science+Business Media B.V. 2010

Abstract In the last years, the Flight Mechanics Research Group of Politecnico di Torino started a wide research activity, focused on exploration and implementation of path planning algorithms for commercial autopilots, typically adopted on unmanned vehicles. Different path planning approaches were implemented in a Matlab/Simulink based tool, generating waypoints sequences with four methods: geometric predefined trajectories, manual waypoints definition, automatic waypoints distribution (i.e. optimizing camera payload capabilities) and, finally, a comprehensive A*-based approach. The tool was also integrated with functions managing the maps used for planning. In this paper, two approaches to path planning in presence of orographic obstacles are detailed. The first algorithm is subdivided in three phases: the generation of a risk map associated with the ground orography, the transformation of the map in a digraph analyzed with the A* algorithm (to obtain the path with minimum risk/minimum distance) and finally the smoothing phase, to obtain a flyable waypoint sequence, realized with the Dubins curves. The structure of this method was defined and implemented, but its optimization is still in progress. The second algorithm is based on the same risk map, but optimizing polynomial curves with a genetic algorithm. This method produces a flyable waypoints sequence, minimizing a cost function reflecting path length and collision risk. An extension of this path solver, including aircraft performance constraints, was also considered.

L. De Filippis (✉) · G. Guglieri · F. Quagliotti
Dipartimento di Ingegneria Aeronautica e Spaziale, Politecnico di Torino,
Corso Duca Degli Abruzzi 24, 10129 Torino, Italy
e-mail: luca.defilippis@polito.it
URL: www.polito.it

G. Guglieri
e-mail: giorgio.guglieri@polito.it
URL: www.polito.it

F. Quagliotti
e-mail: fulvia.quagliotti@polito.it
URL: www.polito.it

This method was tested on sample domains and its computational cost has still to be evaluated before the implementation in the tool.

Keywords Path planning · Genetic algorithms · Risk map · A* algorithm · Dubins curves

List of symbols		
a	Coefficient (cubic approximation)	–
b	Coefficient (cubic approximation)	–
c	Coefficient (cubic approximation)	–
C_{D0}	Drag coefficient (zero lift drag)	–
c_i	Shape coefficients (fitness function)	–
d	Coefficient (cubic approximation)	–
d_f	Deficiency function (penalty for fitness function)	–
E	Moving cost	–
f	Fitness function	–
f_i	Auxiliary function (fitness function)	–
F	Cost function	–
G	Moving cost	
g	Acceleration of gravity	m/s^2
h	Nominal flight height (above ground level)	m
H	Moving cost	–
H	Critical flight altitude (above sea level)	m
h_0	Airfield altitude (above sea level)	m
i	Index	–
k	Drag coefficient (induced drag)	–
m	Mass	kg
N	Number of discretizations (between two waypoints)	–
n	Load factor	–
P_i	Position (x,y)	–
r	Turning radius	m
R	Minimum turning radius	m
s	Distance (vectorial)	m
S	Wing surface	m^2
t	Time	s
T	Thrust	N
t_p	Distance (curvilinear)	m
u	Control	–
V	Airspeed	m/s
x	Coordinate	m
y	Coordinate	m
z_i	Obstacle elevation (above sea level)	m
α	Gain factor	–
β	Gain factor	–
γ	Gain factor	–
Δh	Altitude uncertainty	m
Δh_m	Altitude correction (local slope)	m
Δh_{safe}	Altitude penalty	m
θ	Orientation angle	rad
ρ	Air density	kg/m^3
ϕ	Roll angle	rad
ψ	Yaw angle	rad

1 Introduction

Unmanned Aircraft Systems (UAS) represent one of the most interesting technologies in aeronautics, currently in great growth and under development. Thanks to the big effort spent by research centers and aeronautical industries, they are becoming ever more autonomous and mission effective. On this route, an important role has been played by path planning algorithms, which represent a fundamental technology, inside the flight segment, for high autonomy operations. These algorithms were investigated from the early phases of the unmanned platforms development, in order to define the waypoints sequences managed by the autopilots. Starting from simple geometric based algorithms, used to generate directly these sequences on the map, with a predefined geometry or a point-and-click approach, more powerful algorithms have been developed. New families of solutions permit to associate sensor systems, control logics and platform dynamics to generate a real time flight path for obstacles and collision avoidance and even to integrate optimization constraints inside the algorithm itself [1].

The first path planning method presented in this paper belongs to the family of algorithms based on probabilistic approaches and risk maps. These methods solve the path planning problems (i.e. the generation of an optimized path considering static obstacles only) on static domains, but also dynamic domains can be considered assuming that adequate sensor systems are available on board during the flight. Presently, an algorithm was selected to optimize the path through the minimization of the collision risk with static obstacles distributed on a known map. Hence, the flight over domains containing moving objects (obstacle/collision avoidance) will be the subject of further development.

The risk-distribution approach needs to define a probability function, obtained considering different parameters: the conditions of flight, the environment characteristics, but also other particular mission requirements. After the probability function is defined, two different solutions can be followed:

- To generate a risk map decomposing the domain with a mesh, in order to discretize the risk distribution and associate a punctual risk value to a single cell of the mesh,
- To define a probability density function evaluated step-by-step along the path.

This algorithm implements the first option as a result of elaboration of Digital Elevation Model (DEM) maps of the flight area. At the end of the meshing process, a new map is obtained, transforming the risk map into a digraph. Nodes and arcs characterize the digraph, each one associated with a particular cost parameter connected with the probability function and representing a schematization of the domain.

Once the digraph is obtained, a minimum path strategy must be used to generate the optimal trajectory. The techniques used to solve this search process come from computer science, as they may represent a model for data exchange in networks. This family of algorithms is commonly defined "greedy" as they generate a local optimal solution with low computational cost. This feature, together with their straightforward implementation, strongly addressed their use in path planning problems. Well known examples are Djikstra and Bellman-Ford algorithms, that evolved into A*. The first two approaches differ in some aspects related to the arc weights definition, but, nevertheless, they are very similar in their implementation philosophy.

The A* solver represents one of the most widely used algorithms for path generation, largely applied to robotic science problems, space exploration and videogames. This algorithm combines characteristics of Djikstra and Bellman-Ford solvers but, more specifically, it was developed to analyze more effectively the domain in order to avoid distributed obstacles. In presence of passive threats such as canyons and traps, A* associates a heuristic cost function combined with an improved scan of the domain, enhancing the solution process more efficiently than Djikstra and Bellman-Ford methods can do.

After the path between the nodes of the digraph is obtained, it has to be smoothed in order to be adapted to vehicle's flight performances. Indeed, the A* algorithm provides a path connecting the centers of the cells within the mesh without considering edges or kinks that can't be followed by any type of aerial vehicle. In order to obtain a more feasible and realistic path, refinement algorithms have to be used. The kind of smoothing can be very different in nature and various solutions can be found in literature [2] such as simple spline curves used to connect the edges of the path or slightly more complex methods like the Dubins curves [3].

Dubins curves are widely used to solve path optimization problems in robotics, due to their conceptual simplicity. This method connects a predefined waypoints sequence smoothing the path. Defining a starting and an ending point with assigned velocity vectors, the assumptions for path generation are (i) constant curvature or turn radius and (ii) constant speed along the path that connects a pair of waypoints. Under the previous constraints the optimal solutions are reduced to a group of curves, named as R-geodesic curves. These curves modify the nominal waypoints position, according to vehicle's characteristics (maximum bank angle and cruise flight speed). Hence, this solution is chosen to redistribute the waypoints position, obtained with the A*-based search process, minimizing the cross-track and deviation errors over the prescribed path due to aircraft performance limitations. The Dubins optimization approach is a good compromise between spline-based methods and more complex solutions involving vehicle dynamics, such as Model Predictive Control (MPC) and Receding Horizon Control (RHC) [4]. These time-marching algorithms generate step after step a control input that respects the dynamics of the vehicle. Then, integrating sensors information and vehicle states, new predictions can be generated over the successive time horizon. An accurate path estimate is obtained also in presence of moving obstacles and harsh environments, but numerical convergence may be a concern in the application of these methods.

The second proposed method elaborates the same risk map, but it is based on a genetic optimizer, searching the minimum risk path under defined constraints. Other optimization algorithms could be used to generate the same minimum risk paths. Indirect methods were the first solutions applied to differential equations optimization problems. Algorithms based on Lagrange multipliers were widely used to reduce optimization to a boundary condition problem. Sequential Gradient Restoration Algorithm (SGRA) represents an example of application of indirect methods to space trajectory optimization. The limit of these algorithms lies in their inability to solve nonlinear problems over complex discontinuous domains and in their intrinsic attraction for local minimums for the cost function. Differently, genetic algorithms (GA) are very attractive for overcoming both these aspects. Genetic algorithms provide a model of the domain as a population and search the optimal individuals, modeling the evolution of the population according to the genetic rules for matching the chromosomes of children with parents. The slow convergence of the search process typical of genetic drivers may be accelerated with micro-genetic algorithms (micro-GA) occasionally used in physics and engineering. This option significantly reduces the number of function evaluations with a faster convergence to the near-optimal region. Very briefly, a micro-GA starts with a random, very small population. The population evolves in normal GA fashion and converges in few generations. Then, a new random population is chosen while keeping the best individual from the previously converged generation and the evolution process restarts. Note that average population fitness values are not meaningful with a micro-GA because of the start-restart nature of the micro-GA evolution process. Details are provided in [5, 6]. Another limitation of conventional GA is that they do not guarantee the convergence to an optimal solution within a finite amount of time. This point is assessed for micro-GA by tracking the fitness function of the best individual and interrupting the search process just when an adequate performance of the individual is reached, above an expected minimum.

2 The Risk Function and the A* Based Approach

Studies and publications, concerning the probabilistic approach for path planning, confirm that the definition of the risk function (RF) is a major issue [2, 8]. D'Andrea and Jun [2] presented an algorithm for the generation of paths in adverse environments, suggesting a complex RF based on radar interception-field and enemy threat capabilities. Bertucelli and How [7] carried out studies on the definition of probability functions in unknown environments giving important cues for this issue. In all these cases the RF is defined on a probabilistic basis, due to uncertainties on the risk for the vehicle. Conditional probability is a fundamental aspect in the construction of the function, considering the mutual influence of the various parameters involved.

The aim is to correlate the RF to the oroghaphy considered as a possible risk for the vehicle (impact). For a bi-dimensional path plan, the flight altitude can be considered constant and risk distribution can be defined so that the flight over the obstacles is always safe. In this way, conditional probability is not involved in the statement of the problem.

The equation describing the critical flight altitude H is used to define the RF (see Fig. 1):

$$H = h_0 + h - \Delta h - \Delta h_{safe}$$

where:

h_0	airfield altitude (above sea level)
h	flight height (above ground level)
Δh	altitude uncertainty (autopilot/sensor accuracy, controlled aircraft dynamics)
Δh_{safe}	altitude penalty (margin for safe flight over the obstacle).

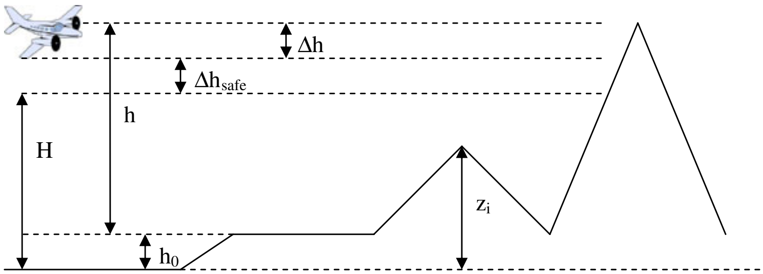


Fig. 1 The altitude decomposition

The uncertainty on the vehicle's altitude is due to different factors: sensing errors associated with the onboard sensors system, uncertainties of the autopilot altitude following system and atmospheric conditions. These three elements represent the modeling basis of the uncertainty, but other components will be included later.

Once the altitude H is defined and introducing the variable z_i, representing the local elevation (orography), the risk function is obtained as:

$$\begin{cases} RF_i = 0 \rightarrow z_i \leq H \\ RF_i = 1 \rightarrow z_i \geq H \end{cases}$$

The term RF_i represents the local probability for the vehicle to fly over a point on the map in unsafe conditions. As already shown, first implementation of the algorithm exploits a step shaped risk distribution, comparing punctual elevation with the modeled critical flight altitude H. Starting from this assumption, the model is improved assigning $RF_i=0.5$ to the cells on the grid neighbouring those with $RF_i=1$. In this way, it is possible to reduce the risk connected with flight close to obstacles. Alternatively, the ground slope may be assumed to change so that the RF varies continuously between 0 and 1. Introducing the parameter Δh_m as the influence of ground slope variation, the RF becomes:

$$\begin{cases} RF_i = 0 \rightarrow z_i < H - \Delta h_m \\ RF_i = \dfrac{z_i - (H - \Delta h_m)}{\Delta h_m} \rightarrow H \leq z_i \leq H - \Delta h_m \\ RF_i = 1 \rightarrow z_i > H \end{cases}$$

For the above case, the risk varies proportionally to the elevation variation along the ridges of the hills. This type of function permits to evaluate the risk of proximity flight near to the obstacle, addressing the search process towards the design of safe paths.

Once the RF is defined, the Risk Distribution Map (RDM) can be created calculating the risk on each cell of the map. The definition of the RDM was widely discussed in previous work by several authors [3, 8]. Delaunay triangulations were one of the first implementing solutions. The Voronoi graph is another widely used method, particularly attractive for problems related with threat exposure risk such as radars or missiles [9]. The Voronoi graph is obtained placing threats on the map and tracing cells as if each threat was centered in the cell. The edges delimiting the cells are traced so as a vehicle follows the path composed with them, passing between two threats with equal distance and minimum risk. This feature makes the Voronoi graph not particularly attractive for the problem described in this paper. Furthermore, orography is spread over the domain and the risk distribution may cover wide areas of the map. Hence, the meshing work on the map is heavier than that required for Voronoi graphs or Delaunay triangulations, as clearly explained by D'Andrea and Jun [2] (an interesting solution defining a mesh with cells of constant shape is also presented).

Two different ways of meshing are hereafter presented. Note that, in order to correlate the risk distribution with orography, geotiff maps are exploited with DEMs of the same area. The coordinates on the maps (pixel) are associated with the corresponding DEM file to assign the given elevation. The nodes of the digraph are defined selecting the single image pixel or mediating over a cluster of pixel. Thus, the first method proposes calculation of the risk over each pixel of a geotiff map associated with a DEM. With this solution the problem of cells-shape definition is avoided and possible errors due to meshing along the edges is reduced. This solution generates a more precise mesh with growing of the map resolution. The drawback of this method is the high computational cost paid using high-resolution maps. Another important consideration, influencing the introduction of a second method, regards the kind of path to be planned. The target is the generation of a waypoints sequence normally managed by the autopilot's planning software. These flight schedulers need a minimum distance between successive waypoints to minimize error on the planned path. So, the second method suggests to mesh the map with square cells and to compute the risk over the cell as the average of each pixel-value inside the cell. This solution guarantees minimum distance between the waypoints sequence and reduces the computational time, being unnecessary to manage matrices of the same size of the map. The number of pixels to be included in a cell is defined considering the autopilot characteristics, the map resolution and the mission requirements.

Once the RDM is obtained, it can be processed as a digraph by a "greedy" algorithm for path generation. These algorithms generate the shortest path in a digraph together with the walking values of each node and analyzing all the possible paths from a node (defined as the start) to another one (defined as the target). Dijkstra developed one of the first implementations of these algorithms. The algorithm obtains the shortest path in a digraph with all positive walking values for the nodes. A further development of the Dijkstra solution is the Bellman-Ford algorithm that generates the shortest path in the same way, evaluating also negative walking values. D'Andrea and Jun [2] presented the capabilities of the Bellman-Ford algorithm used as a path planner. This last solution was applied to problems related with threats, with an implementation that doesn't seems to be addressed to path planning in general. To solve this last kind of problems, "greedy" algorithms evolved into the A* algorithm. This solver was developed for path planning of robots, generating the shortest path with a multi-parametric cost function. The major difference between Bellman-Ford and A* algorithms is the capability of avoiding traps along the path, exploiting a heuristic function constantly updated during the path search. This aspect oriented many users towards the choice of A* as a planning algorithm. Comparative tests between the different "greedy" algorithms go beyond the interests of the present research. Nevertheless, for clarity of presentation, the A* algorithm is briefly described in the next paragraphs.

The A* algorithm generates the optimum path (from a starting to an ending cell) evaluating the cost function for all the cells involved in the search process. The cost function F is normally represented by two elements G and H, where:

$$F = G + H$$

G moving cost from the current cell to a neighboring one,
H moving cost (estimated) from the current cell to the last cell (target).

These are the basic terms in the cost function, but other contributing elements can be added to refine the cost estimation. As an example:

$$F = G + H + E$$

where E is the moving cost connected with the elevation of the neighboring cell. Once the elements of the cost function are modeled their sensitivity must be weighted through specific gain factors addressing the path search according to mission requirements:

$$F = \alpha G + \beta H + \gamma E$$

The implementation keys of the algorithm are shortly summarized:

- **Cost function:** it is important to give the right weight to the parameters in the function, to obtain a path correctly influenced by these parameters.
- **Open list:** it is important to organize the open list in such a way to reduce the computational time.
- **Mesh:** it is important to evaluate the dimensions of the map, to enhance the speed of convergence to a solution for the search process.

The A* algorithm generates the minimum risk path for the vehicle; more strictly it gives the sequence of cells to go from the starting point to the target one with minimum risk. This sequence can be transformed into a path simply connecting the centers of the cells, but this may generate an inconsistent trajectory for the flight of an aerial vehicle. So, smoothing algorithms transform the path generated with A* in a flyable one. Various solutions were found among available references. Chandler used directly flight dynamic constraints to smooth the path [10]. Bortoff [11] developed a method similar to potential field algorithms, using virtual masses emanating a force field. Zabarankin [12] proposed to solve the optimization considering the trajectory constraints and the risk distribution together. These solutions are all very attractive for the smoothing of the optimized output generated with "greedy" algorithms. On the other hand, A* may generate a waypoints sequences constraining the path in such a way that may preclude an effective smoothing process. Furthermore, complex-smoothing solutions may have higher computational time. For these reasons the Dubins curves resulted the best choice that associates a straightforward optimization process to a fast smoothing algorithm [13–15].

The differential equations and the basic assumptions governing the Dubins vehicle are [16–20]:

- planar motion,
- constant speed: $V(t) = V$,
- turning motion bounded by minimum curvature radius: $R \leq r(t)$,

$$\frac{d}{dt}\begin{pmatrix} x \\ y \\ \theta \end{pmatrix} = \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \\ \frac{u}{R} \end{pmatrix} \; |u| \leq 1$$

where:

(x, y)	position vector in the plane,
θ	orientation angle with respect to the planar reference system,
u	control parameter.

The control parameter u is normalized with respect to the maximum curvature and results bounded between ±1, while speed V is normalized to 1.

Dubins analytically demonstrated the existence of the R-geodesic curves and proved that they consist of not more than three portions, being either a straight line or an arc [17]. As a result, this kind of *Dubins path* can be subdivided in two families, grouping six combinations of curves and straight lines:

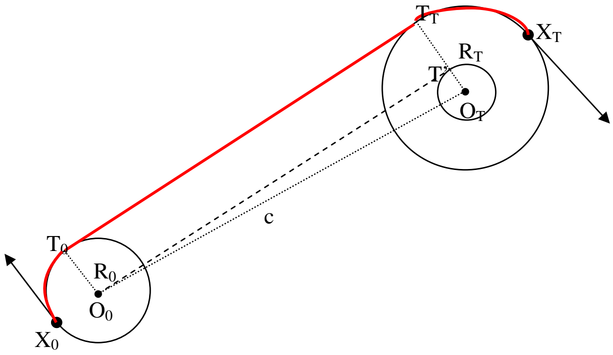
- CCC paths: LRL, RLR,
- CSC paths: LSL, RSR, LSR, RSL.

where:

S	straight lines,
C	curves,
R	clockwise rotations,
L	counter-clockwise rotations.

These groups of paths represent all the optimal trajectories for a *Dubins vehicle*. Therefore, a procedure for the re-construction of the R-geosedic curves can be

Fig. 2 The design of a CSC path



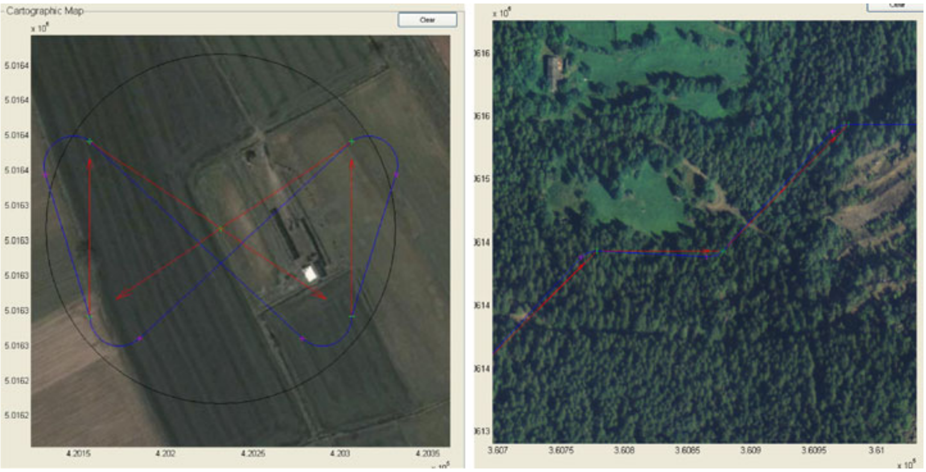


Fig. 3 Smoothing of the path

shortly outlined. Drawing two circles of radius R, tangent to the velocity vector in the initial position and other two for the velocity vector in the final position, can be selected a tangent segment to two circles among the four available. Then, two arcs on the selected circles and a straight line on the tangent segment will compose the optimal path. As an example, the geometrical construction of a CSC path is described in Fig. 2.

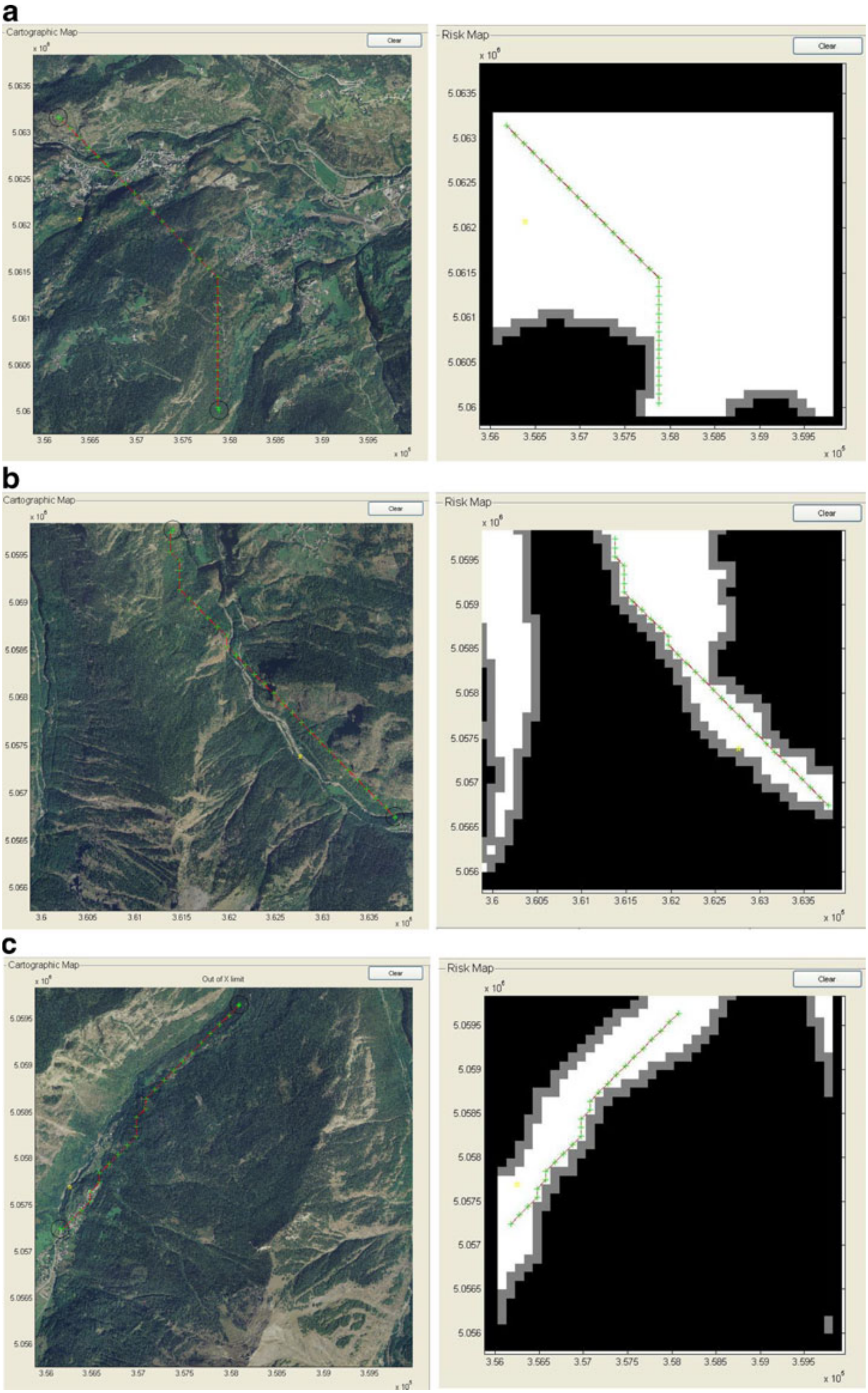
An example of Dubins curves smoothing is shown in Fig. 3, where it is applied to a butterfly-shaped path and to a segment of an A*-based minimum risk path. The results of the smoothing process are evident: the corners are re-designed to be compatible with aircraft performances (maximum bank angle and cruise flight speed i.e. minimum turning radius).

Figure 4 shows four examples of paths generated with the A* algorithm over an alpine area. The paths are locally smoothed using Dubins curves. A flight altitude of 1200 m is considered and the safety margins, Δh_{safe} and Δh, are fixed to 15 m. The path planner generates simple solutions on free zones (*a*), but also paths that follow the valleys and the canyons can be obtained (*b* and *c*) depending on the selected start/end waypoints. The last example (*d*) verifies the capability of overcoming obstacles along the path. These paths show the strong effect of the *H* component of the cost function (measuring distance from the target). Every path tends to the target along grey zones, where risk function is RF = 0.5, in order to minimize the travel distance. Future developments would point to set gain factors inside the cost function to obtain selective paths depending on mission requirements.

The above-mentioned path-planning procedure was integrated into a single software package (implemented in Matlab/Simulink). The application software

Fig. 4 **a** Examples of paths generated over alpine areas. **b** Examples of paths generated over alpine areas. **c** Examples of paths generated over alpine areas. **d** Examples of paths generated over alpine areas

▶


a
Cartographic Map
Clear
x 10^6
5.0635
5.063
5.0625
5.062
5.0615
5.061
5.0605
5.06
3.56
3.565
3.57
3.575
3.58
3.585
3.59
3.595
x 10^5
Risk Map
Clear
x 10^6
5.0635
5.063
5.0625
5.062
5.0615
5.061
5.0605
5.06
3.56
3.565
3.57
3.575
3.58
3.585
3.59
3.595
x 10^5
b
Cartographic Map
Clear
x 10^6
5.0595
5.059
5.0585
5.058
5.0575
5.057
5.0565
5.056
3.6
3.605
3.61
3.615
3.62
3.625
3.63
3.635
x 10^5
Risk Map
Clear
x 10^6
5.0595
5.059
5.0585
5.058
5.0575
5.057
5.0565
5.056
3.6
3.605
3.61
3.615
3.62
3.625
3.63
3.635
x 10^5
c
Cartographic Map
Out of X limit
Clear
x 10^6
5.0595
5.059
5.0585
5.058
5.0575
5.057
5.0565
5.056
3.56
3.565
3.57
3.575
3.58
3.585
3.59
3.595
x 10^5
Risk Map
Clear
x 10^6
5.0595
5.059
5.0585
5.058
5.0575
5.057
5.0565
5.056
3.56
3.565
3.57
3.575
3.58
3.585
3.59
3.595
x 10^5

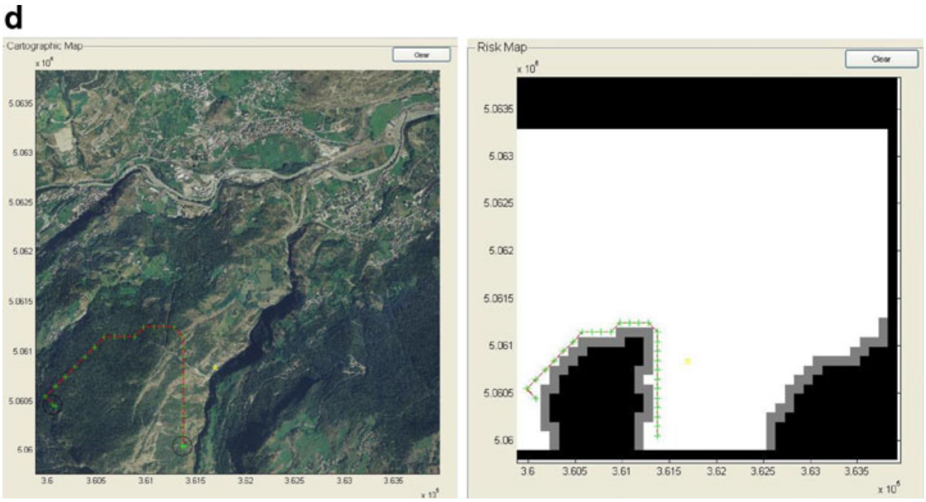

d
Cartographic Map
Clear
x 10^6
5.0635
5.063
5.0625
5.062
5.0615
5.061
5.0605
5.06
3.6
3.605
3.61
3.615
3.62
3.625
3.63
3.635
x 10^5
Risk Map
Clear

Fig. 4 (continued)

(see Fig. 5) handles geotiff and DEM maps providing an analysis of the selected flight area in terms of risk function. Then, a sequence of waypoints is generated, being compatible with the programming scripts for Micropilot commercial autopilots. The tool has a basic Graphical User Interface (GUI) used to manage the map and the path planning sequence.

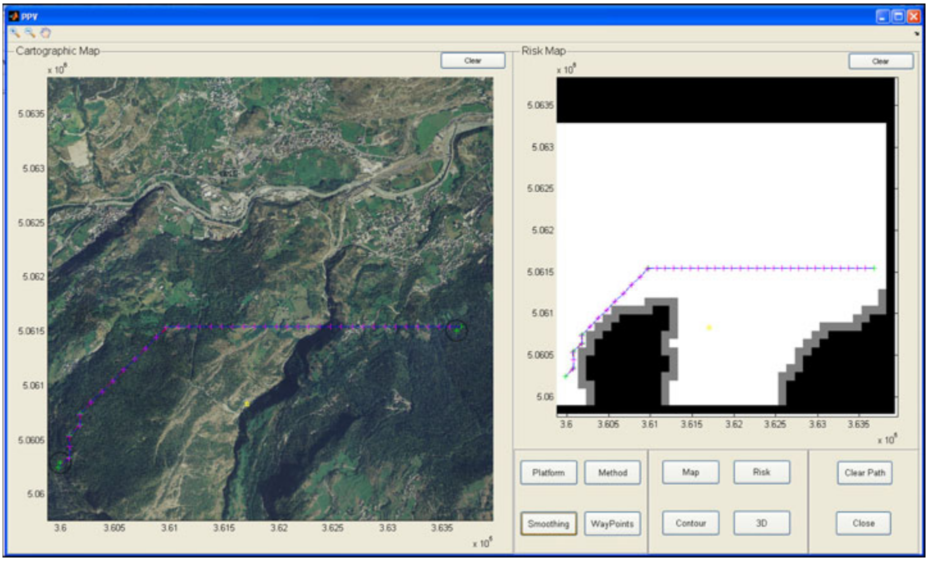

ppy
Cartographic Map
Clear
x 10^6
5.0635
5.063
5.0625
5.062
5.0615
5.061
5.0605
5.06
3.6
3.605
3.61
3.615
3.62
3.625
3.63
3.635
x 10^5
Risk Map
Clear
Platform
Method
Map
Risk
Clear Path
Smoothing
WayPoints
Contour
3D
Close

Fig. 5 The graphical user interface

Four options can be selected by the user to generate the waypoints sequences:

- point and click mode,
- predefined path shapes (square, rectangular and butterfly shapes),
- automatic grid type waypoints generation (grid patterns for photogrammetric use),
- A*-based minimum risk/minimum distance paths.

The waypoints sequences obtained are smoothed with Dubins curves, taking into account the UAVs' turn performances and average flight speed. For grid type patterns, the path generation is optimized for optical type payloads, specifying image overlaps and focal length. The package also allows the manipulation of maps and flight paths (i.e. sizing, scaling and rotation of mapped patterns). 3D surface and contour-level plots are available for enhancing the visualization of the flight path. Coordinates and map formats can also be converted in different standards according to user specification.

3 The Minimum Risk Path Generation with a Genetic Algorithm

The minimum risk paths, defined using the A* algorithm, need to be refined with some type of smoothing methods in order to obtain solutions compatible with real flight. The smoothing process should modify the path without affecting the optimality of the solution obtained with the "greedy" algorithm. The resolution of the domain may compromise the compatibility between optimal and smoothed (i.e. sub-optimal) solutions that may become so altered to cancel out the benefits of the optimization process. This is particularly true when the vehicle should fly inside a small confined area. Therefore, a different path generation process is required. The concept is the introduction of a path-planning algorithm based on a genetic optimizer of a polynomial trajectory under constraints associated with the risk distribution [18, 19]. The implementation is based on a micro-genetic solver [5, 6] that is able to overcome (at least partially) the typical limitations in terms of computational time for this family of evolutionary algorithms.

Using the same previously described risk function to define the domain, a vehicle is supposed to fly at constant altitude (the extension to variable altitude flight would require local reassignment of the risk function) over a rectangular area from point $P_1(x_1, y_1)$ to $P_2(x_2, y_2)$. The obstacles in the domain are static (i.e. mountains, hills, buildings, ...) and they become a threat for the flying vehicle as the trajectory intersects the contour of the obstacle. The trajectory was approximated with a cubic form $y(t) = a\,x(t)^3 + b\;x(t)^2 + c\,x(t) + d$ passing through the start and end points, covering the distance along the curved path t_P different from the vector $s = P_1 - -P_2$. Other functional approximations were considered but the cubic form was found to be adequate for the application without compromising in terms of complexity the numerical solution.

The optimal path was found using a random search algorithm (the genetic driver is described in [6]). The fitness function f to be maximized is given in the following form:

$$\max f = \frac{1}{(f_1 - c_1)^2 + (f_1 - c_2)^2} + \frac{1}{(f_2 - c_3)^2 + (f_2 - c_4)^2} - d_f(x, y)$$

where:

$$f_1 = \frac{1}{N} \cdot \sum_{i=1}^{N} RF\left(x_i, y_i\right) \qquad f_2 = \frac{t_P - s}{s}$$

$$s = \sqrt{\left(x_2 - x_1\right)^2 + \left(y_2 - y_1\right)^2} \qquad t_P = \sum_{i=1}^{N} \sqrt{\Delta x_i^2 + \Delta y_i^2}$$

A deficiency function $d_f(x, y) = 10^6$ is also included to penalize trajectories crossing the outer bounds of the map. Results were obtained for a reference test risk function (Fig. 6). The solution is compared with the output of the A* solver for the same risk domain. The trajectory obtained with the genetic solver respects the following constraints:

- the trajectory cannot cross the map limits (i.e. stay within the outer bounds)
- the average risk over the flight path must be limited ($c_1 = 0$ and $c_2 = 0.25$)
- the trajectory must be as straight as possible ($c_3 = 0$ and $c_4 = 10$).

The comparison between the different solvers (see Fig. 7) confirms that the two approaches are not equivalent when applied to environments that are cluttered with obstacles. As a matter of fact, the A* solution is safer as it may generate path shapes of any form with lower risk exposure. Nevertheless, these paths must be redesigned before flight (see Fig. 7). Differently, the genetic solver is based on polynomial trajectories that are a compromise between an acceptable level of risk and path

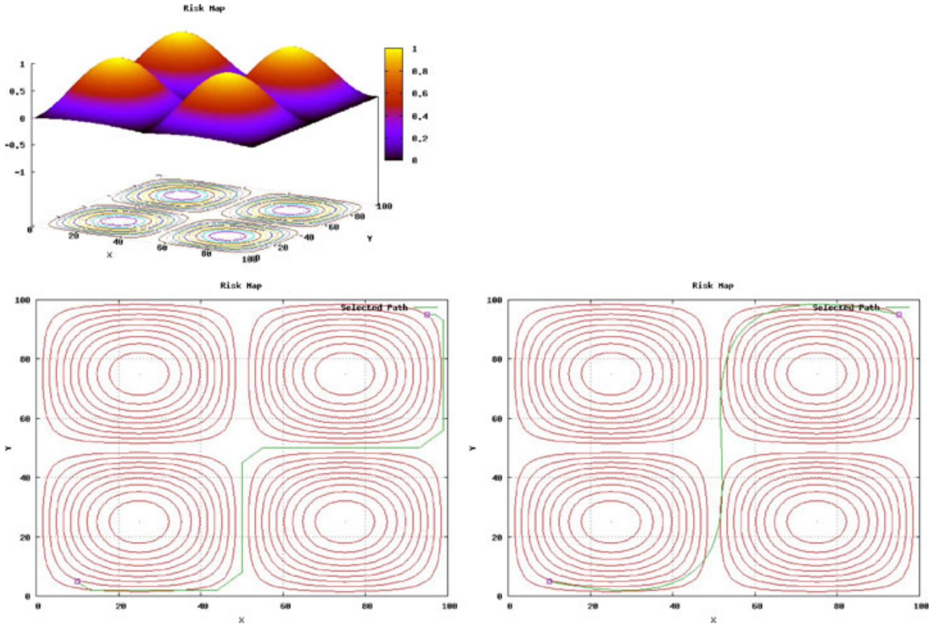


Fig. 6 An example of risk environment: flight path comparison between the A* algorithm (*left*) and the genetic algorithm (*right*)

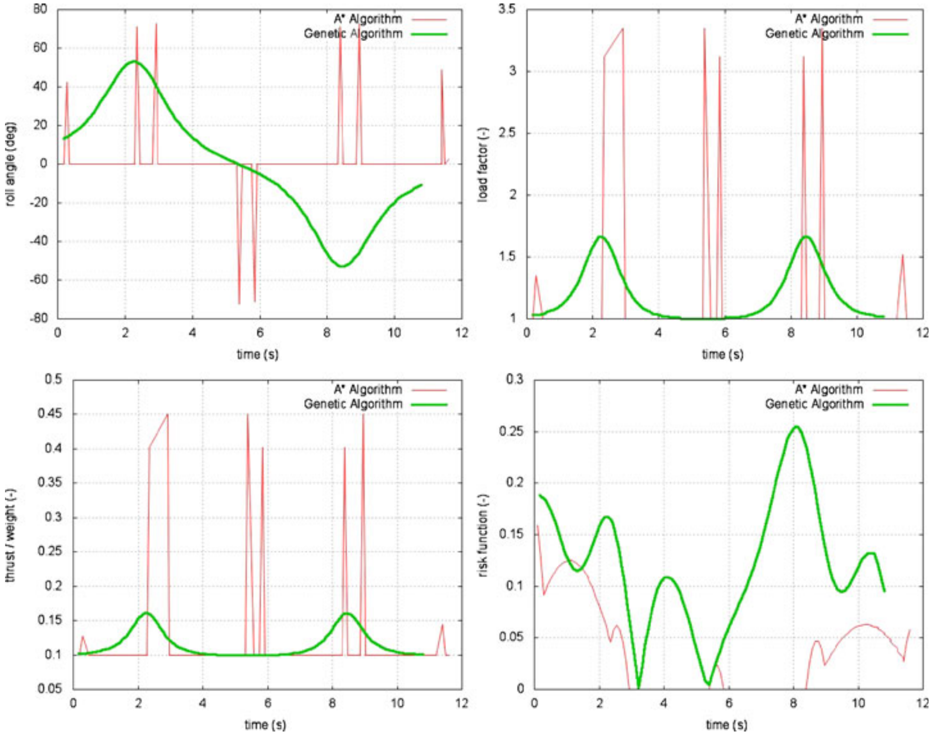


Fig. 7 An example of in-flight performance comparison

smoothness. This last type of solution is generally closer to real controlled fixed wing aircraft trajectory.

As an example, the solution found with the genetic solver was also used to verify the performances of a reference test vehicle (Mavtech MH-600 mini UAV flying at sea level with $V = 15$ m/s and a mass $m = 0.6$ kg). The flight area was scaled to 100 m × 100 m. The conventional equations for aircraft dynamics given in trajectory axes [6] are used to obtain the flight parameters presented in Fig. 7: the heading angle ψ, the roll angle ϕ, the load factor n and the propulsive thrust T.

The equations are hereafter shortly summarized (the formulation is simplified assuming that the airspeed V and the altitude h are constant along the path):

$$\psi(t) = \arctan(dy/dx) \qquad \tan\phi(t) = \frac{V^2}{g}\cdot\frac{d^2y/dx^2}{1+(dy/dx)^2}\cdot\cos\psi(t)$$

$$n(t) = \frac{V^2}{g}\cdot\frac{d^2y/dx^2}{1+(dy/dx)^2}\cdot\frac{\cos\psi(t)}{\sin\phi(t)} \quad T(t) = \frac{1}{2}\rho V^2\cdot S\cdot\left[C_{D0}+K\cdot\left(\frac{n(t)\cdot m\cdot g}{\frac{1}{2}\rho V^2 S}\right)^2\right]$$

Converting the trajectory path into aircraft parameters allows the check for compatibility with the real aircraft flight (the flight parameters should be limited according to the levels of maximum roll angle, load factor and thrust). The same flight variables could be actually included into the fitness function for the search process implemented in the genetic solver, driving the solution towards more feasible trajectories. This last approach was widely detailed in [6] where the same concept was used for waypoints reassignment in terms of minimum energy/fuel consumption.

Nevertheless more complex fitness functions also impact the solution in terms of time for convergence, slowing down the search process, that was found to be 1 or 2 orders of magnitude larger (depending on grid size and cell refinement) if compared with the A* solver. This last limitation may be only partially overcome by decreasing the number of generations accepted for convergence to solution (i.e. relaxing the convergence criteria). Further work is still required before assessing the applicability of this approach to real flight cases.

4 Conclusions and Future Developments

Different path planning approaches were implemented in a software library, generating waypoints sequences with four methods: geometric predefined trajectories, manual waypoints definition, automatic waypoints distribution (i.e. optimizing camera payload capabilities) and, finally, a comprehensive A*-based approach. The tool was also integrated with functions managing the maps used for planning.

Some open issues still remain for the A*-based implementation:

- the risk function used to evaluate the risk over the map should be tested on different environments in order to investigate its modeling capabilities,
- the gains, weighting the risk function, have to be selected in order to obtain differently oriented paths (selective path definition),
- the implementation of the meshing process should be reconsidered as it strongly influences the computational time and the output of the path generation.

An alternative path optimizer, based on a genetic algorithm, was described. This implementation is attractive for its potential ability to include vehicle's dynamics in the optimization process. Even if the results are promising further work is still required before assessing the applicability to real flight cases.

Successive studies will also consider the integration of the path planning algorithms with onboard control and sensors systems simulators. The purpose is the implementation of Model Predictive Control based solutions for in-flight collision avoidance.

Acknowledgements This research work is part of the project SMAT-F1 (Sistema per il Monitoraggio Avanzato del Territorio—Fase 1) funded by Regione Piemonte (Italy).

References

1. De Filippis, L., Guglieri, G., Quagliotti, F.: Flight Analysis and Design for Mini-UAVs. XX AIDAA Congress, Milano, Italy (2009)
2. Jun, M., D'R.: Path Planning for Unmanned Aerial Vehicles in Uncertain and Adversarial Environments, Models, Applications and Algorithms. Kluwer Academic Press (2002)
3. Kingston, D.K.: Implementation Issues of real-time trajectory generation on small UAVs. MS Dissertation, Brigham Young University, USA (2004)
4. Gu, D.W., Postlethwaite, I., Kim, Y.: A comprehensive study on flight path selection algorithms. IEE Seminar on Target Tracking: Algorithms and Applications. Birmingham, UK (2006)
5. Carroll, D.L.: Chemical laser modeling with genetic algorithms. AIAA J. **34**(2), 338–346 (1996)
6. Guglieri, G., Quagliotti, F., Speciale, G.: Optimal trajectory tracking for an autonomous Uav. Autom. Control Aerosp. **1**(1) (2008)

7. Bertuccelli, L.F., How, J.P.: Robust UAV search for environmentas with imprecise probability maps. IEEE Conference of Decision and Control. Seville, Spain (2005)
8. Pfeiffer, B., Batta, R., Klamroth, K., Nagi, R.: Path planning for UAVs in the presence of threat zones using probabilistic modelling. Handbook of Military Industrial Engineering. Taylor and Francis, USA (2008)
9. Erwig, M.: The graph Voronoi diagram with applications. Networks. **36**(3) (2000)
10. Chandler, P.R., Rasmussen, S., Patcher, M.: UAV cooperative path planning. AIAA Guidance, Navigation and Control Conference. Denver (2000)
11. Bortoff, S.A.: Path planning for UAVs. Proc. of the American Control Conference. Chicago (2000)
12. Zabarankin, M., Uryasev, S., Pardalos, P.: Optimal risk path algorithm. Technical Report 2001-4, Department of Industrial and Systems Engineering. University of Florida, USA (2001)
13. Boissonnat, J.D., Cèrèzo, A., Leblond, J.: Shortest paths of bounded curvature in the plane. J. Intell. Robot. Syst. **11**(1–2) (1994)
14. Reeds, J.A, Shepp, L.A.: Optimal path for a car that goes both forwards and backwards. Pac. J. Math. **145**(2) (1990)
15. Sussmann, H.J., Tang, W.: Shortest Paths for the Reeds-Shepp Car: A Worked Out Example of the Use of Geometric Technique in Nonlinear Optimal Control, Report SYCON-91-10. Rutgers University (1991)
16. Ma, X., Castanon, D.A.: Receding horizon planning for dubins traveling salesman problems. IEEE Conference on Decision and Control. San Diego (2006)
17. Dubins, L.E.: On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal positions and tangents. Am. J. Math. **79** (1957)
18. Capozzi, B.J.: Evolution-based Path planning and management for autonomous UAVs. Ph.D. Dissertation, University of Washington, USA (2001)
19. Nikolos, I.K., Tsourveloudis, N.C., Valavanis, K.P.: Evolutionary algorithm based 3-D path planner for UAV navigation. IEEE Trans. Syst. Man Cybern., Part B, Cybern. **33**(6) (2003)
20. Chitsaz, H., LaValle, S.M.: Time-optimal paths for a dubins airplane. IEEE Conference on Decision and Control. New Orleans, USA (2007)

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.