Mathematical Models for Aircraft Trajectory Design: A Survey

D. Delahaye, S. Puechmorel, P. Tsiotras, and E. Feron

Abstract Air traffic management ensures the safety of flight by optimizing flows and maintaining separation between aircraft. After giving some definitions, some typical feature of aircraft trajectories are presented. Trajectories are objects belonging to spaces with infinite dimensions. The naive way to address such problem is to sample trajectories at some regular points and to create a big vector of positions (and or speeds). In order to manipulate such objects with algorithms, one must reduce the dimension of the search space by using more efficient representations. Some dimension reduction tricks are then presented for which advantages and drawbacks are presented. Then, front propagation approaches are introduced with a focus on Fast Marching Algorithms and Ordered upwind algorithms. An example of application of such algorithm to a real instance of air traffic control problem is also given. When aircraft dynamics have to be included in the model, optimal control approaches are really efficient. We present also some application to aircraft trajectory design. Finally, we introduce some path planning techniques via natural language processing and mathematical programming.

Keywords Aircraft trajectory design • B-spline • Principal component analysis • Bézier • Homotopy • Optimal control • Air traffic management • Strategic planning • Pre-tactical planning • Tactical planning

D. Delahaye (✉) • S. Puechmorel
Applied Mathematics Laboratory, French Civil Aviation University, Toulouse, France
e-mail: delahaye@recherche.enac.fr; puechmor@recherche.enac.fr

P. Tsiotras • E. Feron
School of Aerospace Engineering Georgia Institute of Technology, Atlanta, GA, USA
e-mail: tsiotras@gatech.edu; feron@gatech.edu

Electronic Navigation Research Institute (ENRI), *Air Traffic Management and Systems*, Lecture Notes in Electrical Engineering 290, DOI 10.1007/978-4-431-54475-3_12,

1 Introduction

Aircraft trajectory is one of the most fundamental objects within the frame of ATM. However, partly due to the fact that aircraft positions are most of the time represented as radar plots, the time dependence is generally overlooked so that many trajectory statistics conducted in ATM are spatial only. Even in the most favorable setting, with time explicitly taken into account, trajectory data is expressed as an ordered list of plots labeled with a time stamp, forgetting the underlying aircraft dynamics. Furthermore, the collection of radar plots describing the same trajectory can have tenths more samples, nearly all of them redundant. From the trajectory design point of view, this redundancy is real handicap for the optimization process. In this survey, alternative trajectory representations are presented with a description of their advantages and limits. Such new approaches may be applied in many areas:

Aircraft Trajectories Data Compression. As it has been previously mentioned, ATM system manage aircraft trajectories and control them in order to guarantee safety and airspace capacity. Currently those trajectories are represented by the mean of plot lists which are manipulated by ATM software. Every day, all aircraft trajectories are registered into large database for which huge capacity is needed. Based on this new trajectory representation for which redundancy has been removed, the trajectories database may be strongly improved from the capacity point of view. This compressed trajectory format may also be used for improving the trajectories transmission between ATM entities.

Aircraft Trajectories Distance Computation. Although trajectories are well understood and studied, relatively little investigation on the precise comparison of trajectories is presented in the literature. A key issue in performance evaluation of ATM decision support tools (DST) is the distance metric that determines the similarity of trajectories. Some proposed representation may be used to enhance trajectory distance computation.

Aircraft Model Inference. All aircraft models are based on ODEs (Ordinary Differential Equation), including tabular ones. Control input includes condition and model parameters. The model refinement (and computational complexity) ranges from tabular to many degrees of freedom. The aircraft model inference consists in answering the following question: Given a parametrized model and a goal trajectory, can we infer the best parameter values? A model can be viewed as a mapping from the control space into the trajectory space. The way to answer the previous question is then given by the closest model to the goal trajectory.

Trajectory Prediction. Air traffic management research and development has provided a substantial collection of decision support tools that provide automated conflict detection and resolution [13,24,78], trial planning [46], controller advisories for metering and sequencing [19, 75], traffic load forecasting [45, 47], weather impact assessment [25, 41, 74]. The ability to properly forecast future aircraft trajectories is central in many of those decision support tools. As a result, trajectory

prediction (TP) and the treatment of trajectory prediction uncertainty continue as active areas of research and development (e.g. [49, 50, 61, 71, 76]). Accuracy of TP is generally defined as point spatial accuracy (goal attainment) or as trajectory following accuracy. The last one can be rigorously defined by the mean of trajectory space. The first one is a limit case of the second by adding a weight function in the energy functional. Since we may prescribe smoothness accuracy of a simplified model relative to a finer one, may be computed.

Major Flows Definition. When radar tracks are observed over a long period of time in a dense area, it is very easy to identify major flows connecting major airports. The expression "major flows" is often used but never rigorously defined. Based on an exact trajectory distance and a learning classifier, it is possible to answer the following questions: Given a set of observed trajectories, can we split it into "similar" trajectory classes? If yes, classes with highest number of elements will rigorously define the major flows. Given those classes and a new trajectory, can we tell if it belongs to a major flow and which one? The principle of the major flows definition is to use shape space to represent trajectory shapes as points and to use a shape distance (the shape of a trajectory is the path followed by an aircraft, that is the projection in the 3D space of its 4D trajectory. The speed on the path has no impact).

Trajectory Planning. To improve Air Traffic Management, projects have been initialized in order to compel the aircraft in position and in time (4D trajectory) so as to avoid potential conflict and allow for some optimality with respect to a given user cost index, environmental criteria (noise abatement, pollutant emission ...). Depending on the time horizon, several kind of plannings can be designed:

- At a strategical level, only macroscopic indicators like congestion, mean traffic complexity, delays can be taken into account, considering the high level of uncertainty;
- at a pre-tactical level, the accuracy of previous indicators, specially congestion and complexity increases while at the same time early conflict detection can be performed;
- finally, at the tactical level, conflict resolution is the major concern and optimality of the trajectories is only marginally interesting.

As we can see, there are many areas of ATM where trajectories are the main objects that have to be manipulate.

The second part of this survey presents some relevant features of aircraft trajectories. The third part, presents dimension reduction tricks for optimization approaches. The fourth part describes approaches based on wave front propagation in isotropic and anisotropic environments. The fifth part presents automatic control approaches with some application to air traffic control. Finally, the sixth part introduces some path planning techniques via natural language processing and mathematical programming. Finally, the seventh part gives some air traffic management applications of such trajectory design approaches.

2 Some Trajectories Features

In the following all aircraft trajectories will be described as mappings from a time interval $[a,b]$ to a state space E with E either $\mathbb{R}^3$ or $\mathbb{R}^6$ depending on whether speed is assumed to be part of aircraft state or not. Extension to trajectories on a sphere (typically long haul flights) will be sketched only.

2.1 Notations and Terminology

The reference for this section is [6]. Let $\gamma[a,b] \to E$ be a trajectory. The origin of the trajectory is $\gamma(a)$ and the destination is $\gamma(b)$. Those two points are called the endpoints of the trajectory. All trajectories are assumed to be at least continuously differential (class C^1) so that the length of a trajectory $\gamma[a,b] \to E$ is well defined as:

$$l(\gamma) = \int_a^b \|\gamma'(t)\| dt \tag{1}$$

If $\|\gamma'(t)\| = 0$ for some $t \in (a,b)$ the point t is said to be singular. A parametrized curve of class C^p (or more concisely a C^p curve) will be a C^p mapping from an **open** time interval (a,b) to the state space E with no singular points. Any C^1 curve can be parametrized by arc length. Let $\gamma(a,b) \to E$ be such a curve. Defining the mapping $s(a,b) \to (0,l(\gamma))$ by:

$$s(t) = \int_a^t \|\gamma'(t)\| dt \tag{2}$$

We see that by the non-singularity assumption on γ, $s'(t) = \|\gamma'(t)\| > 0$ for any $t \in (a,b)$, so that s is an invertible mapping. Now, $\gamma \circ s^{-1}$ is a mapping from the open interval $(0,l(\gamma))$ to E satisfying:

$$\|(\gamma \circ s^{-1})'\| = \|(\gamma' \circ s^{-1}) \circ (s^{-1})'\| = 1 \tag{3}$$

In the following, we will simply write $\gamma(s), s \in (0,l(\gamma))$ for a curve parametrized by arc length, dropping the variable t.

Remark 1. One must be careful with the respective definitions of trajectories and curves: a curve is defined on an open interval and thus has no endpoints. Nevertheless, any trajectory $\gamma[a,b] \to E$ has an associated curve, namely $\gamma(a,b) \to E$. It is generally more convenient to deal with curves to avoid special treatment of the endpoints.

Remark 2. The non singularity assumption on the underlying curve is very natural when dealing with aircraft trajectories in $\mathbb{R}^3$ since it is not possible for an aircraft to stop except at the endpoints of the trajectory.

Remark 3. While the case $E = \mathbb{R}^3$ is very natural and intuitive, care must be taken when $E = \mathbb{R}^6$ since all the preceding definitions apply in a completely different setting: for example, the non singularity assumption does not implies nowhere zero speed, but only that speed and acceleration cannot both vanish at the same time. The arc length parametrization allows to define very important geometrical quantities when $E = \mathbb{R}^3$.

Definition 1. Let:

$$\gamma(0,l) \to \mathbb{R}^3 \tag{4}$$

be a C^1 curve parametrized by arc length. The unit tangent vector to γ at $s \in (0,l)$ is:

$$\tau(s) = \gamma^{'}(s) \tag{5}$$

It is clear from the definition of parametrization by arc length that $\tau(s)$ is a unit vector.

Definition 2. Let:

$$\gamma(0,l) \to \mathbb{R}^3 \tag{6}$$

be a C^2 curve parametrized by arc length. The curvature of γ at $s \in (0,l)$ is:

$$K(s) = \|\gamma^{''}(s)\| \tag{7}$$

The curvature can be explicitly computed even if the curve γ is not parametrized by arc length. The general formula is:

$$K(t) = \frac{\|\gamma^{'}(t) \wedge \gamma^{''}(t)\|}{\|\gamma^{'}(t)\|^3} \tag{8}$$

with $\wedge$ the vector cross product. Curvature is of primary importance for ATM related studies since as mentioned before aircraft trajectories are mainly made of straight lines and arcs of circle and so have piecewise constant curvature. If at point t the curvature is not zero, the curve is said to be biregular at t. For a curve γ parametrized by arc length, the unit normal vector $\nu(s)$ is defined at all biregular points by:

$$\nu(s) = \frac{\gamma^{''}(s)}{K(s)} \tag{9}$$

Remark 4. A straight line has everywhere zero curvature. However, it is clearly possible to define a unit normal vector. At a biregular point, $\tau(s)$ and $\nu(s)$ are well defined. Taking their cross product gives a new vector $\beta(s) = \tau(s) \wedge \nu(s)$. If the curve γ is assumed to be C^3, it can be shown that $\dot{\beta}(s)$ and $\nu(s)$ are collinear:

$$\beta(s) = T(s).\nu(s) \tag{10}$$

The real number $T(s)$ is called the torsion of the curve at s and represents an obstruction for the curve to be planar. As for the curvature, it is possible to compute the torsion even if the curve is not parametrized by arc length:

$$T(t) = -\frac{det(\gamma'(t), \gamma''(t), \gamma'''(t))}{\|\gamma'(t) \wedge \gamma''(t)\|^2} \tag{11}$$

Torsion is not so useful as curvature for en-route data analysis since only a few number of trajectories have non zero torsion. However, it is very relevant in terminal areas.

Remark 5. The $E = \mathbb{R}^6$ case is again very different, since the geometric meaning of curvature and torsion is not obvious in this setting. Furthermore, the extra degrees of freedom will impose using higher order derivatives in order to build up an equivalent description. A complete treatment goes beyond the scope of the present paper and has little interest for our purpose (in practical applications, the speed information, when available, is used to improve estimates of curvature and torsion and not to study a trajectory in $\mathbb{R}^6$).

3 Trajectory Models for Optimization

This section presents some dimension reduction tricks in order to reduce the dimension of the state space for which an optimization process is searching for an optimal vector of parameter.

This approach is summarized by Fig. 1.

The optimization process controls the parameter vector which is then used to build the trajectory γ for evaluation.

Each coordinate can be considered separately in order to build a given trajectory: $\gamma(t) = [x(t), y(t), z(t)]^T$.

In this section, several trajectory models are presented and compared. Simple models are first presented.

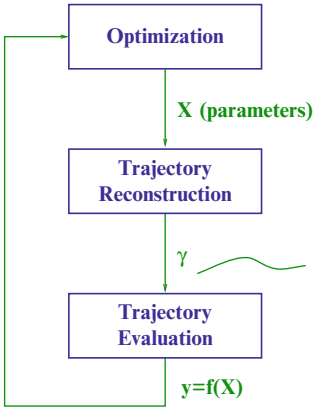

Fig. 1 The optimization process control the X vector in order to build a trajectory γ for evaluation

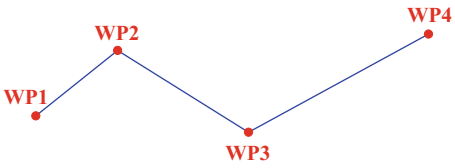

Fig. 2 Trajectory defined by four way points connected by *straight lines*

3.1 Straight Line Segments

One of the easiest way to design trajectory is to use way points connected by straight lines (see Fig. 2). This easy principle ensures continuity for the trajectory but not for its derivatives. If one want to approximate trajectory with many shape turns, one have to increase the number of way points in order to reduce the error between the model of the real trajectory.

In order to improve concept Lagrange interpolation process adjust a polynomial function to a given set of way points.

3.2 Lagrange Interpolation

Given $n+1$ real numbers $y_i, 0 \leq i \leq n$, and $n+1$ distinct real numbers $x_0 < x_1 < \ldots < x_n$, *Lagrange polynomial [39] of degree n* ($L_n(x)$) associated with $\{x_i\}$ and $\{y_i\}$ is a polynomial of degree n solving the interpolation problem:

$$L_n(x_i) = y_i, \ 0 \leq i \leq n \tag{12}$$

$$L_n(x) = \sum_{i=0}^{n} y_i . l_i(x) \tag{13}$$

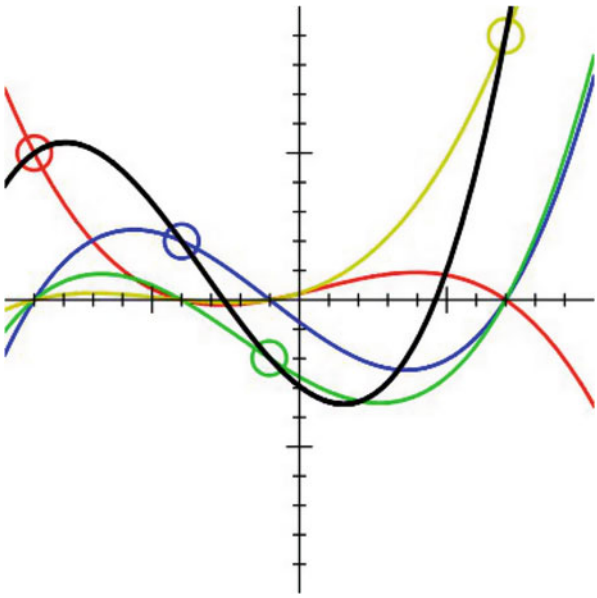

Fig. 3 $L_n(x)$ is represented by the *black curve*. The *other curves* are the polynomials $l_i(x)$

where

$$l_i(x) = \prod_{j \neq i} \frac{(x - x_j)}{(x_i - x_j)} \tag{14}$$

An example of Lagrange interpolation is given in Fig. 3 for which four points are interpolated by the black curve which represents $L_4(x)$. The four polynomial functions $\{l_0(x), l_1(x), l_2(x), l_3(x)\}$ are also given by the red, blue, green and yellow curves.

When derivatives have also to be interpolated, Hermite interpolation has to be used.

3.3 Hermite Interpolation

Hermite interpolation [3] generalizes Lagrange interpolation by fitting a polynomial $(H(x))$ to a function f that not only interpolates f at each knot but also interpolates a given number of consecutive derivatives of f at each knot. This means that the first derivative of the polynomial $H(x)$ have to fit the first derivatives of the function $f(x)$:

$$\left[\frac{\partial^j H(x)}{\partial x^j}\right]_{x=x_i} = \left[\frac{\partial^j f(x)}{\partial x^j}\right]_{x=x_i} \tag{15}$$

for all $j = 0, 1, \ldots, m$ and $i = 1, 2, \ldots, k$

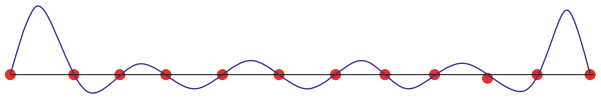

Fig. 4 Lagrange interpolation result for a set of aligned points

This means that $n(m+1)$ values

$$\begin{array}{llll} (x_0,y_0), & (x_1,y_1), & \ldots, & (x_{n-1},y_{n-1}), \\ (x_0,y_0'), & (x_1,y_1'), & \ldots, & (x_{n-1},y_{n-1}'), \\ \vdots & \vdots & & \vdots \\ (x_0,y_0^{(m)}), & (x_1,y_1^{(m)}), & \ldots, & (x_{n-1},y_{n-1}^{(m)}) \end{array} \tag{16}$$

must be known, rather than just the first n values required for Lagrange interpolation. The resulting polynomial may have degree at most $n(m+1)-1$, whereas the Lagrange polynomial has maximum degree $n-1$.

These interpolation polynomials seem attractive but they both induce oscillations between interpolation points (*Runge's phenomenon*). Runge's phenomenon is a problem of oscillation at the edges of an interval that occurs when using polynomial interpolation with polynomials of high degree (which is the case for Lagrange and Hermite interpolation). An example of such Runge's phenomenon is given in Fig. 4 for which Lagrange interpolation has been used.

We can conclude that interpolation with high degree polynomial is risky. In order to avoid this drawback of high degree polynomial interpolation one must use piecewise interpolation.

3.4 Piecewise Linear Interpolation

This is the simplest piecewise interpolation method.

Given $n+1$ real numbers y_i,$0 \leq i \leq n$, and $n+1$ distinct real numbers $x_0 < x_1 < \ldots < x_n$, we consider the n linear curves $lin_i(x) = a_i x + b_i$ on the intervals $[x_i, x_{i+1}]$ for $i = 0, \ldots n-1$ ($lin_i(x)$ represent linear functions for which a_i is the slope and b_i a constant).

Each $l_i(x)$ has to connect two points $((x_i,y_i),(x_{i+1},y_{i+1}))$

$$y_i = a_i x_i + b_i x_i \text{and} y_{i+1} = a_i x_{i+1} + b_i x_{i+1} \tag{17}$$

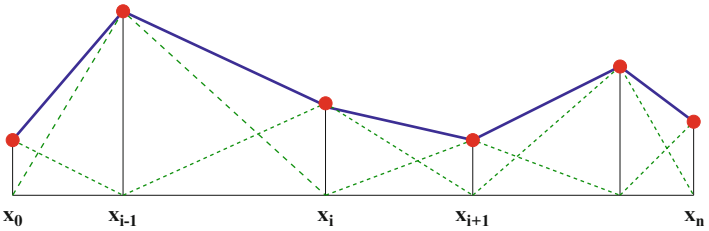

Fig. 5 Piecewise linear interpolation

In order to associate a piecewise formulation of this interpolation method, the following "tent" functions are defined:

$$\psi_i(x) = \begin{cases} \frac{x-x_{i-1}}{x_i-x_{i-1}} & if x \in [x_{i-1}, x_i] \\ \frac{x_{i+1}-x}{x_{i+1}-x_i} & if x \in [x_i, x_{i+1}] \\ 0 & elsewhere \end{cases} \tag{18}$$

Then,

$$f(x) = \sum_{i=0}^{i=n} y_i . \psi_i(x) \tag{19}$$

An example of such a linear piecewise interpolation is given in Fig. 5.

The derivative of the resulting curve is not continuous. In order to fix this drawback, one can use piecewise quadratic interpolation

3.5 Piecewise Quadratic Interpolation

We consider the n quadratic curves $\psi_i(x) = q_i(x) = a_i x^2 + b_i x + c_i$ on the intervals $[x_i, x_{i+1}]$ for $i = 0, \ldots n-1$. Each $q_i(x)$ has to connect two points $((x_i, y_i), (x_{i+1}, y_{i+1}))$; $\Rightarrow y_i = a_i x_i^2 + b_i x_i + c_i$ and $y_{i+1} = a_i x_{i+1}^2 + b_i x_{i+1} + c_i$. Furthermore, on each point, the derivative of the previous quadratic has to be equal to the derivative of the next one; $\Rightarrow 2a_i + b_i = 2a_{i-1} + b_{i-1}$. For the first segment the term $2a_{i-1} + b_{i-1}$ is arbitrarily chosen (this will affects the rest of the curve). An example of piecewise quadratic interpolation is given in Fig. 6. The main drawback of piecewise quadratic interpolation is linked to the effect induced on the curve by moving on point. As a matter of fact moving one point may totally change the shape of the interpolating curve. The piecewise cubic interpolation avoid this drawback.

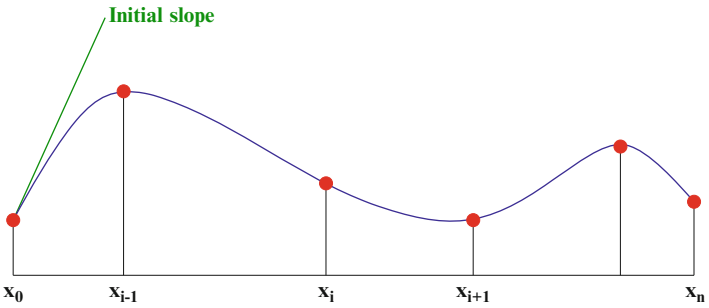

Fig. 6 Piecewise quadratic interpolation. The shape of the entire curve depend of the choice of the initial slope. Between two points, a quadratic polynomial is fitted

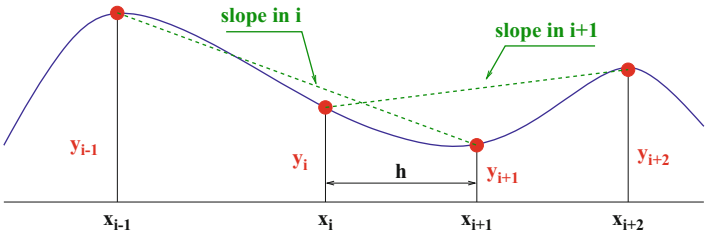

Fig. 7 Piecewise cubic interpolation. The derivative at point x_i is given by *line* joining the point (x_{i-1}, y_{i-1}) and (x_{i+1}, y_{i+1}). Between two points, a cubic polynomial is fitted. The term h represents the distance two consecutive points

3.6 Piecewise Cubic Interpolation

This interpolation is also called Hermite cubic interpolation [33]. For this interpolation:

$$\psi_i(x) = C_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i \tag{20}$$

and we have the following constraints:

$$C_i(x_i) = y_i \qquad C_i(x_{i+1}) = y_{i+1} \tag{21}$$

$$C_i'(x_i) = y_i' = \frac{y_{i+1} - y_{i-1}}{x_{i+1} - x_{i-1}} \; C_i'(x_{i+1}) = y_{i+1}' = \frac{y_{i+2} - y_i}{x_{i+2} - x_i} \tag{22}$$

An example of piecewise cubic interpolation is given in Fig. 7.

Moving a point do not affect all the curve which is the main advantage of this interpolation. The resulting curve is C^1 but not C^2 (the second derivative is not

continuous). The curvature radius of a curve may be expressed by the following expression:

$$R = \frac{1 + \left(\frac{df(x)}{dx}\right)^{\frac{3}{2}}}{\left|\left(\frac{d^2 f(x)}{dx^2}\right)\right|} \tag{23}$$

The piecewise cubic interpolation do not insure that trajectory curvature is continuous which is not adapted for aircraft trajectory mainly in TMA[1] areas and cubic spline interpolation has to be used.

3.7 Cubic Spline Interpolation

This method has been developed by General Motor in 1964 [12]. For this piecewise interpolation $\psi_i(x) = S_i(x)$ with the following constraints:

$$\begin{array}{ll} S_i(x_i) = y_i & S_i(x_{i+1}) = y_{i+1} \\ S_i^{'}(x_i) = S_{i-1}^{'}(x_{i+1}) & S_i^{'}(x_{i+1}) = S_{i+1}^{'}(x_{i+1}) \\ S_i^{''}(x_i) = S_{i-1}^{''}(x_{i+1}) & S_i^{''}(x_{i+1}) = S_{i+1}^{''}(x_{i+1}) \end{array} \tag{24}$$

One can show that $S_i(x)$ for $x \in [x_i, x_{i+1}]$ is given by:

$$\begin{array}{rl} S_i(x) = & \frac{\sigma_i}{6} \cdot \frac{(x_{i+1}-x)^3}{x_{i+1}-x_i} + \frac{\sigma_{i+1}}{6} \cdot \frac{(x-x_i)^3}{x_{i+1}-x_i} \\ & + y_i \cdot \frac{x_{i+1}-x}{x_{i+1}-x_i} - \frac{\sigma_i}{6} \cdot (x_{i+1}-x_i)(x_{i+1}-x) \\ & + y_{i+1} \cdot \frac{x-x_i}{x_{i+1}-x_i} - \frac{\sigma_{i+1}}{6} \cdot (x_{i+1}-x_i)(x-x_i) \end{array} \tag{25}$$

where

$$\sigma_i = \frac{d^2 S_i(x)}{dx^2} \tag{26}$$

An example of such interpolation is given in Fig. 8.

Such spline is also called natural spline because it represents the curve of a metal spline constrained to interpolate some given points.

When interpolation is not a hard constraint, one can use some control points which change the shape of a given trajectory without forcing this trajectory to go through such control point; such approach is called approximation for which one of the famous methods is the Bézier curve.

[1] TMA: "Terminal Maneuvering Area".

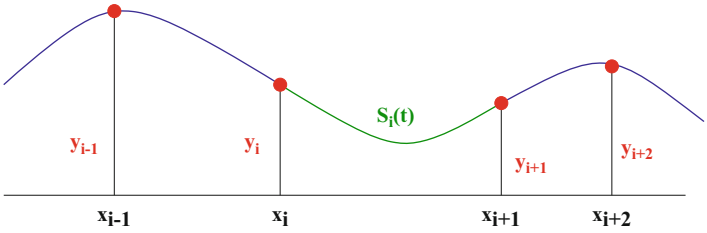

Fig. 8 Cubic spline interpolation

Fig. 9 Bézier curve with two points

3.8 Bézier Approximation Curve

Bézier curves[28] were widely publicized in 1962 by the French engineer Pierre Bézier, who used them to design automobile bodies. But the study of these curves was first developed in 1959 by mathematician Paul de Casteljau using de Casteljau's algorithm [27], a numerically stable method to evaluate Bézier curves. A Bézier curve is defined by a set of control points P_0 through P_n, where n is called its order ($n = 1$ for linear, 2 for quadratic, etc.). The first and last control points are always the end points of the curve; however, the intermediate control points (if any) generally do not lie on the curve. Given points P_0 and P_1, a linear Bézier curve $B(t)$ is simply a straight line between those two points (see Fig. 9). The curve is given by:

$$B(t) = P_0 + t(P_1 - P_0) = (1-t)P_0 + tP_1 \ , t \in [0,1] \tag{27}$$

With four points (P_0, P_1, P_2, P_3), a Bézier curve of degree three can be built. The curve starts at P_0 going towards P_1 and arrives at P_3 coming from the direction of P_2. Usually, it will not pass through P_1 or P_2; these points are only there to provide directional information (see Fig. 10).

Properties

- The polygon formed by connecting the Bézier points with lines, starting with P_0 and finishing with P_n, is called the Bézier polygon (or control polygon).
- The convex hull[2] of the Bézier polygon contains the Bézier curve.

[2]The convex hull or convex envelope of a set X of points in the Euclidean plane or Euclidean space is the smallest convex set that contains X.

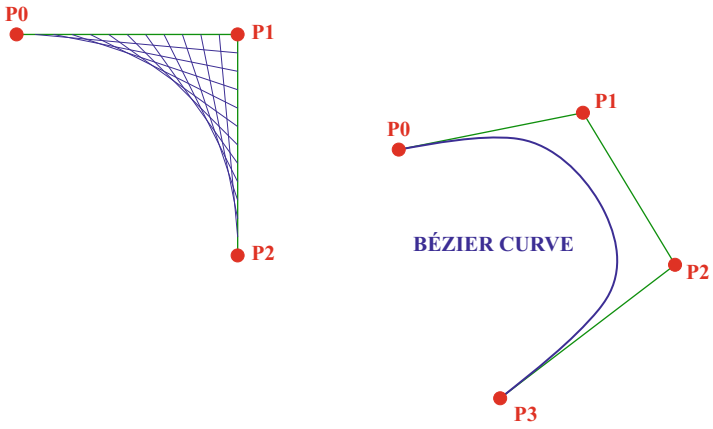

Fig. 10 Cubic Bézier curve

- The start (end) of the curve is tangent to the first (last) section of the Bézier polygon.

  The explicit form of the curve is given by:

$$B(t) = (1-t)^3 P_0 + 3(1-t)^2 t P_1 + 3(1-t) t^2 P_2 + t^3 P_3 \, , t \in [0,1]. \tag{28}$$

$$B(t) = \sum_{i=0}^{n} b_{i,n}(t) P_i, \quad t \in [0,1] \tag{29}$$

where the polynomials

$$b_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}, \quad i = 0, \ldots n \tag{30}$$

are known as Bernstein basis polynomials of degree n. So, if there are many points, one has to manipulate polynomials with high degree. In order to circumvent this weak point one must use Basis-Splines.

3.9 Basis Splines

A B-spline [21] is a spline function that has minimal support with respect to a given degree, smoothness, and domain partition. B-splines were investigated as early as the nineteenth century by Nikolai Lobachevsky. A fundamental theorem states that every spline function of a given degree, smoothness, and domain partition, can

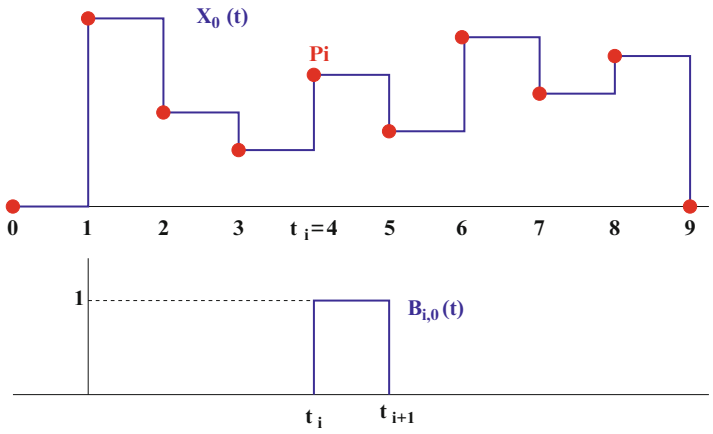

Fig. 11 Uniform B-splines of degree zero

be uniquely represented as a linear combination of B-splines of that same degree and smoothness, and over that same partition. It is a powerful tool for generating curves with many control points, B stands for basis. A single B-spline can specify a long complicated curve and B-splines can be designed with sharp bends and even "corners". B-Spline interpolation is preferred over polynomial interpolation because the interpolation error can be made small even when using low degree polynomials for the spline. Furthermore, spline interpolation avoids the problem of Runge's phenomenon which occurs when interpolating between equidistant points with high degree polynomials.

3.9.1 Uniform B-Splines of Degree Zero

We consider a node vector $T = \{t_0, t_1, \ldots, t_n\}$ with $t_0 \leq t_1 \leq, \ldots, \leq t_n$ and n points P_i. One want to build a curve $X_0(t)$ such that:

$$X_0(t_i) = P_i \tag{31}$$

$\Rightarrow X_0(t) = P_i \; \forall t \in [t_i, t_{i+1}]$.

$$X_0(t) = \sum_i B_{i,0}(t).P_i \tag{32}$$

where

$$B_{i,0}(t) = \begin{cases} 1 \text{ if} t \in [t_i, t_{i+1}] \\ 0 \text{ elsewhere} \end{cases} \tag{33}$$

The shape of the $X_0(t)$ function in one dimension is given in Fig. 11.

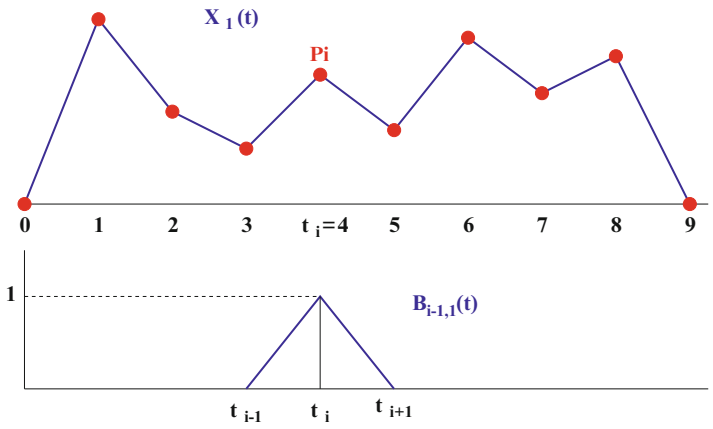

Fig. 12 Uniform B-splines of degree one

3.9.2 Uniform B-Splines of Degree One

We are searching for a piecewise linear approximation $X_1(t)$ for which:

$$X_1(t) = \left(1 - \frac{t - t_i}{t_{i+1} - t_i}\right) P_{i-1} + \left(1 - \frac{t - t_i}{t_{i+1} - t_i}\right) P_i \quad \forall t \in [t_i, t_{i+1}] \tag{34}$$

One can write $X_1(t)$:

$$X_1(t) = \sum_i B_{i,1}(t).P_i \tag{35}$$

where

$$B_{i,1}(t) = \begin{cases} \frac{t - t_{i-1}}{t_i - t_{i-1}} & \text{if } t \in [t_{i-1}, t_i] \\ \frac{t_{i+1} - t}{t_{i+1} - t_i} & \text{if } t \in [t_i, t_{i+1}] \\ 0 & elsewhere \end{cases} \tag{36}$$

The shape of the $X_1(t)$ function in one dimension is given in Fig. 12.

3.9.3 Uniform B-Splines of Degree Three

Those B-Splines have been developed at Boeing in the 1970s and represent one of the simplest and most useful cases of B-splines. Degree 3 B-Spline with $n + 1$ control points is given by:

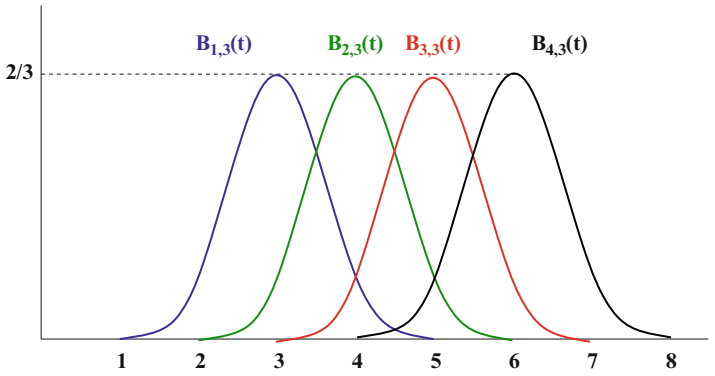

Fig. 13 Order 3 basis function

$$X_3(t) = \sum_{i=0}^{n} B_{i,3}(t).P_i \ 3 \leq t \leq n+1 \tag{37}$$

where $B_{i,3}(t) = 0$ if $t \leq t_i$ or $t \geq t_{i+4}$.

$$X_3(t) = \sum_{i=j-3}^{j} P_i.B_{i,3}(t) \ t \in [j, j+1], \ 3 \leq j \leq n \tag{38}$$

When a single control point P_i is moved, only the portion of the curve $X_3(t)$ is changed (with $t_i < t < t_{i+4}$) insuring local control property. The basis functions have the following properties:

- They are translates of each other i.e. $B_{i,3}(t) = B_{0,3}(t-i)$
- They are piecewise degree three polynomial
- Partition of unity $\sum_i B_{i,3}(t) = 1$ for $3 \leq t \leq n+1$
- The functions $X_i(t)$ are of degree 3 for any set of control points

$$B_{i-2,3}(t) = \frac{1}{h} \begin{cases} (t-t_{i-2})^3 \ \text{ if } t \in [t_{i-2}, t_{i-1}] \\ h^3 + 3h^2(t-t_{i-1}) + 3h(t-t_{i-1})^2 - 3(t-t_{i-1})^3 \\ \text{ if } t \in [t_{i-1}, t_i] \\ h^3 + 3h^2(t_{i+1}-t) + 3h(t_{i+1}-t)^2 - 3(t_{i+1}-t)^3 \\ \text{ if } t \in [t_i, t_{i+1}] \\ (t_{i+2}-t)^3 \text{ if } t \in [t_{i+1}, t_{i+2}] \\ 0 \text{ otherwise} \end{cases} \tag{39}$$

where h is the distance between two consecutive points.

Those basis functions are shown in Fig. 13.

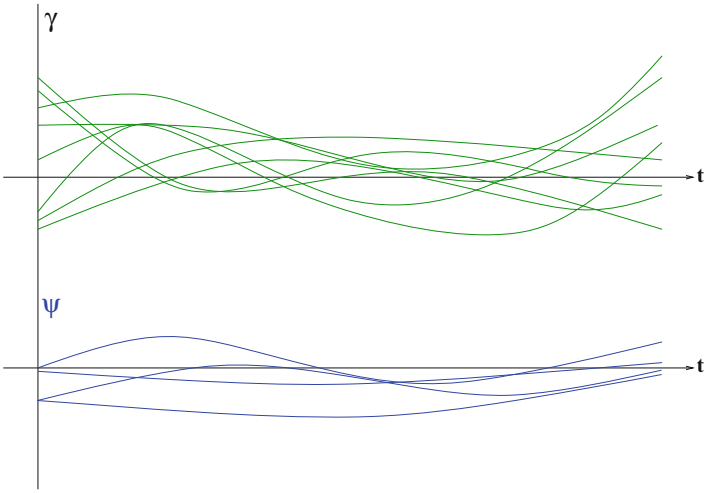

Fig. 14 The *black* trajectories represent registered samples for which four principal components are extracted (in this artificial example) for minimum error reconstruction process

3.9.4 Principal Component Analysis

When trajectories samples are available (from radar for instance), one can build a dedicated bases which will minimize the number of coefficient for trajectory reconstruction. Principal component analysis (PCA) is a mathematical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. The number of principal components is less than or equal to the number of original variables.

In the example presented in Fig. 14 a set of trajectories $\gamma_i(t), i = 1 \ldots n$ are used to build $K = 4$ principal components $(\psi_k(t))$ which can be used to reconstruct the initial trajectories.

$$\gamma_i(t) = \sum_{k=1}^{k=K} a_{ik} \psi_k(t) \tag{40}$$

When probability density functions of the coefficient a_{ik} can be identified, one can use this trick to plug a stochastic optimization process which generates random coefficients in order to produce relevant random trajectory. More information about Functional PCA can be found in [56].

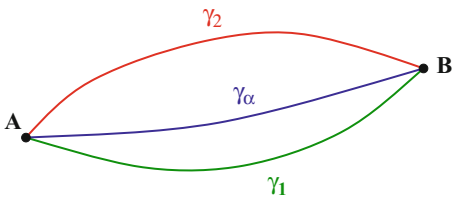

Fig. 15 One new trajectory is built by using a weighted sum of two reference trajectories

3.9.5 Homotopy Trajectory Design

An easy way to build trajectory is to used reference trajectory (regular trajectories used by aircraft) and to compute a weighted sum of such reference trajectories to build a new one. If we consider two (or more) references trajectories joining the same origin destination pair (see Fig. 15) (past flown trajectories may be considered):

$$\gamma_1, \gamma_2 \tag{41}$$

One can create a new trajectory γ_α by using an homotopy:

$$\gamma_\alpha = (1-\alpha)\gamma_1 + \alpha\gamma_2 \tag{42}$$

In this example only one coefficient α has been used but one can extend this principle to several parameters.

All the models described in this section, may be used in an optimization process for which dimension reduction is needed. Depending on the targeted properties of the designed trajectories one must select the most adapted representations for the underlying application.

After having review algorithms based on optimization, the next section presents wave front propagation approaches.

4 Wavefront Algorithms

4.1 Generalities

It is a well-known fact in physics that waves of high frequencies tend to propagate along the path whose traveling (or propagation) time is shortest (such minimum time path are called geodesic). The knowledge of the wave velocity at each point of space allows for the computation of wave fronts that are the set of all points reached by the wave at a given time, assuming it has started at a point source. The principle of the wavefront propagation algorithms is to simulate such physical propagation model in order to find optimal path based on a criterion that has to be optimized. As an example, congestion will be taken into account by velocity reduction, so that paths

crossing congested areas will be penalized. Depending on the fact that the metric is or not isotropic (the later case being the one to be investigated when the wind is used into the criterion and has a non negligible velocity), two classes of algorithms are used.

4.2 Fast Marching Algorithms

The *Fast Marching* method, presented by Sethian [66] is a part of the more general methods called *Level Set* [53]. These techniques are designed to track the evolution of interfaces. The evolution of the wavefront can be compared to deform a curve or a surface from a partial differential equation. The *Fast Marching* method is used in the particular case where the wavefront speed is isotropic. It can still be applied if the anisotropy is low enough. For air traffic applications, this last assumption is valid in some areas of the airspace (of course it is not the case in the vicinity of jet streams).

In the particular case where the Fast Marching method is applicable, the calculus of the minimum time T to reach any points of the environment from the initial point is equivalent to solve the Eikonal equation of the form:

$$|\nabla T(x)| = \frac{1}{F(x)}, \qquad F(x) > 0, \qquad \text{and} \qquad \mathrm{T}(\mathrm{x}_{\mathrm{initial}}) = 0 \tag{43}$$

where $x \in \mathbb{R}^2$ represents the position in space, $T \in \mathbb{R}$ the minimum time and $F : \mathbb{R} \to \mathbb{R}$ the speed of propagation.

In free space, the wave speed F is equivalent to the aircraft speed. When we have forbidden areas, we force the propagation speed at zero in order to get a barrier value since the time to reach this point will be equal to infinity. Thereby, we have guaranteed avoidance property for those areas. For the congestion, the method is different, we want to penalize some areas where the congestion is high but we do not want to ban aircraft from driving through these areas. We just need to reduce the propagation speed. Thus, the time is increased proportionally to the congestion value, penalizing the crossing.

To design the optimal path between the arrival point and the departure point, we can then perform a gradient descent using the calculated values of T on the space, from the arrival point to the initial point. There is no risk to get stuck on a local minimum since the function T has only one optimum which is global.

The numerical resolution is like the graph search algorithms. However, in opposition to these graph search algorithms, the *Fast Marching* method is consistent since when the grid is refined, the obtained solution converges on the exact solution of the Eikonal equation [66] that is a geodesic curve.

4.3 Ordered Upwind Algorithm

When wind is to be taken into account and has a non negligible speed with respect to the one of the aircraft, the propagation is no longer isotropic. The speed of the wavefront depends on the position and the directions of wind. A specific algorithm, called *Ordered Upwind*, has been developed to overcome this problem in [67], at the expense of a higher algorithmic complexity. Basically, an extra parameter, the anisotropy ratio, is considered: it is the ratio of the fastest to slowest propagation speed for each points. Given a point in space, the algorithm first considers the points on the current wavefront that are closest to it: it gives a time to travel when taking as propagation speed the slowest. Now, the other points to be considered are located no farther than the anisotropy ratio times the minimal distance. By maintaining a list of potential points contributing to the information at the point of interest, the ordered upwind algorithm can still be implemented in a single pass.

In order to keep the efficiency of the *Fast Marching* algorithm, Petres proposed an extension of the algorithm *Fast Marching* in [55], he assumed the field is smooth. He applied this extension to plan a path for autonomous underwater vehicles taking underwater currents into account. We propose here a similar extension to Petres's method of the *Fast Marching* method, our extension is specific to aircraft trajectories.

In the next section, we present an example of application of such wave propagation algorithm for air traffic management problem.

4.4 Light Propagation Algorithm

In geometric optics, light behavior is modeled using rays. Light emitted from a point is assumed to travel along such a ray through space. In an effort to explain the motion through space taken by rays as they pass through various media, Fermat (1601–1665) developed his *principle of least action* [29]:

The path of a light ray connecting two points is the one for which the time of transit, not the length, is a minimum.

In the framework of geometrical optics, light propagates through space respecting Descartes' Law:

Let n_1 be the index of refraction of medium 1 in which the incident ray propagates and n_2 that of the medium in which the refracted ray propagates (medium 2). We obtain:

$$\begin{cases} n_1.\sin(\theta_1) = n_2.\sin(\theta_2) \\ v_1 = \frac{c}{n_1}, v_2 = \frac{c}{n_2}, \end{cases} \tag{44}$$

where v_1,v_2 are the speed values in media 1 and 2, θ_1 the angle of incidence, θ_2 the angle of the refracted ray and c the speed of light in a vacuum.

We can make several observations as a result of Fermat's principle:

- In a homogeneous medium, light rays are rectilinear. That is, within any medium where the index of refraction is constant, light travels in a straight line.
- In an in-homogeneous medium, light rays follow smooth geodesic curves with minimum transit time.

Light therefore tends to avoid high index areas where rays are slowed down. Light reaches lowest speed for the highest encountered index.

Based on this principle of least action, we introduce an optimal path planning algorithm which computes smooth geodesic trajectories in environments with static or dynamic obstacles.

For a given mobile, we wish to find the shortest path between two points of $\mathbb{R}^n$, taking account of a given metric (time, distance etc.) whilst avoiding obstacles and respecting speed constraints. The trajectories produced must also respect a regularity criterion characterized by a maximum curvature value.

This algorithm mimics light propagation between a starting point towards a destination point, with obstacles modeled by high-index areas. By controlling the index landscape, it is possible to ensure that the computed trajectories meet the speed constraints and remain at a specified minimum distance from obstacles.

We begin by positioning a light source at the departure point (origin) which emits rays in a hemisphere oriented towards the destination point (this restriction prevents the generation of unrealistic trajectories which begin by turning 180° before turning back towards the destination). The path followed by the first beam to reach the arrival point corresponds to the geodesic trajectory we wish to obtain (see Fig. 16).

5 Optimal Control for Trajectory Generation

5.1 Optimal Trajectory Generation

In the physical space, a trajectory is occasionally represented as a four-dimensional flight path, following the tradition of air traffic control [18], with time as the fourth dimension, in addition to the normally used three-dimensional representation of a path. Generating *time-parameterized* paths necessitates the incorporation of the aircraft dynamics and/or kinematics, which makes the problem much more difficult than simply finding a path that avoids obstacles in the physical three-dimensional space. Path-planning is a term commonly used in the robotics and artificial intelligence communities to refer to the problem of generating an obstacle-free path to be followed by a vehicle (robot, aircraft, vehicle, etc.) in a two or three dimensional space containing obstacles [42].

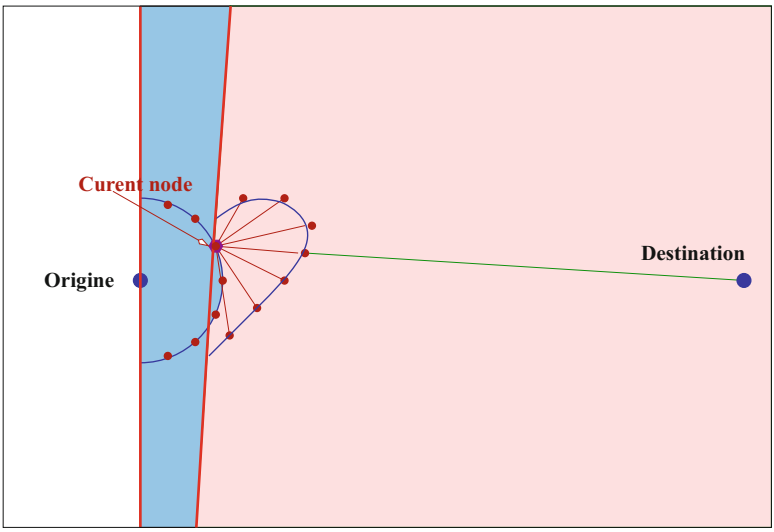

Fig. 16 Launching rays from the departure point. Geodesic computation (A^* like algorithm or Triangle mesh algorithm)

Because the vehicle dynamics are not taken into account in these path-planning methods (the solution of which only considers the geometric constraints of the problem) it is often the case that the resulting path is infeasible, that is, it cannot be followed exactly or even closely by the vehicle. One way to ensure that the resulting paths correspond to feasible trajectories satisfying the vehicle dynamics, is to use optimal control theory. The objective of optimal control theory is to determine the control input(s) that will cause a process (i.e., the response of a dynamical system) to satisfy the physical constraints, while, at the same time, minimize (or maximize) some performance criterion. Feasibility of the trajectories is automatically ensured using this approach. The typical optimal control problem (OCP) can be stated as follows:

Given initial conditions x_0, final conditions $x_f \in \mathscr{X}$, and an initial time $t_0 \geq 0$, determine the final time $t_f > t_0$, the control input $u(t) \in \mathscr{U}$ and the corresponding state history $x(t)$ for $t \in [t_0, t_f]$ which minimize the cost function

$$J(x,u) = \int_{t_0}^{t_f} L(x(t),u(t))\,\mathrm{d}t, \tag{45}$$

where $x(t)$ and $u(t)$ satisfy, for all $t \in [t_0, t_f]$ the differential and algebraic constraints:

$$\dot{x}(t) - f(x(t),u(t)) = 0, \tag{46}$$

$$C(x(t),u(t)) \leq 0. \tag{47}$$

Optimal control has its roots in the theory of calculus of variations, which originated in the seventeenth century by Fermat, Newton, Liebniz, and the Bernoullis, and was subsequently further developed by Lagrange, Weirstrass, Legendre, Clebsch and Jacobi and others in the eighteenth and nineteenth centuries [26]. Calculus of variations deals with the problem of minimizing (45) subject to the simple differential equality constraint of the form $\dot{x}(t) - u(t) = 0$, and is not able to handle more complicated differential equality constraints such as (46) or algebraic constraints such as (47). It was not until the middle of the twentieth century when the Soviet mathematician L.S. Pontryagin developed a complete theory that could handle constraints such as (46) and (47). Simply put, Pontryagin's celebrated Maximum Principle [54] states that the optimal control for the solution of the problem (45)–(47) is given as the pointwise minimum of the so-called Hamiltonian function, that is,

$$u_{\text{opt}} = \text{argmin}_{u \in U} H(t, x, \lambda, u), \tag{48}$$

where $H(t, x, \lambda, u) = L(x, u) + \lambda^T f(x, u)$ is the Hamiltonian, and λ are the co-states, computed from

$$\dot{\lambda}(t) = -\frac{\partial H}{\partial x}(x(t), \lambda(t), u(t)). \tag{49}$$

subject to certain boundary (transversality) conditions on $\lambda(t_f)$. Unfortunately, an analytic solution to the previous problem is difficult. The optimal control formulation of a trajectory optimization problem using Pontryagin's Maximum Principle (PMP) leads to a Two-point Boundary Value Problem (TBVP), or a Multi-point Boundary Value Problem (MBVP) when the optimal trajectory is composed of multiple phases. Numerical techniques such as shooting and multiple shooting methods can be applied to solve accurately these TBVP and MBVP, but their convergence is very sensitive to the choice of an initial guess for the solution. A software that solves the optimal control problem using this approach is BNDSCO [51]. BNDSCO is an example of a class of numerical optimization methods which are often referred to as indirect methods. The term indirect reflects the fact that in these methods a solution is sought not by maximizing (or minimizing) the cost (45) but, rather, by computing potential optimizers by solving the corresponding necessary optimality conditions (48)–(49).

In recent years, direct methods have become increasingly popular for solving trajectory optimization problems, the major reason being that direct methods do not require an analytic expression for the necessary conditions, which for complicated nonlinear dynamics can be intimidating. Moreover, direct methods do not need an initial guess for the co-states whose time histories are difficult to predict a priori. As mentioned earlier, direct methods, do not try to satisfy the necessary conditions of optimality from PMP, instead, they minimize directly (45) subject to (46)–(47).

The main idea behind direct methods is to discretize the states and controls of the original continuous-time optimal control problem in order to obtain a

finite-dimensional nonlinear programming problem (NLP). The solution of this NLP, which consists of discrete variables, is used to approximate the continuous control and state time histories. Typical direct methods are collocation methods, which discretize the ordinary differential equations (ODEs) of the problem using collocation or interpolation schemes [23, 32, 60, 79]. They introduce the collocation conditions as NLP constraints together with the initial and terminal conditions. The so-called pseudospectral methods use orthogonal polynomials to choose the collocation points and are very efficient, exhibiting superlinear convergence when the solution is smooth. Numerical optimal control software packages that implement direct methods for the solution of OCPs include SOCS [8], RIOTS [65], DIDO [59], PSOPT [4], GPOPS [57], MTOA [36] and DENMRA [83] among many others. A recent survey of numerical optimal control techniques for trajectory optimization can be found in [7].

In all these direct methods, the convergence rate and the quality of solution depends on the grid used to discretize the equations, the cost and the problem constraints. Uniform or fixed grid methods tend to perform poorly, especially when the problem has several discontinuities or irregularities. Not surprisingly, adaptive grid methods have been developed to accurately capture any discontinuities or switchings in the state or control variables. The main idea behind all these adaptive grid methods is to use a high resolution (dense) grid only in the vicinity of control switches, constraint boundaries etc., and a coarse grid elsewhere. Examples of such adaptive gridding techniques for the solution of optimal control problems are [9–11, 30, 37, 65].

A major issue with almost all current trajectory optimization solvers (direct or indirect) is the fact that their computational complexity is high and their convergence dependents strongly on the initial conditions, unless certain rather stringent convexity conditions hold. As a result, the solution of trajectory optimization problem in *real-time* is still elusive. A common line of attack for solving trajectory optimization problems in real time (or near real time) is to divide the problem into two phases: an offline phase and an online phase [38, 44, 69, 81]. The offline phase consists of solving the optimal control problem for various reference trajectories and storing these reference trajectories onboard for later online use. These reference trajectories are used to compute the actual trajectory online via a neighboring optimal feedback control strategy typically based on the linearized dynamics. Another strategy for computing near-optimal trajectories in real-time is to use a receding horizon (RH) approach [5, 52, 80]. In a receding horizon approach a trajectory that optimizes the cost function over a period of time, called the *planning horizon*, is designed first. The trajectory is implemented over the shorter *execution time* and the optimization is performed again starting from the state that is reached at the end of the execution time. A third approach is to use a two-layer architecture, where first an acceptable (in terms of length, safety, etc.) path is computed using common path-planning techniques, and then an *optimal* time-parameterization is imposed on this path to yield a feasible trajectory. As mentioned earlier such an approach needs to be carefully designed to ensure compatibility of the resulting path with the vehicle dynamics. However, when successful, such an approach is numerically very efficient

and can be implemented in real-time with current computer hardware. Even if the resulting trajectory is not exactly feasible, it is often close to a feasible trajectory, or it can be made as such using smoothing techniques [85]. As a result, alternatively, the final trajectory can be used as a good initial guess for a follow-up optimal trajectory generator. The next section summarizes this approach for applications related to aircraft maneuvering under strict time and fuel constraints. For more details the interested reader is referred to [2, 82, 84].

5.2 *Emergency Aircraft Trajectory Design*

Aircraft maneuvering was one of the first areas where optimal control theory was used to generate optimal trajectories. Not surprisingly, traditionally, most work has been focused on military aircraft. Relevant references on this subject are too many to enumerate here. We just mention the work on fuel and range optimization studied in [31, 68, 70], and the minimum-time, three-dimensional aircraft trajectory optimization problem considered in [64]. In the latter, an approximation of the aircraft dynamics using an energy state was used to reduce the dimension of the problem for better convergence. This type of model reduction technique is commonly used for aircraft trajectory optimization [40].

Some of these results have been extended to commercial airline operations. For example, the work of [16] deals with the problem of minimum-fuel trajectories with fixed time-of-arrival (TOA) for several civil aviation aircraft including B737, B747 and B767. Trajectory planning problems have also been studied in the context of air traffic management (ATM) and automation. Jackson et al. [34] performed a sensitivity analysis of trajectory prediction for ATM. The aircraft trajectory synthesis problem is studied in [71] to provide some basic tools for air traffic automation. Somewhat related is the recent work of Sridhar [72], in which he considered the generation of wind-optimal trajectories for cruising aircraft while avoiding the regions of airspace that facilitate persistent contrails formation. A shooting method was employed to solve the associated optimal control problem by minimizing a weighted sum of flight time, fuel consumption, and a term penalizing the contrail formation. The airspace avoidance problem has also been considered in [35]. In that reference, the avoidance of restricted airspace is formulated as a non-convex constrained trajectory optimization problem; it is claimed that with a feasible starting guess, the efficiency of the optimization algorithm is not degraded too much by the non-convex airspace constraints. Finally, some researchers have used ideas from *stochastic* optimal control to deal with issues related to the unpredictability of future trajectory, wind effects, presence of additional aircraft in the ATM airspace etc. [43, 77].

In this work we deal with the problem of *efficiently* generating minimum-time and minimum-fuel (with fixed TOA) landing trajectories for commercial aircraft. The former problem is of relevance in case of an on-board emergency where the pilot has to land the airplane quickly and safely to the closest airport or airfield;

the latter problem is of interest for typical terminal ATC phase applications. Prior work in emergency landing includes the abort landing problem in the presence of windshear [14, 15, 48], and emergency landing during loss of thrust [73]. The latter references generate feasible trajectories using segments of trajectories corresponding to selected trim condition maneuvers. The search results are however limited to those that can be generated by connecting trim state trajectory segments with stable transitions. Because the unstable flight conditions are not considered in the search, the algorithm cannot identify any feasible trajectories containing unstable flight modes. Furthermore, the path length is used as the search criterion, which is less appropriate when compared to flight time for emergency landing, or fuel consumption for normal flight. Related work includes the investigation of Atkins et al. [1], where the problem of emergency landing due to the loss-of-thrust was studied using a hybrid approach. A two-step landing-site selection/trajectory generation process was adopted to generate safe emergency plans in real time under situations that require landing at an alternate airport. In the trajectory generation routine, a heuristic path planner was used to generate a three-dimensional trajectory connecting the current position of the aircraft to the runway, which consists of straight lines and circular arcs. This method is fast and simple. However, it is limited to conservative aircraft maneuvers (typically Dubins paths) in order to reduce the chance of obtaining an infeasible trajectory. As a result, the optimality of the generated trajectory could be unacceptable for emergency landing, and further research is necessary to reduce such a conservatism.

In our approach, we start with the assumption that the path to be followed by the aircraft is given. Note that this does not mean that the *trajectory* to be followed is given. A trajectory requires a time-parameterized path and it is, indeed, the main goal of this approach to provide such a time parameterization so as to meet feasibility along with certain optimality specifications. The assumption that the path is given is not as unusual or atypical as one may initially think. Commercial airliners during the terminal landing phase, are required to follow strict Air Traffic Control (ATC) rules, which guide the airplanes to follow "virtual" three-dimensional corridors all the way to the landing strip. Furthermore, since our approach leads to very fast computation of feasible trajectories, one can use the approach over new, locally modified paths repeatedly till a satisfactory path is found.

To this end, let a path in the three-dimensional space, parameterized by the path coordinate s, be given as follows: $x = x(s)$, $y = y(s)$, $z = z(s)$, where $s \in [s_0, s_f]$. The main objective is to find a time-parameterization along the path, i.e., a function $s(t)$, where $t \in [0, t_f]$ such that the corresponding time-parameterized trajectory $(x(s(t)), y(s(t)), z(s(t)))$ minimizes either the flight time t_f (emergency landing case) or fuel (terminal landing operation). As shown in [82] all control (thrust, angle of attack, load factor, etc.) constraints can be mapped into constraints involving the specific kinetic energy of the aircraft, $E = v^2/2$ where v is the aircraft velocity of the form

$$\underline{g}_w(s) \leq E(s) \leq \overline{g}_w(s),$$

for some path-dependent functions $\underline{g}_w(s)$ and $\overline{g}_w(s)$. The original problems therefore reduce to the following simplified problems:

For the Minimum-Time Problem we have

$$\min_T \quad \int_{s_0}^{s_f} \frac{\mathrm{d}s}{\sqrt{2E(s)}} \tag{50a}$$

$$\text{subject to } E'(s) = \frac{T(s)}{m} - D(E(s),s) - g\sin\gamma(s), \tag{50b}$$

$$\underline{g}_w(s) \leq E(s) \leq \overline{g}_w(s), \tag{50c}$$

$$T_{\min} \leq T(s) \leq T_{\max}, \tag{50d}$$

and for the Minimum-fuel Problem with fixed TOA we have

$$\min_T \quad \int_{s_0}^{s_f} T(s)\,\mathrm{d}s, \tag{51a}$$

$$\text{subject to } E'(s) = \frac{T(s)}{m} - D(E(s),s) - g\sin\gamma(s), \tag{51b}$$

$$t'(s) = \frac{1}{\sqrt{2E(s)}}, \tag{51c}$$

$$\underline{g}_w(s) \leq E(s) \leq \overline{g}_w(s) \tag{51d}$$

$$T_{\min}(s) \leq T(s) \leq T_{\max}(s), \tag{51e}$$

where $D(E(s),s)$ is the drag, T is the thrust, γ is the flight-path angle, and where prime denotes differentiation with respect to path length s. The main advantage of these problem formulations is the dimensionality reduction of the problem that can be leveraged to solve both of these problems very efficiently and reliably. In fact, these OCPs are simple enough so that the optimal switching structure of the optimal solution can be unraveled using the necessary conditions from PMP. For the minimum-fuel problem the switching structure varies depending on the given TOA. However, for a given path and a fixed TOA, the structure is uniquely determined. This helps tremendously the convergence properties of the algorithm.

The overall architecture for optimal on-line trajectory generation is shown in Fig. 17. As shown in this figure, the method first generates a trajectory by assigning an optimal time parameterization along the path given by the geometric path planner via the solution of one of the previous two optimization problems. If the trajectory is feasible, then it is used as an initial guess for the numerical optimal control solver. Meanwhile, such a feasible trajectory is also stored as a back-up plan in case of the failure of the NLP solver. If the trajectory generated by the time-optimal path tracking method is not feasible, then the path is revised using the path smoothing method described in [85], and the optimization is applied again to the smoothed path. The process is repeated until either the trajectory is feasible, or the maximum number of iterations is reached. If no feasible trajectory can be

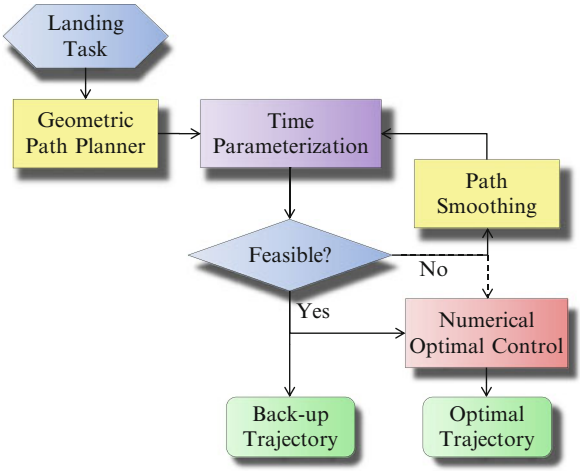

Fig. 17 Schematic of landing trajectory optimization

obtained after reaching the iteration limit, the infeasible trajectory is passed to the numerical optimal control algorithm, which makes a last attempt to produce a feasible trajectory. If this last attempt is not successful, then it does not exist a feasible trajectory that solves the problem.

6 Path Planning Techniques via Natural Language Processing and Mathematical Programming: A Paradigm for Future Aircraft Trajectory Management

Aircraft trajectory planning has reached enough maturity to shift the trajectory planning problem from the mathematical optimization of the aircraft trajectory to the automated parsing and understanding of desired trajectory goals, followed by their re-formulation in terms of a mathematical optimization program. To a large extent, we propose a possible evolution of the Flight Management System currently used on all transport aircraft to become a full-fledged autonomous logic that may also be used on unmanned aerial vehicles as an alternative to the current -and bulky- Remotely-Piloted Vehicle (RPV) paradigm. What follows is a direct extension of work originally presented in [63].

The overall proposed architecture is shown in Fig. 18. It consists of several nested feedback loops. At the operator level, the information is presented to the operator in the form of sentences expressed in natural language (e.g. that used by air traffic control phraseology). At the level of trajectory planning automation, the information is presented as a mix of continuous parameters (aircraft position and speed), and discrete parameters describing mission status (completed tasks, tasks remaining to be completed). The operator formulates the vehicles's goals (e.g. flight plans) using natural language. A natural language interpreter and task scheduler

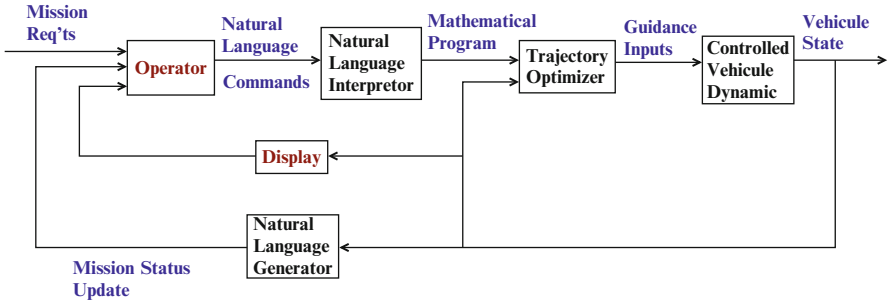

Fig. 18 Basic autonomous mission management loop for future aircraft and UAVs

transforms the operator's requirements into tractable mathematical optimization programs that may be executed by the vehicle through its flight control computer. The vehicle's innermost dynamics (that consist of raw vehicle dynamics and stability augmentation system), although critical to vehicle stability, are not shown.

6.1 Natural Language Parsing and Generation

The main goal of using a Natural Language Interface (NLI) for interacting with a computer-based system is to minimize the workload on the operator. Using normal English sentence commands and reports indeed allows an operator to communicate efficiently and effectively with an aircraft, as if it were a human pilot. The NLI module that we have developed for demonstration purposes consists of two major components. The first one takes sentence commands from the operator (presumably an air traffic controller) and turns them into a formally coded command that looks like the formulation of an optimization problem. The second component takes a coded command set from the aircraft and generates natural language responses for the air traffic controller to interpret. A sample dialog between the human operator and the machine could be as follows:

Controller: *Flight AA1234, this is Air Traffic Control.*
Aircraft: *Go ahead, Air Traffic Control.*
Controller: *Add new waypoint. Proceed to waypoint Echo-Charlie 5 in minimum time. and wait for further instructions after the task is completed*
Aircraft: *Roger. Acknowledge task information - proceeding to waypoint Echo-Charlie 5.*
Controller: *AA1234, out.*

The NLI module analyzes the natural sentences produced by the air traffic controller using parsing, which is the process of converting an input sentence, e.g. "Proceed to waypoint Echo-Charlie 5 in minimum time," into a formal representation. The latter is typically a tree structure which can in turn be translated

into an explicit formal command. In our system, parsing consists of first applying entity extraction to all the individual concepts (e.g. “Flight AA1234” or “Echo-Charlie 5”) and then combining these concepts through cascades of finite-state transducers using techniques derived from those described in [58]. While natural language processing represents our progress so far, it is easy to imagine that it could now be completed by a voice recognition device to further ease the level of communication between controller (or operator) and aircraft.

6.2 Task Scheduling and Communications Interfacing

The task scheduling and communications processing components are designed to centralize all of the aircraft mission processing in one module. Together with the Natural Language Interface, it provides flexibility for an operator to insert and change flight plan during the flight. The aircraft software keeps track of the flight tasks, waypoint locations and known obstacles to pass on to the guidance algorithm. The communications processing component provides the air traffic controller or operator with the authority to send commands and receive status updates, threat or obstacle avoidance information and acknowledgement messages. It also provides remote pilots monitoring the flight with the ability to override the guidance system in the event of an emergency or error. The system sends threat and override information to the air traffic controller before any status or update information in an effort to send the most important data relevant to the demonstration before any auxiliary information. Input/Output data are processed every 1 Hz frame before the task planner and guidance step to ensure that the most up-to-date information is used by the aircraft trajectory planner. The task scheduling component operates like a Flight Management System and allows the aircraft operator or the air traffic manager to enter a flight plan using a pre-defined list or as programmed during a mission. Many additional features may be added to such a task scheduler, such as orders to follow loiter patterns, “take me home” or low-emission approaches functionalities. In addition, he or she has the option of providing (in real- time via the NLI) the optimization metric used by the trajectory generation algorithm (i.e. minimum time, minimum fuel, or the amount of time to finish the flight). Next, the operator can either give the aircraft a new plan or change the current plan it is performing. A “New Plan” command is added to the end of the aircraft task list and is executed after all of the tasks currently scheduled have been completed. A “Change Plan” command, on the other hand, modifies the current task performed by the aircraft. Once a task is completed, it is removed from the list. After each of these actions, an acknowledgement is sent to the air traffic controller and the updated task information is included in the data sent to the Trajectory Generation Module.

6.3 Trajectory Planning

After the Natural Language Interface and Flight Planning and Scheduling components have converted the flight plan into a series of tasks for the aircraft to perform, the Trajectory Generation Module guides the vehicle from one task to the next, i.e. from an initial state to a desired one, through an obstacle field while optimizing a certain objective. The latter can be to minimize time, fuel or a combination of both. Much of the functionality described below becomes increasingly available in today's avionics systems, and also include such real-world factors as weather and wind conditions. For our purposes, 2D scenarios were considered in which special-use airspace and other no-fly zones are viewed as "obstacles" and detected while the flight proceeds. The environment is always fully characterized inside a certain detection region D around the aircraft. The resulting formulation can, however, be easily generalized to account for any detection shape, such as a radar cone, and for unknown areas within that shape. Since trajectories must be dynamically feasible, the aircraft dynamics and kinematics should be accounted for in the planning problem. For optimization purposes, the vehicle is characterized by a discrete time, linear state space model (A,B) in an inertial 2D coordinate frame (east-north). As such, the state vector $\mathbf{x}$ consists of the east-north position (x,y) and corresponding inertial velocity $(\dot{x},\dot{y})$. Depending on the particular model, the input vector u is an inertial acceleration or reference velocity vector. In both cases, however, combined with additional linear inequalities in x and u, the state space model must capture the closed-loop dynamics that result from augmenting the aircraft with a waypoint tracking controller. Since the environment is only partially-known and further explored in real-time, a receding horizon planning strategy is used to guide the vehicle towards the desired destination.

The destination is denoted by x_f and is an ingress/egress state of a waypoint with a corresponding inertial velocity vector. At each time step, a partial trajectory from the current state towards the goal is computed by solving the trajectory optimization problem over a limited horizon of length T. Because of the computation delay, the initial state $\mathbf{x}_0 = (x_0, y_0, \dot{x}_0, \dot{y}_0)$ in the optimization problem should be an estimate $\mathbf{x}_{\text{estim}}$ of the position and inertial velocity of the aircraft when the plan is actually implemented. The solution to the optimization problem provides a sequence of waypoints (x_i, y_i) and corresponding inertial reference velocities $(\dot{x}_i, \dot{y}_i)$ to the aircraft for the next T time steps. Typically, however, only the first waypoint and reference velocity of this sequence are given to the waypoint follower, and the process is repeated at the next time step. As such, new information about the state of the vehicle and the environment can be taken into account at each time step. By introducing a cost function J_T over the T time steps, the general trajectory optimization problem can be formulated as to

$$\text{Minimize}_{\mathbf{x}_i,u_i} J_T = \sum_{i=1}^{T} f_i(\mathbf{x}_i,u_i,\mathbf{x}_f) + f_T(\mathbf{x}_T,\mathbf{x}_f)$$

$$\text{Subject to} \begin{cases} \mathbf{x}_{i+1} = A\,\mathbf{x}_i + Bu_i, \;\; i = 0,\ldots,T-1 \\ \mathbf{x}_0 = \mathbf{x}_{\text{estim}} \\ \mathbf{x}_i \in \mathscr{X}_0, \;\; i = 1,\ldots,T \\ u_i \in \mathscr{U}_0, \;\; i = 0,\ldots,T-1 \\ (x_i,y_i) \in \mathscr{D}_0, \;\; i = 1,\ldots,T \\ (x_i,y_i) \notin \mathscr{O}_0, \;\; i = 1,\ldots,T \end{cases}$$

The objective function consists of stage costs $f_i(\mathbf{x}_i,u_i,\mathbf{x}_f)$ corresponding to each time step i, and a terminal cost term $f_T(\mathbf{x}_T,\mathbf{x}_f)$ that accounts for an estimate of the cost-to-go from the last state $\mathbf{x}_T$ in the planning horizon to the goal state $\mathbf{x}_f$. The sets $\mathscr{X}_0$ and $\mathscr{U}_0$ represent the (possibly non-convex) constraints on the vehicle dynamics and kinematics, such as bounds on velocity, acceleration and turn rate. Here, the 0-subscript denotes the fact that these constraints can be dependent on the initial state. Lastly, the expressions $(x_i,y_i) \in \mathscr{D}_0$ and $(x_i,y_i) \notin \mathscr{O}_0$ capture the requirement that the planned trajectory points should lie inside the known region $\mathscr{D}_0$, but outside the obstacles $\mathscr{O}_0$ as given at the current time step $i = 0$. Note that they are assumed to hold for $\mathbf{x}_0$; if not, the trajectory optimization problem would be infeasible from the start. As demonstrated in [62], however, despite the detection region and avoidance constraints, the above receding horizon strategy has no safety guarantees regarding avoidance of obstacles in the future. Namely, the algorithm may fail to provide a solution in future time steps due to obstacles that are located beyond the surveillance and planning radius of the vehicle. For instance, when the planning horizon is too short and the maximum turn rate relatively small, the aircraft might approach a no-fly zone too closely before accounting for it in the trajectory planning problem. As a result, it might not be able to turn away in time, which translates into the optimization problem becoming infeasible at a future receding horizon iteration. In [62], a safe receding horizon scheme was therefore proposed based on maintaining a known feasible trajectory from the final state $\mathbf{x}_T$ in the current planning horizon towards an obstacle-free holding pattern. The latter must lie in the region $\mathscr{D}_0$ of the environment that is fully characterized at the current time step, and is computed and updated online. Assuming that the planned trajectories can be accurately tracked, at each time step, the remaining part of the previous plan together with the holding pattern can then always serve as an a priori safe backup or "rescue" plan. In practice, we have found that formulating the problem of finding the nominal and rescue trajectories optimization as mixed-integer programs (MIP) works very well in practice, though such choices are not mandatory and may be replaced by the other techniques discussed in this paper.

7 Application to Air Traffic Management

Aircraft trajectory design can be applied in many areas of air traffic managements like strategic planning, pre-tactical planning, tactical planning, SID-STAR design, emergency trajectory design, etc.. . . In the following three Air Traffic Management applications are presented for which trajectory design is critical. The first one presents the strategic aircraft trajectory planning (optimization approach), the second one the pre-tactical and tactical planning by light propagation algorithm (wave propagation approach) finally the last one describes Emergency trajectory design by optimal control (optimal control approach).

7.1 Strategic Aircraft Trajectory Planning

The aim of strategic planning is to reduce airspace congestion by modifying takeoff slots and routes for a set of flights on a country-wide scale.We thus considered a sectorized airspace and a set of flight plans for a given day. For each flight, we defined a set of alternative routes and a set of possible takeoff slots. Our goal was to produce an optimal assignment to minimize airspace congestion. Our problem can be presented as follows:

- We consider all of the flight plans associated with the airspace of a country.
- For each airplane k, we suppose that the following elements are known:
  - a set of possible routes (+ associated costs)
  - a set of possible takeoff times (+ associated costs)
  - the set of flights connected with flight k at departure and arrival points (hub phenomenon)

We wish to obtain the configuration which allows us to reduce airspace congestion, minimize cost and respect connection plans.

The architecture of the chosen approach (see Fig. 19) is made up of a traffic simulator and a genetic algorithm (GA).

The simulator is used in pre-treatment to obtain the data needed for optimization. It receives flight plans (airplane identification, airplane type, origin-destination, departure time, flight profile, route as a set of beacons) and constructs airplane trajectories, while saving the necessary data (sector input and output, flight position and direction every minute, etc.).

Genetic algorithms simulate the process of natural selection in a hostile environment linked to the problem under consideration [20]. They use a vocabulary similar to that found in natural genetics, without neglecting the fact that the underlying principles involved in the two domains are considerably more complex in their natural context. We thus refer to individuals in a population, and often individuals will be reduced to a single chromosome. These chromosomes themselves are made up of genes which contain the hereditary characteristics of the individual. We also use principles of selection, crossing (crossover), mutation etc.

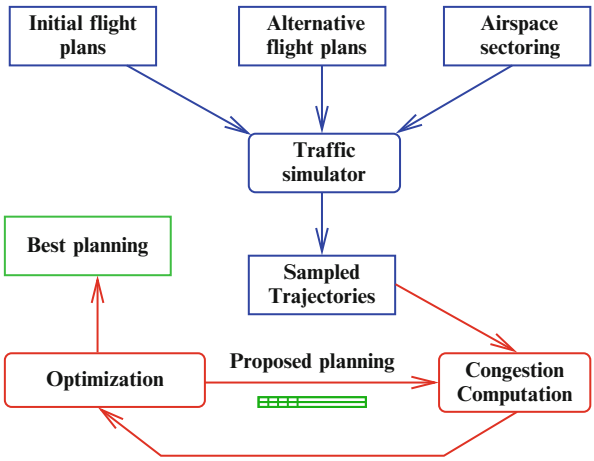

Fig. 19 Optimization structure

In the context of optimization, each individual represents a point in the state space to which we associate the value of the criterion to optimize. We then randomly generate a population of individuals from which the genetic algorithm aims to select the best specimens while ensuring efficient exploration of the state space.

The optimization algorithm then manipulates possible plannings, represented by disturbances, in terms of routes and slots, to the initial flight plans.

The optimization generates plannings which are increasingly efficient in terms of minimizing congestion. The best planning is then put forward as the result of the optimization process.

This approach has been successfully applied for removing conflict in European airspace with 32000 aircraft. The purpose of this research is to design a gate-to-gate conflict free planning by adding way points (inducing a maximum of 10 % extra-distance) and/or by shifting the time on departure (maximum shift: ± 30 min). The optimal altitude profiles have been used. Direct route planning induces $\simeq$400,000 interactions between trajectories. Figure 20 shows the European traffic with interaction in red.

Figure 21 represents the conflict free traffic after optimization.

More information about this work may be found in [17].

7.2 Pre-tactical and Tactical Planning by Light Propagation Algorithm

The objective of the proposed method, based on an analogy with optics, is to find an optimal 4D trajectory for each aircraft which avoids congestion or conflicts and minimizes a criterion based on a local metric.

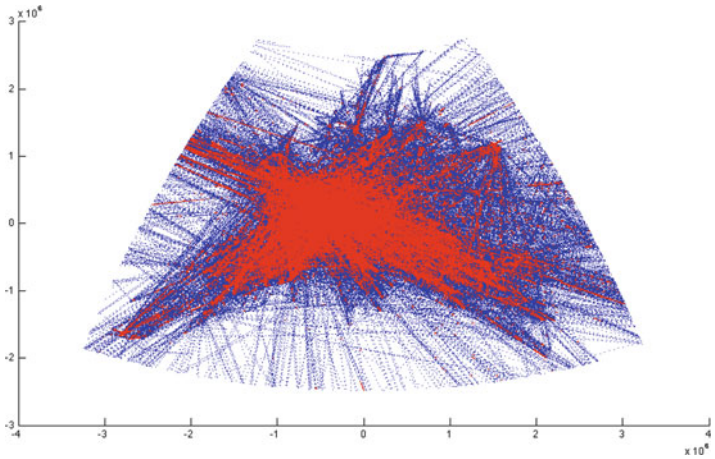

Fig. 20 Direct route traffic over Europe

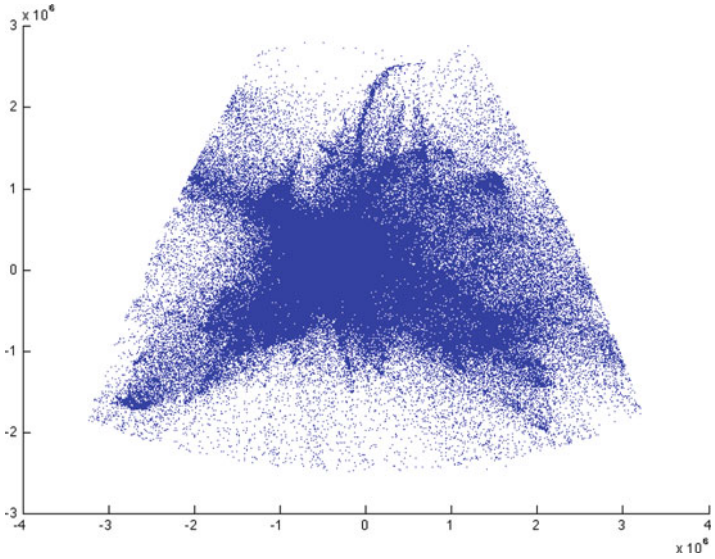

Fig. 21 Traffic picture after optimization

Congestion and the *protection zone* (volume surrounding the aircraft where no other aircraft may enter) of other aircraft will be modeled as high-index areas. Our *light propagation algorithm* (LPA) is designed from a particular aircraft point of view. It is assumed that the aircraft knows the surrounding aircraft trajectories (the set of trajectories of the other aircraft is a given input of the algorithm).

Some examples of geodesic trajectories computation are given on the following Fig. 22 for which the red areas contain high index values.

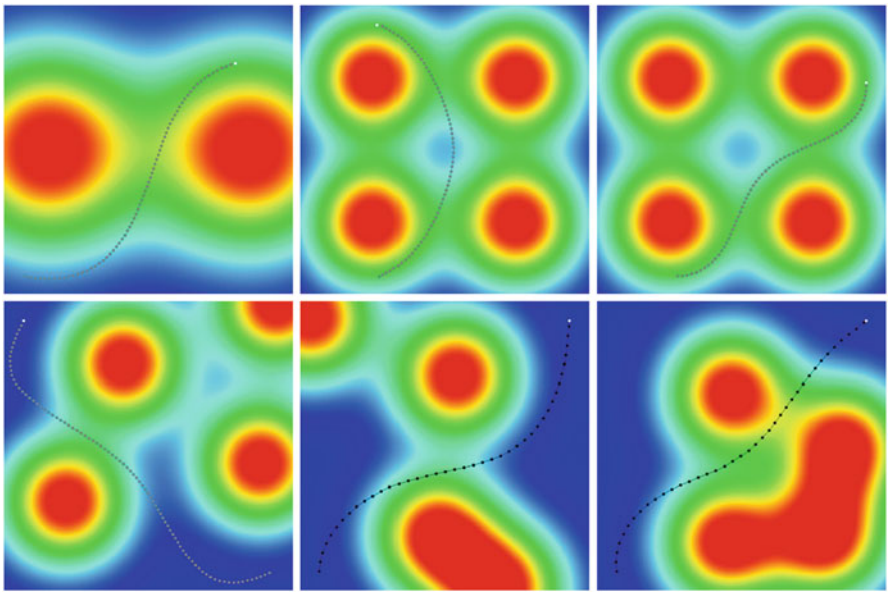

Fig. 22 Example of optimal geodesic curves. The *red color* indicate high index areas

This algorithm has also been used at tactical phase in order to produce conflict free trajectories. To generate a trajectory, we use a wavefront propagation algorithm in 2D + time (in order to conform to operational practice, we carry out resolution in terms of heading, but with a detection method which takes account of altitude) with temporal sampling (the wave is propagated with a time step dt).

Real flight plans of 12 August 2008 with about 8,000 flights have been used to produce reference trajectories. The initial trajectories (before conflict resolution) induce a total number of around 4,000 conflicts. Based on a sliding time window, trajectory segments are extracted in order to built conflict cluster for which LPA is applied. To address the conflict resolution, LPA is sequentially applied to aircraft. We assign a trajectory to the first aircraft disregarding the other aircraft (without considering any constraints). Then, LPA looks for a trajectory for the subsequent aircraft by considering the trajectory of the first aircraft as a constraint, and so on, up to the m^{th} aircraft which considers the $m-1$ previous aircraft trajectories as constraints. In practice, some operational criteria may also be used in order to select a specific sequence (for instance: first-come first-served rule, some aircraft may have higher priority, trajectory length, etc.). The time window is then shifted and the process is applied again.

The algorithm nearly solves all conflicts, with only 28 situations for which conflict-free trajectories have not been found. However, these situations correspond to some aircraft being already in conflict at the beginning of the simulation, for instance at their starting point. Only 1,501 trajectories have been modified to reach such a conflict-free planning. In many cases, the new computed trajectories are

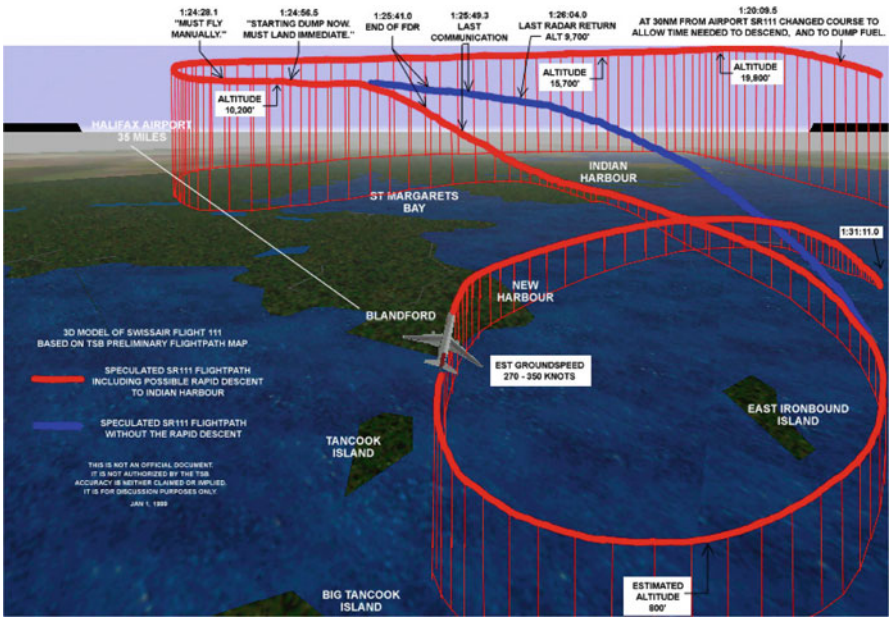

Fig. 23 Swissair 111 trajectory reconstruction

shorter than the initial ones (those that follow waypoints), due to the fact that LPA is searching for the shortest path trajectories and proposes direct routes when possible. More information about this work may be found in [22].

7.3 Emergency Trajectory Design by Optimal Control

This work presents an application of optimal control for computing backup trajectories in case of emergency. In such a situation, one must be able to design backup landing trajectories for aircraft that have lost having loose their thrust. The proposed algorithm begin to find a flyable path to avoid obstacles (Dubins paths generation with continuous descent), then find a feasible trajectory to follow along this path.

This requires the solution of optimal time parametrization (or velocity generation) problem. The latter is a one-dimensional optimal control problem that can be solved very efficiently.

Two cases are presented in the following, the first one is link to the Swissair 111 crash and the second one to the US Air 1549 crash.

On Wednesday, 2 September 1998, the Swiss Air 111 flight crashed into the Atlantic Ocean southwest of Halifax International Airport (due to fire on board).

The Swissair 111 trajectory is given in Fig. 23.

The algorithm has been applied to this case and has produced several backup trajectories bringing back the aircraft at Halifax airport as it can be shown in Fig. 24.

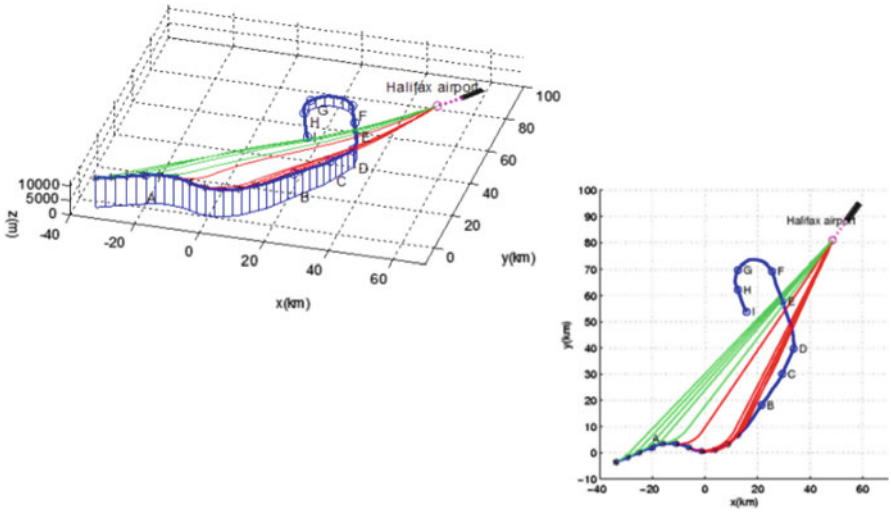

Fig. 24 Swissair 111 case backup trajectories generation in *green*. The *blue* trajectory represent the real flown trajectory

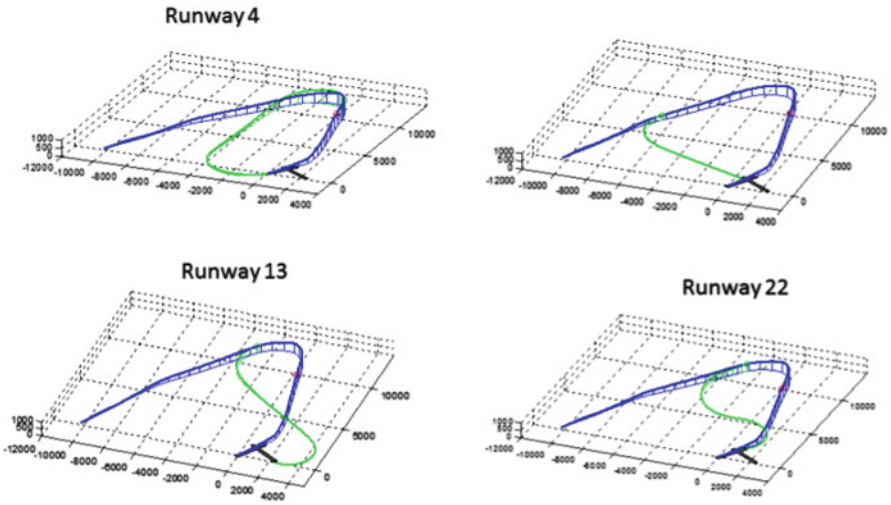

Fig. 25 US Air 1549 backup trajectories in *green*. The *blue* trajectory represent the real flown trajectory ending in Hudson river

This algorithm has also been applied to the US Air 1549 case. On January 15, 2009, the US Air 1549 flight struck a flock of Canada Geese during its initial climb out, lost engine power, and ditched in the Hudson River off midtown Manhattan. Again, the algorithm found four backup trajectories bringing back safely the aircraft the Laguardia airport (see Fig. 25).

8 Conclusion

This survey has shown several approaches for trajectory modeling. As it has been mentioned, trajectories are belonging to infinite dimension space for which dimension reduction has to be implemented. Using trajectory samples vectors is really redundant and inefficient. After having presented some definitions and some features of aircraft trajectories, some dimension reductions methods have been presented in order to be included in an optimization process. Interpolation and approximation technique have been presented and some information have also been given about Principal Component Analysis and Homotopy. The next section has presented wave front propagation approaches and gives some details on Fast Marching Algorithms, Ordered upwind algorithms and the light propagation algorithm (LPA). The fourth part has focused on methods coming automatic control for which vehicle dynamics are explicitly included in the models. Some applications to air traffic management have also been presented. The sixth part has presented an original path planning techniques mixing natural language processing and mathematical programming. Finally, some air traffic management applications have been presented for realistic cases.

References

1. Atkins EM, Portillo IA, Strube MJ (2006) Emergency flight planning applied to total loss of thrust. J Guid Control Dyn 43(4):1205–1216
2. Bakolas E, Zhao Y, Tsiotras P (2011) Initial guess generation for aircraft landing trajectory optimization. In: AIAA guidance, navigation, and control conference. AIAA
3. Bartels RH, Beatty JC, Barskyn BA (1998) An introduction to splines for use in computer graphics and geometric modeling. Computer graphics. Morgan Kaufmann, San Francisco
4. Becerra VM (2011) Psopt optimal control solver user manual. Tech. report
5. Bellingham J, Kuwata Y, How J (2003) Stable receding horizon trajectory control for complex environment. In: AIAA guidance, navigation, and control conference and exhibit. AIAA
6. Berger M, Gostiaux B (1988) Differential geometry: manifolds, curves and surfaces. Springer, New York
7. Betts JT (1998) Survey of numerical methods for trajectory optimization. J Guid Control Dyn 21(2):193–207
8. Betts JT, Huffman WP (1997) Sparse optimal control software SOCS. Tech. report, Mathematics and Engineering Analysis Technical Document MEALR-085, Boeing Information and Support Services, The Boeing Company
9. Betts JT, Huffman WP (1998) Mesh refinement in direct transcription methods for optimal control. Optim Control Appl Methods 19(1):1–21
10. Betts JT, Biehn N, Campbell SL, Huffman WP (2000) Compensating for order variation in mesh refinement for direct transcription methods. J Comput Appl Math 125:147–158
11. Binder T, Blank L, Dahmen W, Marquardt W (2000) Grid refinement in multiscale dynamic optimization. Tech. report, RWTH Aachen
12. Birkhoff G, de Boor C (1964) Piecewise polynomial interpolation and approximation. In: Proceeding of the general motors symposium of 1964. General Motors

13. Brudnicki DJ, McFarland AL (1997) User request evaluation tool (uret) conflict probe performance and benefits assessment. In: Proceeding of the air traffic management seminar, FAA/Eurocontrol
14. Bulirsch R, Montrone F, Pesch HJ (1991) Abort landing in the presence of windshear as a minimax optimal control problem. Part I: necessary conditions. J Optim Theory Appl 70(1):1–23
15. Bulirsch R, Montrone F, Pesch HJ (1991) Abort landing in the presence of windshear as a minimax optimal control problem. Part II: multiple shooting and homotopy. J Optim Theory Appl 70(2):223–254
16. Burrows JW (1983) Fuel-optimal aircraft trajectories with fixed arrival times. J Guid Control Dyn 6(1):14–19
17. Chaimatanan S, Delahaye D, Mongeau M (2012) Conflict free strategic planning. In: Proceeding of the 2012 interdisciplinary science for innovative air traffic management conference. ERAU
18. Chakravarty A (1985) Four-dimensional fuel-optimal guidance in the presence of winds. J Guid Control Dyn 8(1):16–22
19. Coppenbarger R, Lanier R, Sweet D, Dorsky S (2004) Design and development of the enroute descent advisor (eda) for conflict-free arrival metering. In: Proceeding of the AIAA-2004-4875 AIAA GNC conference. AIAA GNC
20. Davis L (1991) Handbook of genetic algorithms. Van Nostrand Reinhold, New York
21. de Boor C (1978) A practical guide to splines. Springer, New York
22. Dougui N, Delahaye S, Puechmorel D, Mongeau M (2012) A light-propagation model for aircraft trajectory planning. J Glob Optim 56:873–895
23. Enright PJ, Conway BA (1992) Discrete approximation to optimal trajectories using direct transcription and nonlinear programming. J Guid Control Dyn 15:994–1002
24. Erzberger H, Paielli RA, Isaacson DR, Eshowl MM (1997) Conflict detection in the presence of prediction error. In: Proceeding of the air traffic management seminar, FAA/Eurocontrol
25. Evans J et al (2003) Reducing severe weather delays in congested airspace with weather support for tactical air traffic management. In: Proceeding of the air traffic management seminar, FAA/Eurocontrol
26. Ewing GM (1969) Calculus of variations with applications. Norton, New York, reprinted by Dover, 1985
27. Farin G (1993) Curves and surfaces for computer aided geometric design. A practical guide. Academic, San Diego
28. Farin G, Hansford D (2000) The essentials of CAGD. A K Peters, Natick
29. Giancoli DC (1989) Physics for scientists and engineers with modern physics, 2nd edn. Prentice-Hall, Englewood Cliffs
30. Gong Q, Fahroo F, Ross IM (2008) Spectral algorithm for pseudospectral methods in optimal control. J Guid Control Dyn 31(3):460–471
31. Grimm W, Well K, Oberle H (1986) Periodic control for minimum-fuel aircraft trajectories. J Guid Control Dyn 9(2):169–174
32. Hargraves CR, Paris SW (1992) Direct trajectory optimization using nonlinear programming and collocation. J Guid Control Dyn 15:994–1002
33. Heath MT (2002) Scientific computing, an introductory survey. Computer graphics. McGraw-Hill, New York
34. Jackson MR, Zhao Y, Slattery RA (1999) Sensitivity of trajectory prediction in air traffic management. J Guid Control Dyn 22(2):219–228
35. Jacobson M, Ringertz UT (2010) Airspace constraints in aircraft emission trajectory optimization. J Aircraft 47:1256–1265
36. Jain S, Tsiotras P (2008) Multiresolution-based direct trajectory optimization. J Guid Control Dyn 31(5):1424–1436
37. Jain S, Tsiotras P (2008) Trajectory optimization using multiresolution techniques. J Guid Control Dyn 31(5):1424–1436
38. Jardin MR, Bryson AE (2001) Neighboring optimal aircraft guidance in winds. J Guid Control Dyn 24:710–715

39. Jeffreys H, Jeffreys BS (1988) Methods of mathematical physics. Cambridge University Press, Cambridge
40. Kelley HJ (1973) Control and dynamic systems: advances in theory and applications. Academic, New York
41. Kirk DB et al (2001) Problem analysis resolution and ranking (parr) development and assessment. In: Proceeding of the air traffic management seminar, FAA/Eurocontrol
42. LaValle SM (2006) Planning algorithms. Cambridge University Press, Cambridge
43. Liu W, Hwang I (2012) Probabilistic aircraft mid-air conflict resolution using stochastic optimal control. IEEE Intell Transp Syst Trans Mag
44. Lu P (1999) Regulation about time-varying trajectories: precision entry guidance illustrated. J Guid Control Dyn 22:784–790
45. Masalonis A et al (2004) Using probabilistic demand prediction for traffic flow management decision support. In: Proceeding of the AIAA-2004-4875 AIAA GNC conference. AIAA GNC
46. McNally BD, Bach RE, Chan W (1998) Field test evaluation of the ctas conflict prediction and trial planning capability. In: Proceeding of the AIAA-1998-4480 AIAA GNC conference. AIAA GNC
47. Meckiff C, Chone R, Nicolaon JP (1998) The tactical load smoother for multi-sector planning. In: Proceeding of the air traffic management seminar, FAA/Eurocontrol
48. Miele A (1990) Optimal trajectories and guidance trajectories for aircraft flight through windshears. In: Proceedings of the 29th IEEE conference on decision and control. IEEE
49. Mondoloni S, Bayraktuta I (2005) Impact of factors, conditions and metrics on trajectory prediction accuracy. In: Proceeding of the air traffic management seminar, FAA/Eurocontrol
50. Mondoloni S, Pagli SM, Green S (2002) Trajectory modeling accuracy for air traffic management decision support tools. In: Proceeding of the ICAS conference. ICAS, Toronto
51. Oberle HJ, Grimm W (1989) BNDSCO - a program for the numerical solution of optimal control problems. Tech. report. Institute for Flight System Dynamics, German Aerospace Research Establishment Oberpfaffenhofen
52. Ohtsuka T (2002) Quasi-Newton-type continuation method for nonlinear receding horizon control. J Guid Control Dyn 24:685–692
53. Osher S, Sethian JA (1988) Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. J Comput Phys 79(1):12–49
54. Pontryagin LS, Boltyanski VG, Gamkrelidze RV, Mischenko EF (1962) The mathematical theory of optimal processes. Interscience, New York
55. Pêtrès C, Pailhas Y, Patron P, Petillot Y, Evans J, Lane D (2007) Path planning for autonomous underwater vehicles. IEEE Trans Robot 23(2):331–341
56. Ramsay JO, Silverman BW (2005) Functional data analysis. Springer series in statistics. Springer, New York
57. Rao AV, Benson D, Huntington GT (2011) User's manual for GPOPS version 4.x: a matlab package for software for solving multiple-phase optimal control problems using hp-adaptive pseudospectral methods. Tech. report
58. Roche E (1997) Parsing with finite-state transducers. In: Roche E, Schabes Y (eds) Finite-state language processing. MIT Press, Cambridge
59. Ross IM (2005) User's manual for DIDO: a MATLAB application package for solving optimal control problems. Tech. report, Naval Postgraduate School
60. Russell RD, Shampine LF (1972) A collocation method for boundary value problems. Numerische Mathematik 19:13–36
61. Ryan HF, Paglione M, Green S (2004) Review of trajectory accuracy methodology and comparison of error measure metrics. In: Proceedings of the AIAA-2004-4787 AIAA GNC conference. AIAA GNC
62. Schouwenaars T, How J, Feron E (2004) Receding horizon path planning with implicit safety guarantees. In: American control conference, Boston, MA, June 2004, pp 5576–5581
63. Schouwenaars T, Valenti M, Feron E, How J, Roche E (2006) Linear programming and language processing for human/unmanned-aerial-vehicle team missions. AIAA J Guid Control Dyn 29(2):303–313

64. Schultz RL (1990) Three-dimensional trajectory optimization for aircraft. J Guid Control Dyn 13(6):936–943
65. Schwartz A (1996) Theory and implementation of numerical methods based on runge-kutta integration for solving optimal control problems. Ph.D. thesis, Université Montpellier II, France
66. Sethian JA (1999) Fast marching methods. SIAM Rev 41(2):199–235
67. Sethian JA, Vladimirsky A (2003) Ordered upwind methods for static Hamilton-Jacobi equations: theory and algorithms. SIAM J Num Anal 41:325–363
68. Seywald H (1994) Long flight-time range-optimal aircraft trajectories. J Guid Control Dyn 19(1):242–244
69. Seywald H, Cliff EM (1994) Neighboring optimal control based feedback law for the advanced launch system. J Guid Control Dyn 17:1154–1162
70. Seywald H, Cliff EM, Well K (1994) Range optimal trajectories for an aircraft flying in the vertical plane. J Guid Control Dyn 17(2):389–398
71. Slattery RA, Zhao Y (1997) Trajectory synthesis for air traffic automation. J Guid Control Dyn 20(2):232–238
72. Sridhar B, Ng HK, Chen NY (2011) Aircraft trajectory optimization and contrails avoidance in the presence of winds. J Guid Control Dyn 34:1577–1583
73. Strube MJ, Sanner RM, Atkins EM (2004) Dynamic flight guidance recalibration after actuator failure. In: AIAA 1st intelligent systems technical conference. AIAA
74. Sud V et al (2001) Air traffic flow management collaborative routing coordination tools. In: Proceeding of the AIAA-2001-4112 AIAA GNC conference. AIAA GNC
75. Swensen HN et al (1997) Design and operational evaluation of the traffic management advisor at the forth worth air route traffic control center. In: Proceeding of the air traffic management seminar, FAA/Eurocontrol
76. Swierstra S, Green S (2003) Common trajectory prediction capability for decision support tools. In: Proceeding of the air traffic management seminar, FAA/Eurocontrol
77. Tomlin C, Pappas GJ, Sastry S (1998) Conflict resolution for air traffic management: a study in multi-agent hybrid systems. IEEE Trans Automat Control 43:509–521
78. Vink A (1997) Eatchip medium term conflict detection: part 1 eatchip context. In: Proceeding of the air traffic management seminar, FAA/Eurocontrol
79. Weiss R (1974) The application of implicit Runge-Kutta and collocation methods to boundary value problems. Math Comput 28:449–464
80. Williams P (2004) Application of pseudospectral methods for receding horizon control. J Guid Control Dyn 27:310–314
81. Yan H, Fahroo F, Ross IM (2002) Real-time computation of neighboring optimal control laws. In: AIAA guidance, navigation, and control conference and exhibit. AIAA
82. Zhao Y (2011) Efficient and robust aircraft landing trajectory optimization. Ph.D. thesis, School of Aerospace Engineering, Georgia Institute of Technology
83. Zhao Y, Tsiotras P (2010) Density functions for mesh refinement in numerical optimal control. J Guid Control Dyn 34(1):271–277
84. Zhao Y, Tsiotras P (2010) Time-optimal parameterization of geometric path for fixed-wing aircraft. In: Infotech@Aerospace. AIAA, Atlanta
85. Zhao Y, Tsiotras P (2011) Stable receding horizon trajectory control for complex environment. In: American control conference (ACC), San Francisco, CA