ENRI Int. Workshop on ATM/CNS. Tokyo, Japan (EIWAC2013)

[EN-023] Mathematical Models for Aircraft Trajectory Design : A Survey.

+ D. Delahaye S. Puechmorel * E. Feron P. Tsiotras **

* Applied Math Lab (MAIAA)
Ecole Nationale Aviation Civile(ENAC)
Toulouse, France
delahaye@recherche.enac.fr

** School of Aerospace Engineering
Georgia Institute of Technology
Atlanta, USA
feron@gatech.edu

Abstract Air traffic management ensure 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 Air traffic conflict resolution, genetic algorithm, B-Spline approximation

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 guaranty 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 [35, 22, 2],trial planning [7], controller advisories for metering and sequencing [28, 55],traffic load forecasting [19, 26], weather impact assessment [25, 27, 24]. 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 (eg [56, 63, 49, 61, 62]). 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 many areas of ATM where trajectories are the main objects that have to be manipulate.

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

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 the fact that 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 [47]. 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 differentiable (class C^1) so that the length of a trajectory $\gamma[a, b] \to E$ is well defined as:

$$l(\gamma) = \int_a^b \|\gamma'(t)\| dt$$

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 arclength. Let $\gamma(a, b) \to E$ be such a curve. Defining the mapping $s(a, b) \to (0, l(\gamma))$ by:

$$s(t) = \int_a^t \|\gamma'(t)\| dt$$

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:

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

In the following, we will simply write $\gamma(s), s \in (0, l(\gamma))$ for a curve parametrized by arclength, 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 $\gamma[a,b] \rightarrow E$ has an associated curve, namely $\gamma(a,b) \rightarrow E$. 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 arclength parametrization allows to define very important geometrical quantities when $E = \mathbb{R}^3$.

Definition 1. Let:

$$\gamma(0,l) \rightarrow \mathbb{R}^3$$

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

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

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

Definition 2. Let:

$$\gamma(0,l) \rightarrow \mathbb{R}^3$$

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

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

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

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

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 arclength, the unit normal vector $\nu(s)$ is defined at all biregular points by :

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

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 :

$$\beta(s) = T(s).\nu(s)$$

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 arclength :

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

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 the figure 1.

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

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

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

3.1 Straight line segments

One of the easiest way to design trajectory is to use waypoints connected by straight lines (see figure 2. This easy principle ensure 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 waypoints 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 waypoints.

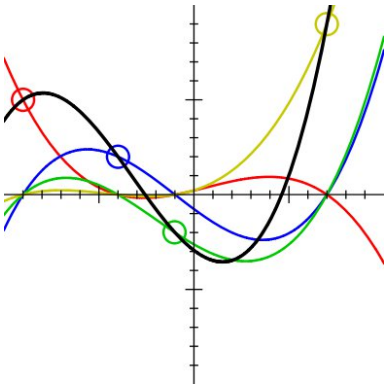

Figure 3 $L_n(x)$ is represented by the black curve. The others curves are the polynomials $l_i(x)$.

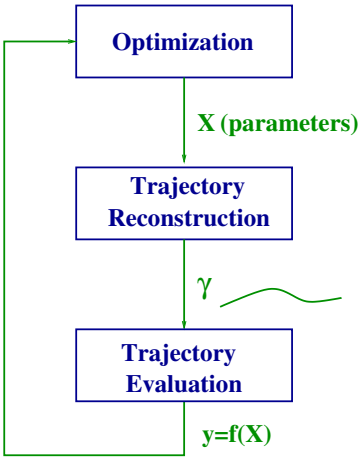

Figure 1 .The oprimization process control the X vector in order to build a trajectory γ for evaluation.

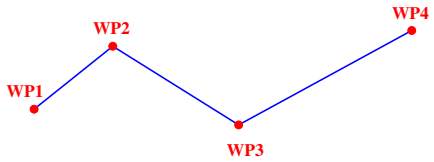

Figure 2 Trajectory defined by four waypoints connected by straight lines.

3.2 Lagrange interpolation

Given $n+1$ real numbers y_i,$0 \leq i \leq n$, and $n+1$ distinct real numbers $x_0 < x_1 < ... < x_n$, *Lagrange polynomial [43] of degree n* ($L_n(x)$) associated with $\{x_i\}$ and $\{y_i\}$ is a polynomial of degree n solving the interpolation problem :

$$p_n(x_i) = y_i, \quad 0 \leq i \leq n$$

$$L_n(x) = \sum_{i=0}^{n} f(x_i) l_i(x)$$

where

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

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

3.3 Hermite interpolation

Hermite interpolation [6] generalizes Lagrange interpolation by fitting a polynomial 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.

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

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

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

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

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 $n1$.

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 on figure 4 for which Lagrange interpolation has been used.

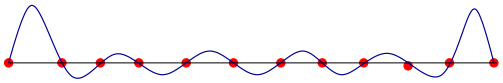

Figure 4 Lagrange interpolation result for a set of aligned points.

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.

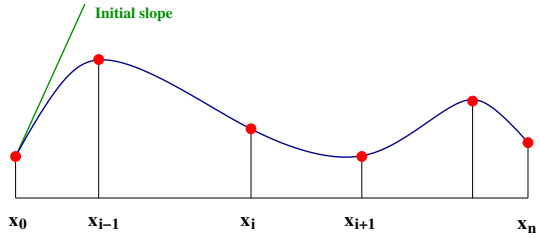

Figure 6 Piecewise quadratic interpolation.

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 $x_0 < x_1 < ... < x_n$, we consider the n linear curves $l_i(x) = a_i x + b_i$ on the intervals $[x_i, x_{i+1}]$ for $i = 0, ...n-1$.

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

$$y_i = a_i x_i + b_i x_i \;\; y_{i+1} = a_i x_{i+1} + b_i x_{i+1}$$

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

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

Then,

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

An example of such a linear piecewise interpolation is given on figure 5

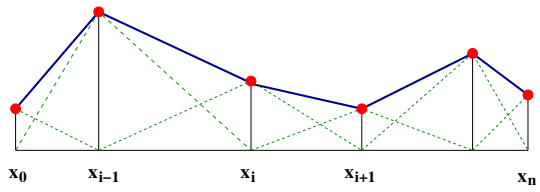

Figure 5 Piecewise linear interpolation.

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 $i = 0, ...n-$ 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 on figure 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.

3.6 Piecewise cubic interpolation

This interpolation is also called Hermite cubic interpolation [37]. For this interpolation :

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

and we have the following constraints :

$$\begin{array}{cc} C_i(x_i) = y_i & C_i(x_{i+1}) = y_{i+1} \\ C_i'(x_i) = y_i' = \frac{y_{i+1}-y_{i-1}}{x_{i+1}-x_{i-1}} & C_i'(x_{i+1}) = y_{i+1}' = \frac{y_{i+2}-y_i}{x_{i+2}-x_i} \end{array}$$

An example of piecewise cubic interpolation is given on figure 7.

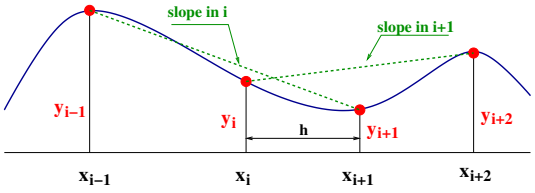

Figure 7 Piecewise cubic interpolation.

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 :

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

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

3.7 Cubic Spline Interpolation

This method has been developed by General Motor in 1964 [15]. For this piecewise interpolation $psi_i(x) = S_i(x)$ with the following constraints :

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

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

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

where

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

An example of such interpolation is given on figure 8.

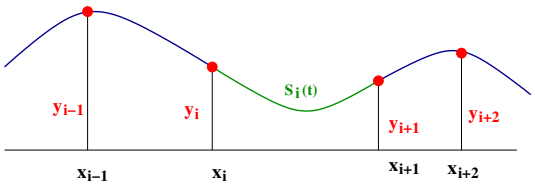

Figure 8 Cubic Spline Interpolation.

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.

3.8 Bézier Approximation Curve

Bézier curves[30] 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 [31], a numerically stable method to evaluate Bézier curves. A Bézier curve is defined by a set of control points $\vec{P}_0$ through $\vec{P}_n$, 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 $\vec{P}_0$ and $\vec{P}_1$, a linear Bézier curve $\vec{B}(t)$ is simply a straight line between those two points (see figure 9). The curve is given by :

$$\vec{B}(t) = \vec{P}_0 + t(\vec{P}_1 - \vec{P}_0) = (1-t)\vec{P}_0 + t\vec{P}_1 \ , \ t \in [0,1]$$

Figure 9 Bézier Curve with 2 points.

With four points ($\vec{P}_0$, $\vec{P}_1$, $\vec{P}_2$, $\vec{P}_3$), a Bézier curve of degree three can be built. The curve starts at $\vec{P}_0$ going towards $\vec{P}_1$ and arrives at $\vec{P}_3$ coming from the direction of $\vec{P}_2$. Usually, it will not pass through $\vec{P}_1$ or $\vec{P}_2$; these points are only there to provide directional information (see figure **??**).

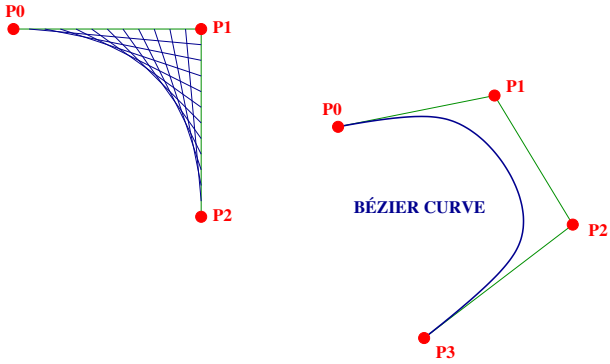

Figure 10 Cubic Bézier curve.

Properties

- The polygon formed by connecting the Bézier points with lines, starting with $\vec{P}_0$ and finishing with $\vec{P}_n$, is called the Bézier polygon (or control polygon).
- The convex hull of the Bézier polygon contains the 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 :

$$\vec{B}(t) = (1-t)^3\vec{P}_0 + 3(1-t)^2 t\vec{P}_1 + 3(1-t)t^2\vec{P}_2 + t^3\vec{P}_3 \ , \ t \in [0,1].$$

$$\vec{B}(t) = \sum_{i=0}^{n} b_{i,n}(t)\vec{P}_i, \quad t \in [0,1]$$

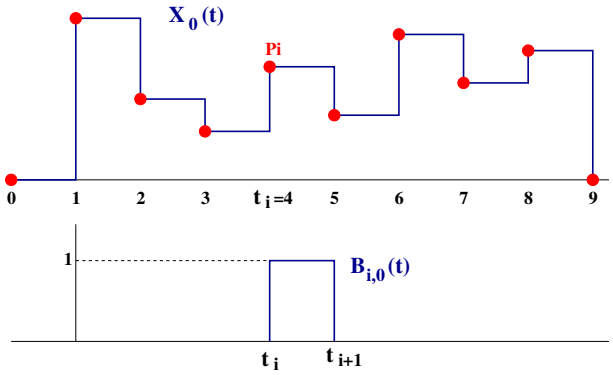

Figure 11 Uniform B-Splines of Degree Zero

where the polynomials

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

are known as Bernstein basis polynomials of degree n. So, if there are many points, one has to manipulate polynoms 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 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 $\vec{T} = \{t_0, t_1, ..., t_n\}$ with $t_0 \leq t_1 \leq, ..., \leq t_n$ and n points $\vec{P}_i$. One want to build a curve $\vec{X}_0(t)$ such that :

$$\vec{X}_0(t_i) = \vec{P}_i$$

$$\Rightarrow \vec{X}_0(t) = \vec{P}_i \ \forall t \in [t_i, t_{i+1}].$$

$$\vec{X}_0(t) = \sum_i B_{i,0}(t).\vec{P}_i$$

where

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

The shape of the $\vec{X}_0(t)$ function in one dimension is given on figure 11.

3.9.2 Uniform B-Splines of Degree One

We are searching for a piecewise linear approximation $\vec{X}_1(t)$ for which :

$$\vec{X}_1(t) = \left(1 - \frac{t - t_i}{t_{i+1} - t_i}\right)\vec{P}_{i-1} + \left(1 - \frac{t - t_i}{t_{i+1} - t_i}\right)\vec{P}_i \quad \forall t \in [t_i, t_{i+1}]$$

One can write $\vec{X}_1(t)$:

$$\vec{X}_1(t) = \sum_i B_{i,1}(t).\vec{P}_i$$

where

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

The shape of the $\vec{X}_1(t)$ function in one dimension is given on figure 12.

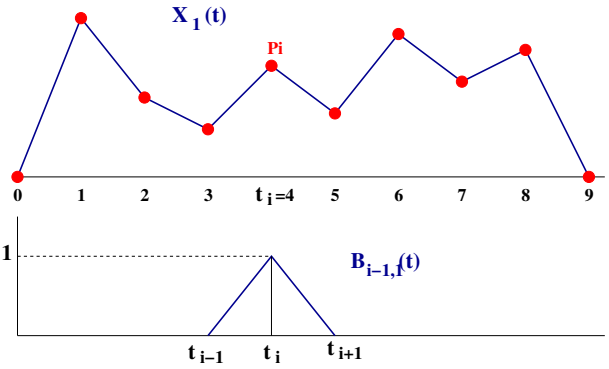

Figure 12 Uniform B-Splines of Degree One

3.9.3 *Uniform B-Splines of Degree Three*

Those B-Splines have been developed at Boeing in the 70s 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 :

$$\vec{X}_3(t) = \sum_{i=0}^{n} B_{i,3}(t).\vec{P}_i \;\; 3 \leq t \leq n+1$$

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

$$\vec{X}_3(t) = \sum_{i=j-3}^{j} P_i.B_{i,3}(t) \;\; t \in [j, j+1], \;\; 3 \leq j \leq n$$

When a single control point $\vec{P}_i$ is moved, only the portion of the curve $\vec{X}_3(t)$ with $t_i < t < t_{i+4}$ is changed $\Rightarrow$ local control. 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 $\sum_i B_i(t) = 1$ for $3 \leq t \leq n+1$
- The function $\vec{X}_i(t)$ are of degree 3 for any set of control points

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

Those basis functions are shown on figure 13.

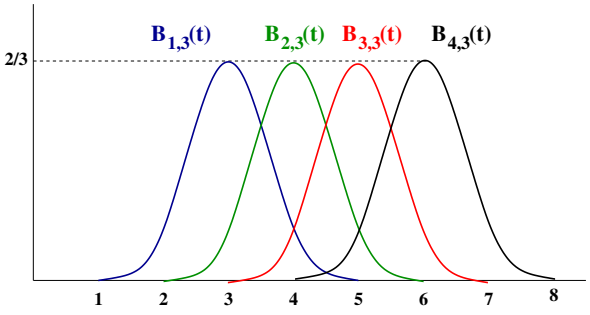

Figure 13 Order 3 basis function

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.

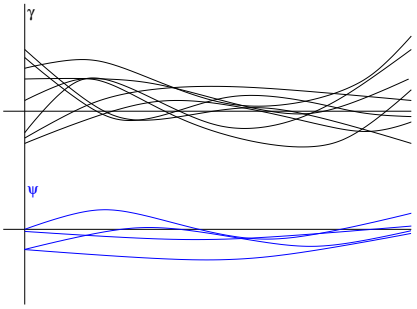

Figure 14 The black trajectories represent registered samples for which 4 principal components are extracted (in this artificial example) for minimum error reconstruction process.

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

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

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 PCA can be found in the reference [3].

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 figure 15) (past flown trajectories may be considered) :

$$\gamma_1, \gamma_2$$

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

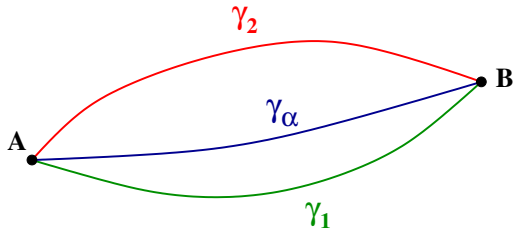

Figure 15 One new trajectory is built by using a weighted sum of two reference trajectories.

$$\gamma_\alpha = (1-\alpha)\gamma_1 + \alpha\gamma_2$$

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

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 very high frequencies tend to propagate along lines that are geodetic with respect to the metric that yields as the length of a curve the propagation time. 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 transpose the previous physical model by assuming a velocity directly related to the criterion to be optimized. As an example, congestion will be taken into account by velocity reduction, so that paths crossing congested areas will be penalized. Dependind 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 [68] is a part of the more general methods called *Level Set* [52]. 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 trafic 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 :

$$|\nabla T(x)| = \frac{1}{F(x)}, \qquad F(x) > 0, \qquad \text{et} \qquad T(x_{\text{initial}}) = 0 \tag{1}$$

where $x \in \mathbb{R}^2$ represents the position in space, $T \in \mathbb{R}$ the minimum time and $F \in \mathbb{R}$ 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 aircrafts from driving through these areas. We just need to reduce the propagation speed. Thus, the time is increased proportionnaly 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 [68] that is a geodetic line.

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 [69], 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 at located no farther than the anistropy ratio time 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 [54], 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 for Air Traffic Management

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* [32]:

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

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 inhomogeneous 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. 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. 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).

We have applied successfully this algorithm to the aircraft conflict resolution problem. To address this conflict resolution problem, aircraft are sequentially resolved using LPA. 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. The trajectory assigned to an aircraft in each resolution step must avoid other aircraft trajectories that are considered as fixed constraints (known data). In our case, the aircraft ordering is chosen at random. 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.). LPA has been applied on a day of traffic (August 12, 2008) with about 8000 flights. The initial trajectories (before conflict resolution) induce a total number of clusters, with some aircraft in real conflict, equal to 3344. 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 1501 trajectories have been modified to reach such a conflict-free planning. In many cases, the new computed trajectories are 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.

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 [20], 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 [44].

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

$$J(x, u) = \int_{t_0}^{t_f} L(x(t), u(t))\,\mathrm{d}t, \tag{2}$$

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

$$\begin{aligned} &\dot{x}(t) - f(x(t), u(t)) = 0, \\ &C(x(t), u(t)) \leq 0. \end{aligned} \tag{3}$$

Optimal control has its roots in the theory of calculus of variations, which originated in the 17th century by Fermat, Newton, Liebniz, and the Bernoullis, and was subsequently further developed by Lagrange, Weirstrass, Legendre, Clebsch and Jacobi and others in the 18th and 19th centuries [29]. Calculus of variations deals with the problem of minimizing (2) 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 (3) or algebraic constraints such as (3). It was not until the middle of the 20th century when the Soviet mathematician L. S. Pontryagin developed a complete theory that could handle constraints such as (3) and (3). Simply put, Pontryagin's celebrated Maximum Principle [53] states that the optimal control for the solution of the problem (2)-(3) is given as the pointwise minimum of the so-called Hamiltonian function, that is,

$$u_{\text{opt}} = \text{argmin}_{u \in U} H(t, x, \lambda, u), \tag{4}$$

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

$$\dot{\lambda}(t) = -\frac{\partial H}{\partial x}(x(t), \lambda(t), u(t)). \tag{5}$$

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 [50]. 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 (2) but, rather, by computing potential optimizers by solving the corresponding necessary optimality conditions (4)-(5).

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 (2) subject to (3)-(3).

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 [60, 77, 23, 36]. 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 [12], RIOTS [67], DIDO [59], PSOPT [8], GPOPS [57], MTOA [40] and DENMRA [81] among many others. A recent survey of numerical optimal control techniques for trajectory optimization can be found in [10].

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 [11, 14, 67, 13, 41, 33].

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 [71, 46, 42, 79]. 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 [51, 9, 78]. 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 [83]. 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 [82, 5, 80].

6 Trajectory Optimization Methods in ATM

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 [34, 72, 70], and the minimum-time, three-dimensional aircraft trajectory optimization problem considered in [66]. 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 [1].

Some of these results have been extended to commercial airline operations. For example, the work of [18] 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. Reference [38] performed a sensitivity anal-

ysis of trajectory prediction for ATM. The aircraft trajectory synthesis problem is studied in [73] to provide some basic tools for air traffic automation. Somewhat related is the recent work of Sridhar [74], 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 Ref. [39]. 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 [45, 76].

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 [16, 17, 48], and emergency landing during loss of thrust [75]. 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 [4], 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 [80] 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

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

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

$$\min_{T} \quad \int_{s_0}^{s_f} \frac{\mathrm{d}s}{\sqrt{2E(s)}} \tag{6a}$$

$$\text{subject to} \quad E'(s) = \frac{T(s)}{m} - D(E(s), s) - g\sin\gamma(s), \tag{6b}$$

$$\underline{g}_w(s) \leq E(s) \leq \overline{g}_w(s), \tag{6c}$$
$$T_{\min} \leq T(s) \leq T_{\max}, \tag{6d}$$

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

$$\min_{T} \quad \int_{s_0}^{s_f} T(s)\,\mathrm{d}s, \tag{7a}$$
$$\text{subject to} \quad E'(s) = \frac{T(s)}{m} - D(E(s), s) - g\sin\gamma(s), \tag{7b}$$
$$t'(s) = \frac{1}{\sqrt{2E(s)}}, \tag{7c}$$
$$\underline{g}_w(s) \leq E(s) \leq \overline{g}_w(s) \tag{7d}$$
$$T_{\min}(s) \leq T(s) \leq T_{\max}(s), \tag{7e}$$

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. 16. 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 [83], 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 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.

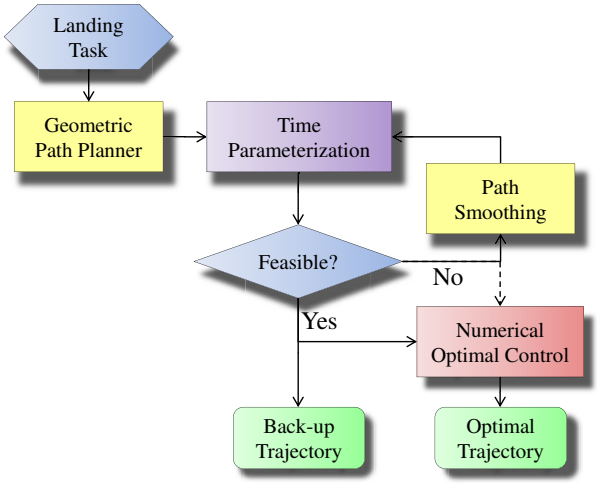

Figure 16 Schematic of landing trajectory optimization.

7 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 [65].

The overall proposed architecture is shown in Fig. 17. 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 (eg 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

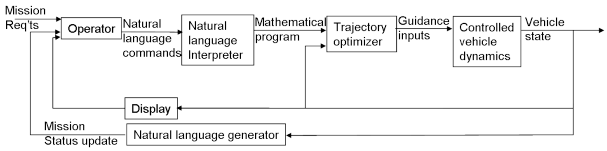

Figure 17 Basic Autonomous Mission Management Loop for future Aircraft and UAVs

formulates the vehicles's goals (eg flight plans) using natural language. A natural language interpreter and task scheduler transforms the operator's requirements into tractable mathematical optimization programs that may be executed by the 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.

7.1 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 the 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.

7.2 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 trafic 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.

7.3 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 latter 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

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

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

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 x_T in the planning horizon to the goal state $\mathbf{x}_f$. The sets $\mathcal{X}_0$ and $\mathcal{U}_0$ 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 $(x_i, y_i) \in \mathcal{D}_0$ and $(x_i, y_i) \notin \mathcal{O}_0$ capture the requirement that the planned trajectory points should lie inside the known region $\mathcal{D}_0$, but outside the obstacles $\mathcal{O}_0$ 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 [64], 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 [64], 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 $\mathcal{D}_0$ 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.

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 and Ordered upwind algorithm. An application to real problem coming from air traffic management has also been given with the description of 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. Finally, the fifth part has presented an original path planning techniques mixing natural language processing and mathematical programming.

References

[1] *Control and dynamic systems: Advances in theory and applications*, Academic Press, 1973.

[2] Vink A., *Eatchip medium term conflict detection: Part 1 eatchip context*, Proceeding of the Air Traffic Management Seminar, FAA/Eurocontrol, 1997.

[3] H. Abdi and L.J. Williams, *Principal component analysis*, Interdisciplinary Reviews: Computational Statistics **2,** (2010), 433–459.

[4] E.M Atkins, I.A Portillo, and M.J Strube, *Emergency flight planning applied to total loss of thrust*, Journal of Guidance, Control, and Dynamics **43** (2006), no. 4, 1205–1216.

[5] E. Bakolas, Y. Zhao, and P. Tsiotras, *Initial guess generation for aircraft landing trajectory optimization*, AIAA Guidance, Navigation, and Control Conference, AIAA, 2011.

[6] R.H Bartels, J.C Beatty, and B.A Barskyn, *An introduction to splines for use in computer graphics and geometric modeling*, Computer graphics, Morgan Kaufmann, 1998.

[7] McNally B.D., Bach R.E., and Chan W., *Field test evaluation of the ctas conflict prediction and trial planning capability*, Proceeding of the AIAA-1998-4480 AIAA GNC Conference, AIAA GNC, 1998.

[8] V.M. Becerra, *Psopt optimal control solver user manual*, Tech. report, 2011.

[9] J Bellingham, Y Kuwata, and J How, *Stable receding horizon trajectory control for complex environment*, AIAA Guidance, Navigation, and Control Conference and Exhibit, AIAA, 2003.

[10] J.T. Betts, *Survey of numerical methods for trajectory optimization*, Journal of Guidance, Control, and Dynamics **21** (1998), no. 2, 193–207.

[11] J.T. Betts, N. Biehn, S.L. Campbell, and W.P. Huffman, *Compensating for order variation in mesh refinement for direct transcription methods*, Journal of Computational and Applied Mathematics **125** (2000), 147–158.

[12] J.T. Betts and W.P Huffman, *Sparse optimal control software SOCS*, Tech. report, Mathematics and Engineering Analysis Technical Document MEALR-085, Boeing Information and Support Services,The Boeing Compan, 1997.

[13] J.T Betts and W.P. Huffman, *Mesh refinement in direct transcription methods for optimal control*, Optimal Control Applications and Methods **19** (1998), no. 1, 1–21.

[14] T. Binder, L. Blank, W. Dahmen, , and W Marquardt, *Grid refinement in multiscale dynamic optimization*, Tech. report, RWTH Aachen, 2000.

[15] Birkhoff and C de Boor, *Piecewise polynomial interpolation and approximation*, Proceeding of the General Motors Symposium of 1964, General Motors, 1964.

[16] R Bulirsch, F. Montrone, and H.J. Pesch, *Abort landing in the presence of windshear as a minimax optimal control problem Part I: Necessary conditions*, Journal of Optimization Theory and Applications **70** (1991), no. 1, 1–23.

[17] ———, *Abort landing in the presence of windshear as a minimax optimal control problem Part II: Multiple shooting and homotopy*, Journal of Optimization Theory and Applications **70** (1991), no. 2, 223–254.

[18] J.W. Burrows, *Fuel-optimal aircraft trajectories with fixed arrival times*, Journal of Guidance, Control, and Dynamics **6** (1983), no. 1, 14–19.

[19] Meckiff C., Chone R., and Nicolaon J.P., *The tactical load smoother for multi-sector planning*, Proceeding of the Air Traffic Management Seminar, FAA/Eurocontrol, 1998.

[20] A Chakravarty, *Four-dimensional fuel-optimal guidance in the presence of winds*, Journal of Guidance, Control, and Dynamics **8** (1985), no. 1, 16–22.

[21] C de Boor, *A practical guide to splines*, Springer-Verlag, 1978.

[22] Brudnicki D.J. and McFarland A.L., *User request evaluation tool (uret) conflict probe performance and benefits assessment*, Proceeding of the Air Traffic Management Seminar, FAA/Eurocontrol, 1997.

[23] P.J. Enright and B.A. Conway, *Discrete approximation to optimal trajectories using direct transcription and nonlinear programming*, Journal of Guidance, Control, and Dynamics **15** (1992), 994–1002.

[24] Evans J. et al, *Reducing severe weather delays in congested airspace with weather support for tactical air traffic management*, Proceeding of the Air Traffic Management Seminar, FAA/Eurocontrol, 2003.

[25] Kirk D.B. et al, *Problem analysis resolution and ranking (parr) development and assessment*, Proceeding of the Air Traffic Management Seminar, FAA/Eurocontrol, 2001.

[26] Masalonis A. et al, *Using probabilistic demand prediction for traffic flow management decision support*, Proceeding of the AIAA-2004-4875 AIAA GNC Conference, AIAA GNC, 2004.

[27] Sud V. et al, *Air traffic flow management collaborative routing coordination tools*, Proceeding of the AIAA-2001-4112 AIAA GNC Conference, AIAA GNC, 2001.

[28] Swensen H.N. et al, *Design and operational evaluation of the traffic management advisor at the forth worth air route traffic control center*, Proceeding of the Air Traffic Management Seminar, FAA/Eurocontrol, 1997.

[29] G.M. Ewing, *Calculus of variations with applications*, Norton, New York 1969, reprinted by Dover, 1985.

[30] Farin, Gerald, and Hansfordand Dianne, *The essentials of cagd*, A K Peters, Ltd, 2000.

[31] G Farin, *Curves and surfaces for computer aided geometric design. a practical guide*, Academic Press, 1993.

[32] Douglas C. Giancoli, *Physics for scientists and engineers with modern physics*, second ed., Prentice-Hall, 1989.

[33] Q Gong, F. Fahroo, and I.M. Ross, *Spectral algorithm for pseudospectral methods in optimal control*, Journal of Guidance, Control, and Dynamics **31** (2008), no. 3, 460–471.

[34] W. Grimm, K. Well, and H. Oberle, *Periodic control for minimum-fuel aircraft trajectories*, Journal of Guidance, Control, and Dynamics **9** (1986), no. 2, 169–174.

[35] Erzberger H., Paielli R.A., Isaacson D.R., and Eshowl M.M., *Conflict detection in the presence of prediction error*, Proceeding of the Air Traffic Management Seminar, FAA/Eurocontrol, 1997.

[36] C.R. Hargraves and S.W. Paris, *Direct trajectory optimization using nonlinear programming and collocation*, Journal of Guidance, Control, and Dynamics **15** (1992), 994–1002.

[37] T Hearth, M, *Scientific computing, an introductory survey*, Computer graphics, McGraw-Hill, 2002.

[38] M.R. Jackson, Y. Zhao, and R.A. Slattery, *Sensitivity of trajectory prediction in air traffic management*, Journal of Guidance, Control, and Dynamics **22** (1999), no. 2, 219–228.

[39] M Jacobson and U.T. Ringertz, *Airspace constraints in aircraft emission trajectory optimization*, Journal of Aircraft **47** (2010), 1256–1265.

[40] S Jain and P. Tsiotras, *Multiresolution-based direct trajectory optimization*, Journal of Guidance, Control, and Dynamics **31** (2008), no. 5, 1424–1436.

[41] S. Jain and P Tsiotras, *Trajectory optimization using multiresolution techniques*, Journal of Guidance, Control, and Dynamics **31** (2008), no. 5, 1424–1436.

[42] M.R. Jardin and A.E. Bryson, *Neighboring optimal aircraft guidance in winds*, Journal of Guidance, Control, and Dynamics **24** (2001), 710–715.

[43] H. Jeffreys and B. S. Jeffreys, *Methods of mathematical physics*, Cambridge University Press, 1988.

[44] S.M LaValle, *Planning algorithms*, Cambridge University Press, 2006.

[45] W. Liu and I Hwang, *Probabilistic aircraft midair conflict resolution using stochastic optimal control*, IEEE Intelligent Transportation Systems Transactions and Magazine (2012).

[46] P. LU, *Regulation about time-varying trajectories: Precision entry guidance illustrated*, Journal of Guidance, Control, and Dynamics **22** (1999), 784–790.

[47] Berger M. and Gostiaux B., *Differential geometry: Manifolds, curves and surfaces*, Springer-Verlag, 1988.

[48] A. Miele, *Optimal trajectories and guidance trajectories for aircraft flight through windshears*, Proceedings of the 29th IEEE Conference on Decision and Control, IEEE, 1990.

[49] S. Mondoloni, Pagli S.M., and Green S., *Trajectory modeling accuracy for air traffic management decision support tools*, Proceeding of the ICAS Conference, ICAS, 2002.

[50] H.J. Oberle and W. Grimm, *BNDSCO - A program for the numerical solution of optimal control problems*, Tech. report, Institute for Flight System Dynamics, German Aerospace Research Establishment Oberpfaffenhofen, 1989.

[51] T. Ohtsuka, *Quasi-Newton-type continuation method for nonlinear receding horizon control*, Journal of Guidance, Control, and Dynamics **24** (2002), 685–692.

[52] S. Osher and J.A. Sethian, *Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations*, Journal of computational physics **79** (1988), no. 1, 12–49.

[53] L.S. Pontryagin, V.G. Boltyanski, R.V. Gamkrelidze, and E.F. Mischenko, *The mathematical theory of optimal processes*, New York, NY: Interscience, 1962.

[54] C. Ptrs, Y. Pailhas, P. Patron, Y. Petillot, J. Evans, and D. Lane, *Path planning for autonomous underwater vehicles*, IEEE Transactions on Robotics, **23** (2007), no. 2, 331–341.

[55] Coppenbarger R., Lanier R., Sweet D., and Dorsky S., *Design and development of the en-route descent advisor (eda) for conflict-free arrival metering*, Proceeding of the AIAA-2004-4875 AIAA GNC Conference, AIAA GNC, 2004.

[56] Slattery R. and Zhao Y., *Trajectory synthesis for air traffic automation*, AIAA Journal Guidance Control and Dynamics **20-2,** (1997), 232–238.

[57] A.V Rao, D Benson, and G.T. Huntington, *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, 2011.

[58] E. Roche, *Parsing with finite-state transducers*, Finite-State Language Processing (E. Roche and Y. Schabes, eds.), MIT Press, 1997.

[59] I.M. Ross, *'user's manual for DIDO: A MATLAB application package for solving optimal control problems*, Tech. report, Naval Postgraduate School, 2005.

[60] R.D. Russell and L.F Shampine, *A collocation method for boundary value problems*, Numerische Mathematik **19** (1972), 13–36.

[61] H.F. Ryan, Paglione M., and Green S., *Review of trajectory accuracy methodology and comparison of error measure metrics*, Proceding of the AIAA-2004-4787 AIAA GNC Conference, AIAA GNC, 2004.

[62] Mondoloni S. and Bayraktuta I., *Impact of factors, conditions and metrics on trajectory prediction accuracy*, Proceeding of the Air Traffic Management Seminar, FAA/Eurocontrol, 2005.

[63] Swierstra S. and Green S., *Common trajectory prediction capability for decision support tools*, Proceeding of the Air Traffic Management Seminar, FAA/Eurocontrol, 2003.

[64] T. Schouwenaars, J. How, and E. Feron, *Receding horizon path planning with implicit safety guarantees*, American Control Conference (Boston, MA), June 2004, pp. 5576–5581.

[65] T. Schouwenaars, M. Valenti, E. Feron, J. How, and E. Roche, *Linear programming and language processing for human/unmanned-aerial-vehicle team missions, journal of guidance, control, and dynamics*, AIAA Journal of Guidance, Control, and Dynamics **29** (2006), no. 2, 303–313.

[66] R.L. Schultz, *Three-dimensional trajectory optimization for aircraft*, Journal of Guidance, Control, and Dynamics **13** (1990), no. 6, 936–943.

[67] A Schwartz, *Theory and implementation of numerical methods based on runge-kutta integration for solving optimal control problems*, Ph.D. thesis, Université Montpellier II (France), 1996.

[68] J.A. Sethian, *Fast marching methods*, SIAM review **41** (1999), no. 2, 199–235.

[69] J.A. Sethian and A. Vladimirsky, *Ordered upwind methods for static Hamilton-Jacobi equations: Theory and algorithms*, SIAM Journal on Numerical Analysis **41** (2003), 325–363.

[70] H. Seywald, *Long flight-time range-optimal aircraft trajectories*, Journal of Guidance, Control, and Dynamics **19** (1994), no. 1, 242–244.

[71] H. Seywald and E.M. Cliff, *Neighboring optimal control based feedback law for the advanced launch system*, Journal of Guidance, Control, and Dynamics **17** (1994), 1154–1162.

[72] H. Seywald, E.M. Cliff, and K. Well, *Range optimal trajectories for an aircraft flying in the vertical plane*, Journal of Guidance, Control, and Dynamics **17** (1994), no. 2, 389–398.

[73] R.A. Slattery and Y. Zhao, *Trajectory synthesis for air traffic automation*, Journal of Guidance, Control, and Dynamics **20** (1997), no. 2, 232–238.

[74] B Sridhar and N.Y. Ng, H.K.and Chen, *Aircraft trajectory optimization and contrails avoidance in the presence of winds*, Journal of Guidance, Control, and Dynamics **34** (2011), 1577–1583.

[75] M.J. Strube, R.M. Sanner, and E.M Atkins, *Dynamic flight guidance recalibration after actuator failure*, AIAA 1st Intelligent Systems Technical Conference, AIAA, 2004.

[76] C Tomlin, G.J. Pappas, and S. Sastry, *Conflict resolution for air traffic management: A study in multi-agent hybrid systems*, IEEE Trans. on Automatic Control **43** (1998).

[77] R Weiss, *The application of implicit Runge-Kutta and collocation methods to boundary value problems*, Mathematics of Computation **28** (1974), 449–464.

[78] P. Williams, *Application of pseudospectral methods for receding horizon control*, Journal of Guidance, Control, and Dynamics **27** (2004), 310–314.

[79] H. Yan, F Fahroo, and I.M. Ross, *Real-time computation of neighboring optimal control laws*, AIAA Guidance, Navigation, and Control Conference and Exhibit, AIAA, 2002.

[80] Y. Zhao, *Efficient and robust aircraft landing trajectory optimization*, Ph.D. thesis, School of Aerospace Engineering, Georgia Institute of Technology, 2011.

[81] Y. Zhao and P. Tsiotras, *Density functions for mesh refinement in numerical optimal control*, Journal of Guidance, Control, and Dynamics **34** (2010), no. 1, 271–277.

[82] ———, *Time-optimal parameterization of geometric path for fixed-wing aircraft*, Infotech@Aerospace, AIAA, 2010.

[83] ———, *Stable receding horizon trajectory control for complex environment*, American Control Conference, ACC, 2011.

Acknowledgements

Copyright