A two-stage route optimization algorithm for light aircraft transport systems

Sharif Azadeh, Sh; Bierlaire, M.; Maknoon, M. Y.

DOI
10.1016/j.trc.2019.01.028

Publication date
2019

Document Version
Final published version

Published in
Transportation Research Part C: Emerging Technologies

Transportation Research Part C 100 (2019) 259–273

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

A two-stage route optimization algorithm for light aircraft transport systems

Sh. Sharif Azadeh[a,b,*], M. Bierlaire[b], M.Y. Maknoon[b,c]

[a] Econometric Department, Erasmus University Rotterdam, P.O. Box 1738, 3000 DR Rotterdam, Netherlands
[b] School of Architecture, Civil and Environmental Engineering, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
[c] Faculty of Technology, Policy, and Management, Delft University of Technology, Jaffalaan 5, 2628 BX Delft, Netherlands

ARTICLE INFO

Keywords:
Route optimization
Light aircraft
MILP
Heuristic
Obstacle avoidance

ABSTRACT

This paper presents a route optimization algorithm for light aircraft operating under visual flight rules. The problem aims at finding a minimum-duration, collision-free route in three spatial dimensions with possible aircraft maneuvers. The optimal route takes into account the aircraft kinodynamic characteristics and its interaction with external wind. A data processing approach is presented to recast the flying environment as a series of polyhedrons based on which a mixed-integer linear model is formulated. A two-stage route optimization model is then introduced to solve real-life instances. Computational experiments depict the efficiency of this approach.

1. Introduction

Traffic congestion is increasingly becoming a global issue, with drivers spending nearly 50% of their driving time in traffic in some cities around the world. In 2016, the average San Francisco resident spent 230 h commuting between work and home meaning half a million hours of productivity lost every single day (Holden and Goel, 2016; Nelson and Rae, 2016). To tackle this problem, several companies such as Uber-Elevate have started to investigate opportunities to provide aircraft-based mobility within metropolitan areas (Vascik, 2017; Kreimeier et al., 2016).

Such flight services are based on point-to-point passenger transport for short-haul trips. These flights create a unique route planning environment which is different from commercial carriers where flyable routes are constrained by airways (i.e., aircraft route between two geographical locations) and governed by Air Traffic Control (ATC) (see, Barnhart and Smith, 2012; Bonami et al., 2013; Blanco et al., 2016; Tang and Mukherjee, 2000).

The light aircraft could not fly the same routes as commercial ones because they cannot handle going over 10,000 feet Cheng (1962). Therefore, the main challenge of using such light aircraft is to find an optimal route (based on minimizing fuel consumption, traveling time or distance). The flying route must be tested for flight safety and efficiency. First, the route must avoid collision with obstacles. Obstacles vary from physical barriers (e.g. mountains) to virtual ones (e.g. major airports, military area, nuclear plant, etc.). Second, it must respect aircraft limited dynamic capabilities (e.g. radius of turn, minimum/maximum speed). Third, this route needs to take into account the weather conditions to ensure flight safety.

In this paper, we present a three-dimensional route planning algorithm for light aircraft used for short-haul passenger transport. We formalize the problem and present a mixed-integer-linear optimization model. The optimization model aims to find the minimum-time, collision-free flying route. We introduce a two-stage heuristic approach that can efficiently solve real-life sized problems. In

* Corresponding author at: b Econometric Department, Erasmus University Rotterdam, P.O. Box 1738, 3000 DR Rotterdam, Netherlands.
E-mail addresses: sharifazadeh@ese.eur.nl (S. Sharif Azadeh), michel.Bierlaire@epfl.ch (M. Bierlaire), M.Y.Maknoon@tudelft.nl (M.Y. Maknoon).

https://doi.org/10.1016/j.trc.2019.01.028
Received 20 August 2018; Received in revised form 18 December 2018; Accepted 28 January 2019
Available online 03 February 2019
0968-090X/

Table 1
Overview of the current literature.

	Path plan	Trajectory plan	Our algorithm
Obstacle avoidance	Yes	Yes	Yes
Wind integration	No	No	Yes
Kinodynamic	No	Yes	Yes
Objective	Distance	Distance-travel time	Travel time
Dim (2D, 3D)	2D-3D	2D-3D	3D

stage one, called *path planning*, we discretize the three-dimensional space to create a truncated feasible (i.e., obstacle-free) corridor that contains the optimal trajectory. In stage two, called *trajectory planning*, we discretize time in the continuous space within the generated corridor in stage one. We solve the proposed mathematical model within this truncated space/feasible corridor. The output indicates a minimum-time route containing detailed speed, acceleration, and location profile of the flight from an origin to a destination while taking weather condition into account. We validate this approach for several case studies in central Europe.

The remainder of this paper is organized as follows: In Section 2, we review the related literature. We present our data processing approach to recasting the flying environment as a series of polyhedrons in Section 3. The mathematical model is presented in Section 4 followed by a two-stage route planning algorithm in Section 5. We present the computational results in Section 6 and concluding remarks in Section 7.

2. Related literature

Finding a “collision-free” and “aircraft dynamic compatible route” are two features of route planning for light aircraft. The research in this field can be classified into two groups of path and trajectory planning that have been extensively studied in the field of robotics as well as dynamic and controls. (See, Qian et al., 2017; Goerzen et al., 2010; Yang et al., 2016; Djokic et al., 2010; Wolek and Woolsey, 2017; LaValle, 2006; Prats et al., 2010; Soler et al., 2014.) Table 1 presents a summary of features embedded in the algorithms used in path and trajectory planning. In the last column of the table, we indicate the characteristics that our proposed algorithm possesses. In Sections 2.1 and 2.2, we explain in detail both path and trajectory planning and the algorithms used in the literature to tackle these problems. In Section 2.3, we review the related literature about wind integration in route planning.

2.1. Path planning

The focal point of the path planning problem is to find the minimum-distance collision-free flying direction in an environment where there are obstacles. Reif et al. proved that this problem is NP-hard in a continuous three-dimensional space (Reif, 1979).

All developed algorithms aim to find the path using two main approximation approaches. In the first approach, a continuous curve is approximated by a series of waypoints connected by links. The free space is modeled using graph and the shortest-path is found by using different search algorithms (see Vandapel et al., 2005; Ferguson et al., 2005; Sinopoli et al., 2001; Tsenkov et al., 2008 for cell decomposition approach and Choset and Burdick, 2000; Howlet et al., 2004; Sch et al., 2012; Wein et al., 2007; Schøler, 2012 for road-map method). In the second approach, the flyable environment is expressed by a set of repulsive functions. The flight direction is computed by using the gradient of these functions. (See Connolly et al., 1990; Ahuja and Chuang, 1997; Scherer et al., 2007; Chen et al., 2016.) Path planning algorithms are simple to implement and computationally efficient. However, they cannot guarantee the kinodynamic feasibility of the path nor do they integrate the weather condition in their search algorithm.

2.2. Trajectory design

Trajectory design problem assesses aircraft kinodynamic constraints when designing a collision-free path. There are three main groups of trajectory design approaches: path-smoothing, direct search, and mathematical models.

For an existing path, calculated with one of the path planning algorithms, a *smoothing* approach transforms the existing path into the kinodynamic feasible one (see, Boyle and Chamitoff, 1999; Judd and McLain, 2001; Suzuki et al., 2005; Bellingham, 2002).

The trajectory designed by path-smoothing approaches may greatly deviate from the optimal one. To overcome this issue, *search-base* approaches explicitly considers kinodynamic constraints. In the “rapidly-expanding random tree”, the search algorithm generates nodes in the free space and as long as they do not fall in (or on) an obstacle, it connects them to one another (LaValle, 1998; LaValle and Kuffner, 2001). For each node, the algorithm checks the closest neighboring ones to link them if the connection is obstacle-free and feasible in a kinodynamic manner (see Frazzoli et al., 2000, 2002; Adiyatov and Varol, 2013; Gammell et al., 2014; Redding et al., 2007; Darbari et al., 2017; Dever et al., 2004).

Finally, the last approach in trajectory design is focused on using mathematical models. Finding the minimum-cost trajectory can be modeled as a non-linear optimal control problem and solved with different numerical approaches (see, Chamseddine et al., 2012; Borrelli et al., 2006; Khuswendi et al., 2011; Ding et al., 2015; Richards et al., 2002; Geem et al., 2001). These models are not computationally applicable to solve real life cases. To address this issue, time discretization techniques are introduced to transform the non-linear problem into a mixed-integer linear one with additional constraints (see, Richards and How, 2002; Valente et al., 2013; Miller et al., 2011; Kuwata, 2003).

In general, these formulations have a large number of variables and constraints due to discretization. The Receding Horizon Algorithm (RHA) is a common remedy to overcome the computational burden of mixed-integer linear models. The main idea of RHA is to sequentially solve the mathematical model for a reduced time duration till reaching the final destination (Mettler and Kong, 2008; Bircher et al., 2016). In this approach, the objective function has two components. A component that minimizes the flight duration for the reduced time period and another component which guides the aircraft towards the final destination (a.k.a cost-to-go). The cost-to-go function is calculated based on the shortest-path on a visibility graph which ignores kinodynamic constraints as well as external wind. As a result, the cost-to-go function can divert the aircraft from the optimal trajectory (Mettler et al., 2010).

To sum up, the focus of trajectory design is on finding a kinodynamic compatible route. However, they fail to obtain the route with the minimum travel time for two main reasons. First, they assume that the aircraft flies at a constant speed at all times. Second, these methods do not take into account the weather conditions. The later is crucial as the travel time is computed based on the aircraft ground speed which is not realistic. In the next subsection, we introduce the algorithms that also consider wind speed and direction.

2.3. Travel time calculation

The optimal trajectory not only depends on the characteristics of the aircraft but also, on the choice of the objective function Kong et al. (2009). Objectives such as minimizing distance, travel time and energy consumption result in different trajectories. This observation is magnified when the weather conditions are integrated into the algorithm. For example, tailwinds reduce the travel time, while headwinds cause an increase. In a dynamic environment, route planning algorithm searches for the ways to potentially reduce the flying time.

Calculation of travel time is a focal research study in the domain of air traffic control. By relaxing kinodynamic compatibility and obstacle avoidance constraints, research studies aim to find the optimal horizontal trajectory (flying in a constant altitude) taking into account wind (see, Jardin and Bryson, 2001, 2012; Marchidan and Bakolas, 2015; Girardet et al., 2014; Marchidan and Bakolas, 2015; Bonami et al., 2013).

The minimum-time flying route of such light aircraft not only needs to be collision-free and kinodynamically compatible but also requires to explicitly consider the effect of wind. All the above-mentioned papers developed a route planning approach by taking into account part of the factors that are required to plan a route for the light aircraft (see Table 1). In this research, we introduce a holistic approach that consolidates all these factors in a single model benefiting from the abilities of each method to find a feasible route with minimum traveling time.

3. Problem description

We model the aircraft route as a point mass flying through a three-dimensional environment. The aircraft movement is constrained by velocity and acceleration boundaries approximating limited turn-rate. The interaction between the aircraft forces (e.g. weight, lift, thrust and drag) and its surrounding area are not considered in our optimization model.

To formalize the route planning, we use polygonal modeling to recast the three-dimensional environment with polyhedrons (noted by the set **M**). In fact, polygones are defined as two-dimensional projections of obstacles, (X, Y). Meanwhile, polyhedrons (representing the obstacles) are created based on the extension of these polygones in three dimensional space to certain level of altitude (or between two levels of specific altitude), (X, Y, Z). These polyhedrons are then classified into two groups of “free space” and “obstacle space” presented by the sets $r \in \mathcal{R}$ and $o \in O$, respectively ($\mathcal{M} = \mathcal{R} \cup O$).

The flying altitude is bounded below by mean sea level and above by 10,000 feet. Moreover, without loss of generality, we assume all polyhedrons are convex. We use an algorithm to transform non-convex polyhedrons into a series of convex ones (see Fernández et al., 2000). These polyhedrons are defined in Cartesian coordinates using the map projection technique proposed by Karney (2011). This projection keeps the shape of polyhedrons while being independent from the choice of origin and destination. Therefore, once polyhedrons are created, they can be used to find a flying route between various origins and destinations.

For a given origin-destination pair, our goal is to find the minimum-time flying route. This route avoids obstacles and must be compatible with the aircraft kinodynamic constraints. We calculate the route duration based on the aircraft ground speed.

All polyhedrons in the set **M** are formed by using three sources of data: restricted airspace, elevation, and wind. In the following sections, we explain our approach to recast our data sources into polyhedrons and explain in more details how this pre-processing phase is implemented. These polyhedrons are the basis of the model presented in Section 4.

3.1. Modeling wind data with polyhedrons

The purpose of modeling wind data is to discover dangerous flying zones and calculating the aircraft ground speed. We partition the atmosphere with cuboids, noted by $c \in C$, (i.e., a cuboid is a box-shaped solid object. It has six flat sides and all angles are right angles. And all of its faces are rectangles). Each cuboid has a non-overlapping horizontal boundary and an altitude range. We assume that inside a cuboid wind is homogeneous in terms of velocity and direction. The wind forecast has three elements: West, North and vertical. These elements are presented by vector $\overrightarrow{w_c}$ assigned to each cuboid. In this problem, we disregard the forecasting uncertainty and do not consider the evolution of wind over time. These assumptions are compatible with our framework as our focus is on short-haul trajectories (less than four hours).

We use Algorithm 2 to transform the wind forecast into polyhedrons added to the set **M**. The algorithm first partitions wind

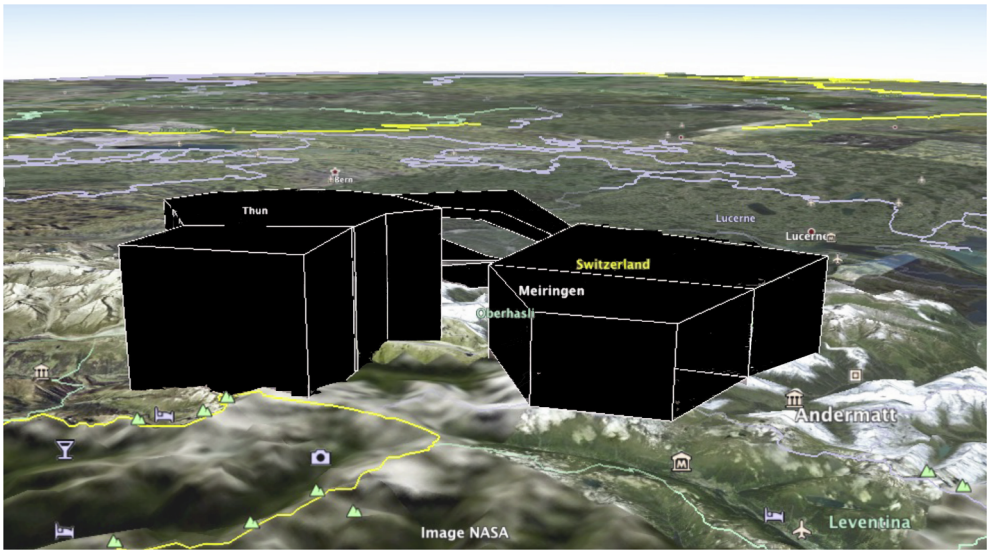

Fig. 1. Polygonal representation of Arispaces.

cuboids into safe and unsafe zones by assessing them regarding the flight safety. The unsafe cuboids are treated as obstacles and the rest are parts of free space. Thereafter, the algorithm forms a larger polyhedrons by merging cuboids. The resultant large polyhedrons are either added to the set O or $\mathcal{R}$. Readers are referred to Appendix A for detailed description of the algorithm.

3.2. Modeling obstacles with polyhedrons

In the route planning problem, obstacles are zones through which flying is prohibited. For our studied problem, three types of obstacles exist: zones with extreme wind condition, restricted airspaces and physical barriers. As mentioned in Section 3.1, cuboids having extreme wind conditions are treated as obstacles (i.e. it is forbidden to fly through them). In the following parts, we explain the definition of restricted airspace as well as elevation data.

3.2.1. Restricted airspace

Airspace is a three-dimensional portion of the atmosphere that has the specific flying rules. Among all types of airspaces, our interest is on those that are prohibited to pass by the aircraft (aka restricted airspace). Fig. (1) shows these airspaces for central Switzerland. In our dataset, restricted airspaces are stored as polyhedron (with horizontal boundaries and altitude range). These polyhedrons are added to the set O.

3.2.2. Elevation

For a given latitude and longitude, the elevation data returns the true altitude (i.e. height above the mean sea level). To transform this elevation data into polyhedrons, we discretize two-dimensional horizontal plane in grid cells. We then use Algorithm 3 to merge these grid cells and form large polyhedrons presenting the flying environment. Interested readers are referred to Appendix B for detailed description of the algorithm. Fig. 2 shows the outcome of applying this algorithm for a given region in Switzerland (Switzerland is particularly a nice and challenging case study due to both mountainous geography and its military sites). In this figure, we see that for a given sub-region trapped between the origin and destination, the polyhedrons are of different altitudes based on the maximum height of that specific sub-area. In addition, polyhedrons have been merged according to the algorithm proposed in Appendix B. Here, we should note that the presented polyhedrons are non-convex. These polyhedrons will be transformed into series of convex-polyhedrons which will be used by the mathematical model.

4. Mathematical formulation

The main idea behind our formulation is to model flyable route with constant sample temporal discretization. We approximate the flight duration as a summation of time intervals in which the aircraft kinodynamic constraints hold. Table 2 summarizes the notation of our formulation.

We define $k \in K$ as the index of time interval and Δt as its duration. At interval k, the aircraft motion is modeled by using three types of variables: (1) position vector $\mathbf{P}^k = \left[p_x^k, p_y^k, p_z^k\right]^T$ (2) velocity vector $\mathbf{V}^k = [v_x^k, v_y^k, v_z^k]^T$ and (3) acceleration vector $\mathbf{A}^k = [a_x^k, a_y^k, a_z^k]^T$. Among them position and velocity vectors are state variables and acceleration vector is a control variable. Assuming constant acceleration during each interval, the flight dynamics at interval $k+1$ is expressed by

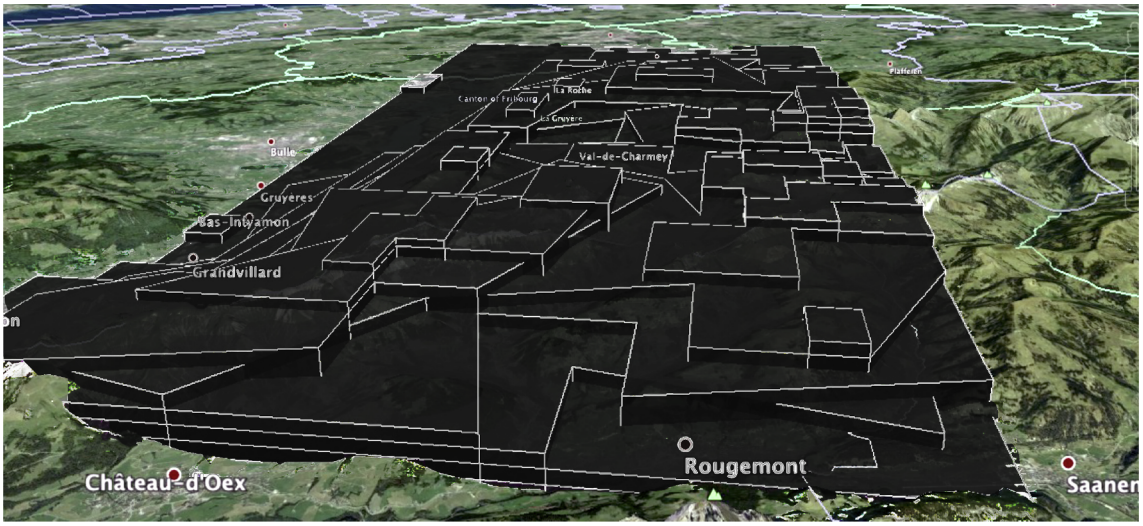

Fig. 2. Polygonal representation of terrain elevation.

Table 2
Table of notations.

Sets:	
$k \in K$	Set of time intervals
$r \in \mathcal{R}$	Set of polyhedrons denoting free space
$o \in O$	Set of polyhedrons modeling obstacles
$z \in \mathcal{Z}$	Set of polyhedrons defining flying boundaries
Parameters:	
Δt	Time interval
$\mathcal{V}^{max}$	Maximum aircraft speed
$\mathcal{V}^{min}$	Minimum aircraft speed
v_z^{min}	Admissible descending speed
v_z^{max}	Admissible ascending speed
$\mathcal{A}^{max}$	Maximum aircraft acceleration
$A_{\cdot}$	Matrix of coefficients in polyhedrons equations
$\overrightarrow{w_r}$	Wind forecast for polyhedron $r \in \mathcal{R}$
$\overrightarrow{i_{dd'}}$	Unit vector
$\mathbf{M}$, $\hat{\mathbf{M}}$	Large constants
Variables:	
$\mathbf{P}^k$	Position vector of the aircraft at interval k
$\mathbf{V}^k$	Velocity vector of the aircraft at interval k
$\mathbf{A}^k$	Acceleration vector of the aircraft at interval k
$s_{dd'}^k$	Binary variable equals to 1 if the aircraft satisfies the minimum speed requirement at interval k in direction of $i_{dd'}$ vector
f_{no}^k	Binary variable equals to 1 if at interval k the aircraft is positioned inside the half-space defined by inequality $n \in \|N\|_o$
e_r^k	Binary variable to detect the location of aircraft in 3D space
y_z^k	Binary variable equals to 1 if the aircraft is positioned inside flying environment bounded by polyhedron z
g^k	Binary variable equals to 1 if the aircraft arrives at destination in time interval k

$\mathbf{P}^{k+1} = \mathbf{P}^k + \mathbf{V}^k(\Delta t) + \frac{1}{2}\mathbf{A}^k(\Delta t)^2$ and $\mathbf{V}^{k+1} = \mathbf{V}^k + \mathbf{A}^k(\Delta t)$. In the next section, we describe the constraints showing the aircraft characteristics.

4.1. Constraints on aircraft characteristics

A kinodynamic compatible route must respect certain rules on aircraft speed and acceleration. Consider the velocity vector $\mathbf{V}^k$ and θ and ϕ as its polar and azimuthal angles. The magnitude of this vector (noted by $||\mathbf{V}^k||$) is measured as $||\mathbf{V}^k|| = v_x^k \sin\phi\cos\theta + v_y^k \sin\phi\sin\theta + v_z^k \cos\phi$ which is a non-linear term. We approximate $||\mathbf{V}^k||$ in a linear fashion (in order to solve the mathematical model in MILP format) with the help of uniformly distributed vectors (noted by $\overrightarrow{i_{dd'}}$). These vectors approximate the direction of velocity and acceleration vectors. Let $\theta_d = \frac{2\pi d}{D}$ $(d = 1, ..., D)$ be the discritization of polar angle θ and $\phi_{d'} = \frac{2\pi d'}{D'}$

($d' = 1, \ldots, D'$) be the discritization of azimuthal angle. The set of uniformly distributed vectors $\overrightarrow{i_{dd'}}$ are defined as follows:

$$\overrightarrow{i_{dd'}} = \begin{bmatrix} \sin\phi_{d'}\cos\theta_d \\ \sin\phi_{d'}\sin\theta_d \\ \cos\phi_{d'} \end{bmatrix} \quad d \in \{1, \ldots, D\},\ d' \in \{1, \ldots, D'\}$$

Based on the aircraft's specifications, we define $\mathcal{V}^{max}$ and $\mathcal{V}^{min}$ as parameters noting aircraft maximum and minimum speed, respectively. Moreover, we define v_z^{min} and v_z^{max} as "admissible vertical speed range" during ascent and descent phases. For time interval k and direction $\overrightarrow{i_{dd'}}$, constraints (1) limit the aircraft speed to its maximum value.

$$\mathbf{V}^k.\ \overrightarrow{i_{dd'}} \leqslant \mathcal{V}^{max} \quad d \in D,\ d' \in D',\ k \in K \tag{1}$$

Similarly, constraints (2) and (3) maintain the restriction on aircraft's minimum speed. To avoid stalling, the restriction on the aircraft minimum speed should be satisfied in at least one direction ($\overrightarrow{i_{dd'}}$). For interval k, we define $s_{dd'}^k$ as a binary variable equal to one if the aircraft speed is more than its minimum value in direction $\overrightarrow{i_{dd'}}$ and 0, otherwise. Constraints (2) and (3) control the conditions on aircraft minimum speed. $\widehat{\mathbf{M}}$ is a vector of large constants used for velocity constraints.

$$\mathbf{V}^k.\ \overrightarrow{i_{dd'}} \geqslant \mathcal{V}^{min} - \widehat{\mathbf{M}}(1 - s_{dd'}^k) \quad d \in \widehat{D},\ d' \in \widehat{D}',\ k \in K \tag{2}$$

$$\sum_{d \in D} \sum_{d' \in D'} s_{dd'}^k \geqslant 1 \quad k \in K \tag{3}$$

Finally, as previously mentioned, constraints (4) bound the aircraft vertical speed to prevent it from stalling.

$$v_z^{min} \leqslant v_z^k \leqslant v_z^{max} \quad k \in K \tag{4}$$

Similar to the velocity vector, constraints (5) limit the acceleration magnitude. Parameter $\mathcal{A}^{max}$ denotes the maximum acceleration limit.

$$\mathbf{A}^k.\ \overrightarrow{i_{dd'}} \leqslant \mathcal{A}^{max} \quad d \in D,\ d' \in D',\ k \in K \tag{5}$$

In the following section, we present obstacle avoidance constraints.

4.2. Constraints on obstacle avoidance

As mentioned in Section 3, obstacles are modeled by convex polyhedrons. Set O denotes all obstacles in our model. Obstacle o has $|N|_o$ faces and defined by a set of inequalities (i.e., $\mathrm{A}_o\mathbf{x} \leqslant \mathrm{b}_o$) in which the size of A_o is $|N|_o \times 3$ and b_o, $|N|_o \times 1$.

Defining binary variables f_{no}^k take 0 if the aircraft position (i.e. $\mathbf{P}^K$) is outside of the half-space defined by the inequality $n \in |N|_o$ and 1, otherwise. The aircraft position is outside obstacle o if its position vector violates at least one of the half-space defined by inequalities $A_o\mathbf{x} \leqslant b_o$. In other words, $f_{no}^k = 1$ for at most $|N|_o - 1$ inequalities. Constraints (6) and (7) enforce the above mentioned constraints avoiding collision with obstacles. $\mathbf{M}$ is a vector of large constants used for position constraints.

$$\mathrm{A}_o\mathbf{P}^k + \mathbf{M}f_{no}^k \geqslant \mathrm{b}_o \quad k \in K,\ n \in |N|_o,\ o \in O \tag{6}$$

$$\sum_{n \in N^o} f_{no}^k \leqslant \left|N\right|_o - 1 \quad k \in K,\ o \in O \tag{7}$$

In this problem the flying route is constructed by connecting aircraft's position vectors. Although constraints (6) and (7) make sure that the aircraft position is outside the given obstacle, they cannot guarantee whether the path connecting two consecutive locations are collision-free. This condition is unlikely to happen if the duration of time interval is relatively small. To decrease the risk of collision with obstacles, in practice one can consider a safety flying margin around each obstacle and inflate the defining polyhedron accordingly.

In the next part, we explain how we integrate external wind in the model.

4.3. Integrating aircraft motion with external wind

In order to calculate flight duration at each time interval, we first need to identify the polyhedrons ($r \in \mathcal{R}$) indicating the free space where the aircraft can fly through. These polyhedrons are defined to be disjoint. Therefore, at each time interval, the aircraft flies through exactly one polyhedron. We define $\mathrm{A}_r\mathbf{x} \leqslant \mathrm{b}_r$ as the set of inequalities to show polyhedron r. Binary variable ℓ_r^k equals to 1, if the aircraft flies through polyhedron r at time interval k. Constraints (8) and (9) are used to identify the polyhedron in which the aircraft flies.

$$\mathrm{A}_r\mathbf{P}^k \leqslant \mathrm{b}_r + \mathbf{M}(1 - \ell_r^k) \quad k \in K,\ r \in \mathcal{R} \tag{8}$$

$$\sum_{r \in R} \ell_r^k = 1 \quad k \in K \tag{9}$$

For polyhedron r, let $\overrightarrow{w_r}$ be the vector representing its wind forecast. For interval k, $\mathbf{V}^k + \overrightarrow{w_r}$ denotes the aircraft ground speed. By having the ground speed, constraints (10) and (11) represent the state of the aircraft at interval $k+1$ taking into account the ground speed.

$$\mathbf{P}^{k+1} = \mathbf{P}^k + \mathbf{V}^k\left(\Delta t\right) + \sum_{r\in\mathcal{R}} \ell_r^k \overrightarrow{w_r}\left(\Delta t\right) + \frac{1}{2}\mathbf{A}^k(\Delta t)^2 \quad k \in 1, \ ..., \left|K\right| - 1 \tag{10}$$

$$\mathbf{V}^{k+1} = \mathbf{V}^k + \mathbf{A}^k(\Delta t) \quad k \in 1, \ ..., |K| - 1 \tag{11}$$

In fact, constraint (10) shows the next position $(k+1)$ of the aircraft by considering the current speed and acceleration as well as the wind speed and direction. The next section explains the constraints that make sure that the aircraft remains in the bounded region. These constraints help to reduce the computational time of the resolution approach.

4.4. Constraints on flying boundaries

Without loss of generality, we assume that the aircraft flies in a bounded environment. That means the aircraft must remain within these boundaries at all times. Let $z \in \mathcal{Z}$ be a set of polyhedrons defining flying boundaries which are represented by inequalities $\mathrm{A}_z\mathbf{x} \leqslant \mathrm{b}_z$. At interval k, we define the binary variable y_z^k as an indicator variable. It takes value 1, if the aircraft position is inside the polyhedron z and 0, otherwise.

$$\sum_{z\in Z} y_z^k = 1 \quad k \in K \tag{12}$$

$$\mathrm{A}_z\mathbf{P}^k \leqslant \mathrm{b}_z + \mathbf{M}(1 - y_z^k) \quad k \in K, \ z \in Z \tag{13}$$

Constraints (12) show that at each time interval, the aircraft flies through exactly one of these polyhedrons. Then, we ensure that the aircraft position remains inside the associated polyhedron using constraints (13). In the following subsection, we explain the objective function.

4.5. Objective function

In this problem, we are interested in finding a route with the minimum traveling time. Initially ($k = 0$), the aircraft is positioned at the origin where the velocity and acceleration vectors equal to zero. This aircraft will reach the destination if its position vector $\mathbf{P^k}$ is inside the destination point. We model the destination as a polyhedron defined by inequality $\mathrm{A}_d\mathbf{P^k} \leqslant \mathrm{b}_d$. We define g^k as a binary variable equal to 1 if the aircraft arrives at destination. Constraints (14) and (15) make sure that the aircraft arrives at destination in only one of the time intervals during the planning horizon.

$$\mathrm{A}_d\mathbf{P}^k \leqslant \mathrm{b}_d + \mathbf{M}(1 - g^k) \quad k \in K \tag{14}$$

$$\sum_{k\in K} g^k = 1 \tag{15}$$

The objective function (minimizing the route duration) can be seen as finding the minimum number of time intervals required to reach the destination (time intervals are of equal sizes). With the aforementioned description the mathematical model is defined as follows:

$$\min \quad \sum_{k\in K} k g^k \tag{16}$$

subject to:

(1)–(15).

In the following section, we explain our proposed two-stage resolution approach.

5. Two-stage route planning approach

Finding the minimum-time route between two locations amid polyhedral obstacles is proven to be NP-hard (Canny and Reif, 1987). In Section 4, we have presented a mathematical formulation to find the flying route. In Section 6, we show that the model finds a kinodynamic feasible route for small-sized instances with a reasonable computational time. However, for larger instances, it becomes computationally expensive due to a large number of constraints and variables required to be handled by the mathematical model.

The primary goal of the two-stage approach is to confine the flying boundaries (i.e. $\mathcal{Z}$) and to reduce the number of variables and constraints in the mathematical model. As presented in Algorithm 1, in the first stage, the algorithm aims to find an approximate kinodynamic feasible path by simplifying the aircraft kinodynamic interaction with the environment. This path is used as a guideline to form a bounded flying region in the second stage. Thus, the mathematical model is solved over a smaller space ($\bar{\mathcal{M}}$) that makes the algorithm computationally more efficient.

Algorithm 1. Two-stage route planning approach

Input: Aircraft specifications, flying environment (**M**), origin, destination
Output: Minimum-time flying route
Stage 1: **Finding kinodynamic compatible path**
(1) $G(V, A) \leftarrow$ Build a graph modeling the free-space
(2) $\hat{\tau} \leftarrow$ Find the minimum-time path on graph G
Stage 2: **Forming the feasible corridor and computing the route**
(3) $\bar{\mathcal{M}} \leftarrow$ Identify the essential polyhedrons for trajectory design (τ)
(4) $Z \leftarrow$ Form the flying boundaries ($\bar{\mathcal{M}}$)
(5) Solve the mathematical model (Z, $\bar{\mathcal{M}}$)

5.1. Stage 1: Finding kinodynamic compatible path

The purpose of the first stage is to find the minimum-time, collision-free path which takes into account the aircraft's kinodynamic behavior as well as its interaction with wind. In step (1) of Algorithm 1, this path is presented on a directed graph $G = (V, A)$ in which V is the set of nodes and A is the set of arcs. In this graph, nodes present either the aircraft ascent/descent flying phase or its movement in the free space.

We model the aircraft ascent and descent phases by a reversed right circular truncated cone. The apex (a cone is a three-dimensional geometric shape that tapers smoothly from a flat base to a point called the apex or vertex) is origin/destination and the maximum ascend/descent angle defines the opening angle of the cone. The height of the cone is the maximum flying altitude (i.e. 10,000 feet). We discretize the cone surface by grid cells and a node is created for each grids' center point. Fig. 3 illustrates how these reversed cones are used to show ascent and descent angles.

We use all polyhedrons in the set **M** to model the aircraft movements in free space. For each obstacle ($o \in O$), a series of nodes are created representing its vertices. For a given polyhedron in the set $\mathcal{R}$ three types of nodes are generated. We first generate a node for each vertex. Second, a set of equidistant nodes are generated on all edges of the polyhedron. Third, a series of nodes are randomly created inside it. For the latter case, we discretize polyhedrons' altitude range into equidistant levels. For each level, a set of uniformly distributed nodes are generated. The number of generated nodes depends on the pre-defined density parameter.

We connect two nodes if the connection satisfies all the following conditions: (1) both nodes should be inside the same free space (i.e. $r \in \mathcal{R}$). (2) the connection should be collision-free (i.e. no obstacle exists in between) and (3) the connection should respect the aircraft acceptable descending/ascending range.

Between each pair of nodes (i.e. i and j), two arcs (in different direction) exist. These arcs have the same Euclidean distance (noted by d_{ij}) with different travel time τ_{ij}. For each direction, first we assume that the aircraft flies with its maximum speed. Second, the

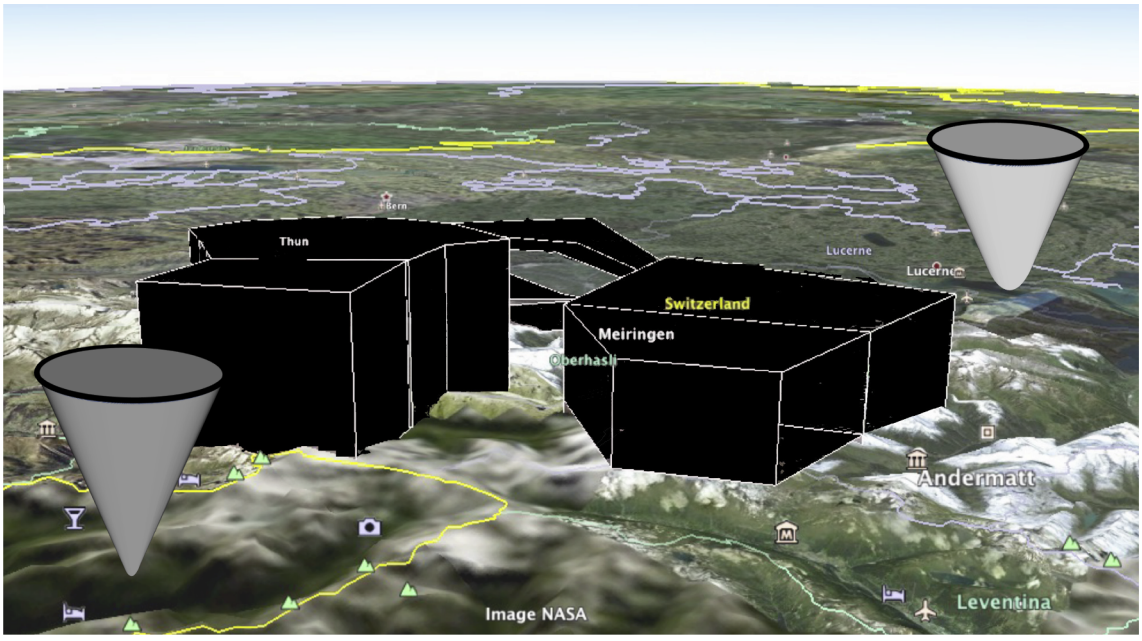

Fig. 3. Reverse right circular truncated cone used to show ascent and descent angles.

aircraft velocity vector is in the same direction as the connecting arc. Between two nodes, the ground speed is calculated as $||\widehat{V}_{ij} + \overrightarrow{w_r}||$ ($\overrightarrow{w_r}$ is the forecast wind). The travel time (τ_{ij}) is then: $\tau_{ij} = \frac{d_{ij}}{||\widehat{V}_{ij} + \overrightarrow{w_r}||}$.

After all elements of the graph are created, in step (2) of Algorithm 1, we use the well-known Dijkstra algorithm (Dijkstra, 1959) to find the shortest path connecting the origin to the destination. We present this path by $\hat{\tau}$.

5.2. Stage two: forming the feasible corridor and computing the route

The shortest path, computed in the first stage, ($\hat{\tau}$) has three main features which help us to form the flying boundaries. (1) It approximately reveals the areas in which the optimal route exists. (2) It makes sure that the selected flying areas are safe and feasible (i.e., obstacle-free route exists). (3) It ensures that the generated route is compatible with aircraft ascent and descent angles. In the second stage (steps (3)–(5) of Algorithm 1), we use $\hat{\tau}$ to identify essential polyhedrons and confine the flying boundaries. These polyhedrons are later used in the mathematical model.

We first inflate the shortest path, $\hat{\tau}$, by + δ value on the surface to ensure the safety of flight and to avoid any collisions. Then, the algorithm verifies the intersection of the inflated path with all polyhedrons in **M** (step (3) in Algorithm 1). For all polyhedrons, we use their projection in the horizontal area. A polyhedron is marked as essential, if its projection has a non-empty intersection with the inflated shortest path. All the marked polyhedrons are added to the set $\bar{\mathcal{M}}$.

In step (4) of Algorithm 1, the algorithm identifies the flyable boundaries by only considering polyhedrons that present obstacles (i.e. $o \in \{O \cap \bar{\mathcal{M}}\}$). We define the flyable boundaries as the smallest bounded area that contains the existing obstacles in the set $\bar{\mathcal{M}}$. This area is defined by disjoint polyhedrons (noted by $\mathcal{Z}$ in the mathematical model). Finally, at step (5) in Algorithm 1, the optimal route is computed by solving the mathematical model over the flyable environment, $\bar{\mathcal{M}}$.

5.3. Pre-processing

In the presented model, constraints (6), (8) and (14) are used to verify the feasibility of the computed route. At each time interval (ΔT), these inequalities verify whether the aircraft is positioned inside/outside of a specific polyhedron. In all cases, the aircraft must remain within the boundaries defined by constraints (13). Thus, by using flying boundaries (noted by $\mathcal{Z}$), one can determine a bound on the position of the aircraft and tighten the above mentioned inequalities. We have defined $\mathbf{M} = [M_x, M_y, M_z]^T$ as a large constant used in these constraints. In this problem, we assume that the aircraft cannot fly above 10,000 feet. This value is chosen for M_z. To find a value for large constants M_x and M_y, we examine all the corner points of polyhedrons defining $\mathcal{Z}$. For each dimension (x and y), we identify the minimum and the maximum values based on their coordinates. We set the difference between these maximum and minimum to define M_x and M_y used in the formulation. For velocity vector, we choose $\widehat{\mathbf{M}} = [3V^{min}, 3V^{min}, 3V^{min}]^T$.

To determine a bound on the number of required time intervals, we use the travel time calculated in the first stage of the algorithm ($\hat{\tau}$). We simply set $|K| = \lceil \kappa\hat{\tau}/\Delta T \rceil$ in which $\kappa > 1$.

6. Computational experiments

In this section, we present the computational experiments of our two-stage route planning algorithm. The algorithm was coded in C++ and CPLEX 12.5 was used as a mixed-integer linear programming solver. All experiments were carried out on a computer with 2.4 GHz CPU and 8 GB of RAM.

6.1. Experimental settings

We evaluate the performance of the two-stage algorithm on a set of case studies derived from route planning between small airports in central Europe. Nine cases have been selected. These cases are chosen based on the three different levels of difficulty: large, medium and small (named as IL, IM, and IS) in terms of flying distance, number of obstacles and mountainousness. Table 3

Table 3
Instance description.

Case	Origin			Destination		
Name	Code	Lat.	Long.	Code	Lat.	Long.
IS-01	LSGK	N46°29.18′	E7°14.92′	LSGE	N46°45.30′	E7°04.57′
IS-02	LSZJ	N47°11.02′	E7°05.45′	LSPL	N47°10.97′	E7°44.48′
IS-03	XX01	N46°26.24′	E7°02.24′	XX11	N46°12.0′	E7°00.36′
IM-01	XX02	N46°23.24′	E7°06.00′	XX12	N46°23.24′	E7°7.12′
IM-02	LSMP	N46°50.55′	E6°54.82′	LSGS	N46°13.15′	E7°19.62′
IL-01	LSGY	N46°45.72′	E6°36.80′	EDTZ	N47°40.65′	E9°08.45′
IL-02	LFCL	N43°35.27′	E1°29.92′	LESO	N43°21.39′	W1°47.44′
IL-03	LFLI	N46°11.52′	E6°16.10′	LFGP	N47°58.88′	E3°46.62′
IL-04	LFNT	N43°59.77′	E4°45.27′	LSZJ	N47°11.02′	E7°5.45′

shows the coordinates of origin and destination airports for each of our test cases.

We have defined the initial flying boundary (i.e. **M**) as a rectangular area. The altitude of this region is set to the maximum flying altitude (3 km). The horizontal boundaries are determined as follows: we calculate the shortest distance from the origin to the destination (d_{od}). We then assume a circle with radius d_{od} in which its center is positioned mid-way between the origin and the destination. We define the smallest rectangle bounding this circle as the initial flying boundaries (**M**). To determine the flying boundaries on reduced environment ($\bar{\mathcal{M}}$), we inflate the shortest path by three km ($\delta = 3$ km).

For the selected flying boundaries, we retrieve the real data needed to model elevation, airspace and wind as polyhedrons. For the wind data, we have chosen a day with relatively stable weather condition which reflects the typical operational condition of the light aircraft. To construct the polyhedrons (elevation and wind), we have chosen the horizontal (i.e. latitude, longitude) and vertical (i.e. altitude) discretization of two km and one km, respectively. We modeled the destination polyhedron as a cuboid with length of 2 km and width of 0.3 km representing the typical runway. We declare the aircraft as landed when its altitude is less than 0.1 km from the ground. We set the discretization interval to $\Delta T = 0.008$ h for all experiments. Finally, we set $\kappa = 1.2$ to calculate the maximum number of time intervals.

We found the route for an aircraft with the following specifications: $\mathcal{A}^{max} = 4.5\ \mathrm{ms}^{-2}$, $\mathcal{V}^{max} = 210\ \mathrm{km\ h}^{-1}$, $\mathcal{V}^{min} = 80\ \mathrm{km\ h}^{-1}$, $v_z^{min} = 9\ \mathrm{km\ h}^{-1}$ and $v_z^{max} = 13\ \mathrm{km\ h}^{-1}$. Finally, we have chosen $D = 8$ and $D' = 8$ to control the maximum velocity and acceleration and $\widehat{D} = 6$ and $\widehat{D}' = 6$ to verify the conditions on minimum velocity for all studied cases.

6.2. Results

We now present the numerical results of the two-stage route planning algorithm. For each case, we compare the solution of the two-stage approach with that of the solution obtained by solving the mathematical model. For the latter case, no reduction is applied to the number of polyhedrons as well as flying boundaries.

Table 4 reports the detailed computational experiments for each case. Table headings are:

- Name: the name of case study.
- E-map: the total time required to recast the data as polyhedrons (i.e. all polyhedrons in **M**). “Pre-processing phase”.
- Graph: the time required to build the graph (Algorithm 1, step 1).
- Path: the computational time to find the shortest path (Algorithm 1, step 2).
- R-map: the time required to build the reduced map (Algorithm 1, steps 3 and 4). The reduced (or truncated) map is denoted by $\bar{\mathcal{M}}$ in our description.
- R-mth: the CPU time to solve the mathematical model over the reduced environment (i.e. R-map).
- Total: total computational time to solve the two-stage approach.
- R-Val: The flight duration (expressed in minutes) when solving the mathematical model over the reduced environment.
- E-mth: The CPU time to solve the mathematical model over the initial space (E-map).
- E-val: The flight duration (expressed in minutes) when solving the mathematical model over the initial space.

As can be seen in Table 4, the two-stage approach finds the flying route in around 325 s, on average. Around 40% of this time is dedicated to the graph construction (139.1 s). In practice, the graph construction can be done as an offline process thus, further reduction in the total computational time is possible.

For most of the cases, the flight duration solved by the two-stage approach (R-Val) is similar to the solution of the mathematical model over the initial space (E-Val). Exceptions are IM-02, IL-01, IL-02 and IL-04. For these cases, the flight duration is slightly higher than the benchmark. The benefit of using our proposed two-stage approach is to find the efficient route between each pair of O-D with less computational time.

Table 4
Computational results.

	Two-Stage - Reduced env.							Extended env.	
Name	E-map (sec)	Graph (sec)	Path (sec)	R-map (sec)	R-mth (sec)	Total (sec)	R-val (min)	E-mth (sec)	E-val (min)
IS-01	86	31.5	3.8	1.3	28.8	65.4	9.4	158.5	9.4
IS-02	65.9	24.1	2.9	2.2	56.7	99.9	14	128.7	14
IS-03	219.6	80.4	9.6	2.2	42.4	148.9	14.3	381.2	14.3
IM-01	195.8	71.7	8.6	3.5	108.2	192	21.4	410.1	21.4
IM-02	270.8	99.2	11.8	5.7	95.6	212.3	21.9	444.9	21.7
IL-01	673.4	246.6	29.4	6.9	130.4	413.3	63.5	1153.4	63.4
IL-02	439.2	160.8	19.2	8.3	186.4	374.7	86.3	801.8	86.0
IL-03	644.2	235.8	28.2	7.7	195.5	467.2	68.1	1237.3	68.1
IL-04	851.0	311.6	37.2	9.4	182.9	541.1	99.1	1475.3	98.9
Average	382.88	139.1	16.8	5.2	114.1	325.74		687.9	

Table 5
Number of Polyhedrons for each case.

Case	E-map				R-map				Reduction
Name	Airspace	Elevation	Wind	Total	Airspace	Elevation	Wind	Total	%
IS-01	3	32	12	47	3	12	6	21	55.3
IS-02	12	15	9	36	10	11	7	28	22.2
IS-03	5	83	32	120	5	17	14	36	70.0
IM-01	8	78	21	107	8	32	19	59	44.9
IM-02	11	101	36	148	11	63	21	95	35.8
IL-01	73	227	68	368	22	78	15	115	68.8
IL-02	48	129	63	240	17	91	31	139	42.1
IL-03	41	240	71	352	11	92	26	129	63.3
IL-04	67	346	52	465	30	103	23	156	66.4
Average									52.1

Regarding the computational time, there is a significant reduction of computational time when running CPLEX over the reduced environment. For the extended environment, CPLEX takes on average 687.9 s to find the optimal route. This value decreases to 114.1 s for the reduced environment. There are two main reasons behind this reduction. First, with the two-stage approach, the mathematical model finds a route in very restricted areas (defined by the set $\mathcal{Z}$ in our formulation). This way, several nodes are eliminated in the branch and bound tree due to the infeasibility of the aircraft position. This is not the case when solving the mathematical model over the extended environment. Second, the mathematical model solved over the reduced environment has fewer constraints and variables due to the elimination of non-essential polyhedrons. Here, we note that in both cases the mathematical model has the same number of time intervals. Moreover, the branch and bound algorithm starts with the same initial solution in both cases. Table 5 compares the number of convex polyhedrons for the extended (E-map) and reduced (R-map) environment. As reported, on average, there is 52% reduction in the number of convex polyhedrons after applying the two-stage approach.

We are now interested in evaluating the solution's sensitivity with respect to the time interval selection. Thus, the algorithm returns the route with the minimum distance. We have chosen IS-01 case in which a restricted airspace exists between origin and destination (marked as a red polyhedron in Fig. 4). Three value's of time intervals (in hours) are tested: $\Delta t = 0.016$, 0.008 and 0.004. Table 6 reports the computational time. For the selected case, the minimum distance route is the one that starts from the origin, touches the airspace corner and arrives at the destination. As can be seen in Fig. 4, by selecting $\Delta = 0.004$, the mathematical model returns the route which is visually similar to the optimal one. However, by increasing the time interval's length, the number of symmetric routes increases. Symmetric routes are those having the same number of time intervals (i.e. route duration deviates at most Δt minutes). In this case, these solutions may be deviated from the optimal one. On the other hand, the better precision comes with the price of higher computational time. For the IS-01 case, the computational time increases from 131.5 to 378.9 s.

To test the impact of wind on flight's duration, we have chosen IL-02 instance, previously introduced in Table 3. For this case, we have considered a homogeneous dominant wind for the entire region. Here, we have assumed that the wind has the speed of 30 km/h

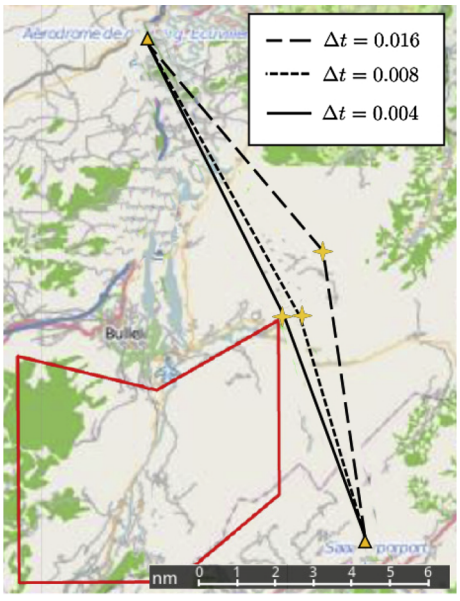

Fig. 4. The effect of time interval length on calculated route (Case IE-01).

Table 6
The effect of time interval length on computational (Case IS-01).

Δt (h)	val (min)	time (sec)
0.016	9.7	131.5
0.008	9.4	158.5
0.004	9.3	378.9

Table 7
The effect of wind on traveling time for instance IL-02.

Wind speed (km/h)	Wind direction (°)	Val. (min)	Time (sec)
30	45	81.9	173.8
30	135	76.6	161.4
30	225	93.5	193.6
30	315	97.4	181.3

and created scenarios by varying its direction. The reference for wind direction is North. That is, North shows 0 degrees, East 90, South 180 and West 270. Table 7 presents the wind information for four tested scenarios. By only taking into account the wind direction, the flight duration can vary from 76.6 to 97.4 min.

7. Future work and conclusion

In this paper, we present the route planning algorithm for light aircraft. Our goal is to find a collision-free, kinodynamic compatible route in a three-dimensional space. Moreover, we take into account the interactions of the aircraft with wind. We develop a series of algorithms to recast the flying environment and present it with a series of polyhedrons. We introduce a mixed-integer linear formulation of the problem. Moreover, we create a two-stage approach which can find the near-optimal route with a reasonable computational time. Numerical results show the efficiency of our approach. In this research, we find the minimum-time route between two locations. We only study the route planning for a given aircraft. In the future steps, we would like to expand the algorithm for the case where there are several flying aircraft within the same area. This algorithm has been proven to be computationally efficient. It can be applied for the network of drones with minor changes in the formulation. The proposed route, in this paper which is based on the traveling time, may not be fuel-efficient. In the future step, approaches such as aircraft performance model (see Wasiuk et al., 2015) can be integrated in our resolution method to investigate the trade-off between flying time and fuel consumption and derive a multi-objective version of the route planning problem investigated in this paper.

Acknowledgment

This research is supported by the Commission for Technology and Innovation (CTI) under grant 16610.1 PFES-ES. This support is gratefully acknowledged.

Appendix A. Processing wind data

Algorithm 2 shows the steps to represent the wind forecast data with large polyhedrons. We discretize the space into a set of cuboids (noted by $c \in C$) in which vector $\overrightarrow{w_c}$ denotes its wind forecast. Set C is further partitioned into sets C^s and C^u denoting safe and unsafe flying environments, respectively.

In step (1)–(4), the algorithm checks the cuboids for flight safety based on the aircraft specifications. Then, it either assigns them to the set C^s or C^u. Flying through all cuboids in C^u must be avoided. These cuboids are considered as obstacles.

For a given cuboid, we define its neighboring cuboids as those having a common face with the selected one. In other words, two cuboids are neighbors if they have a non-empty intersection. With our representation, cuboid c has at most six neighboring cuboids. We denote these by the set $\mathbf{N}_c$. We define $\mathcal{L}$ as set of cuboids merging to form a large polyhedron. For this set, $\mathbf{N}_{\mathcal{L}}$ denotes all neighboring cuboids. The neighboring cuboids have at least one non-empty intersection with one of the cuboids in $\mathcal{L}$. In steps (5)–(10), the algorithm reduces the number of cuboids by merging neighboring ones. Here, we merge two neighboring cuboids if both of them belong to the set C^u. Thereafter, we extract all large polyhedrons and add them to the set O.

The similar process is also applied to merge cuboids in the set C^s. In this case, cuboid c' is added to the set of the large polyhedron $\mathcal{L}$ when it has similar wind forecast as all the cuboids inside the set $\mathcal{L}$. We measure this similarity by cosine. Two wind vectors are assumed to be similar if their cosine is greater than $1 - \epsilon_a$ and the difference between their magnitude is less than ϵ_m (ϵ_a and ϵ_m are small fractions). Finally, the formed large polyhedron is added to the set $\mathcal{R}$ (step 16). For this polyhedron ($\mathcal{L}$), the nominal wind is calculated as the average wind vector of all merged cuboids.

Algorithm 2. General steps to recast wind forecast with polyhedrons

Input: Wind forecast, aircraft specification
Output: Polyhedrons in the sets $\mathcal{R}$ and O
for $c \in C$ do
  check c for flight safety
    if c is unsafe then add c to C^u
    else add c to C^s
for all $c \in C^u$ do
  if c has not been marked as visited then
    $\mathcal{L} = \{\}; \mathcal{L} \leftarrow c$
    while $\exists\ c' \in \mathbf{N}_{\mathcal{L}} | c' \in C^u$
      $\mathcal{L} \leftarrow c'$; mark c' as visited
    Form $\mathcal{L}$ as one polyhedron; $O \leftarrow \mathcal{L}$
for $c \in C^s$ do
  if c has not been marked as visited then
    $\mathcal{L} = \{\}; \mathcal{L} \leftarrow c$
    while $\exists\ c' \in \mathbf{N}_{\mathcal{L}}$ in which $\forall\ c \in \mathcal{L}$ we have $\overrightarrow{w_c} \cdot \overrightarrow{w_{c'}} \geqslant 1 - \epsilon_a$ and $|\overrightarrow{w_c}| - |\overrightarrow{w_{c'}}| \leqslant \epsilon_m$ then
      $\mathcal{L} \leftarrow c'$; mark c' as visited
    Form $\mathcal{L}$ as one polyhedron; $\mathcal{R} \leftarrow \mathcal{L}$

Appendix B. Processing elevation data

To form polyhedrons from the elevation data, we discretize the horizontal plane by grids cells ($g \in \mathcal{G}$). We assume that each grid cell has a flat surface and its nominal altitude is denoted by $alt(g)$. In a similar fashion, we discretize the altitude by layers. The altitude discretization is bounded below by mean sea level and above by maximum flying altitude (i.e. $\omega \in \Omega = \{0, ..., 10, 000\}$). For each layer ω, we define L^{ω} as a set of grid cells in which $alt(g) \geqslant \omega$. Finally, $l \in L$ is the power set of L^{ω} (i.e. $L = \cup L^{\omega}$).

Algorithm 3 shows the steps of converting elevation data into larger polyhedrons. For each grid cell first, the algorithm samples points to determine its true altitude. Among all random points, the one with the maximum true altitude is chosen. This value presents the nominal altitude of the grid cell (steps (1)–(2)).

In steps (3)–(5), the algorithm assigns grid cells to layers. The assignment is based on the grid cell nominal altitude ($alt(g)$). Grid cell g is assigned to layer ω if its altitude is greater than layers' representative altitude (i.e. $alt(g) \geqslant \omega$). Here, we should note that one grid cell can be assigned to multiple layers. After applying our assignment rule, grid cells are transformed into cuboid presenting the elevation.

To merge grid cells, we define $\bar{\mathbf{N}}_g$ as a set of neighboring grid cells of g. Two grid cells are neighbors if they have a common face. For our elevation data, each grid cell has at most four neighboring grid cells. Similarly, we define $\mathcal{L}$ as set of cuboids merging to form a large polyhedron. For this set, $\mathbf{N}_{\mathcal{L}}$ denotes all neighboring cuboids. The neighboring cuboids have at least one non-empty intersection with one of the cuboids in $\mathcal{L}$. In steps (6)–(12) the algorithm merges g and g' if they belong to the same set L^{ω}.

Algorithm 3. Modeling elevation data with polyhedrons

Input: Elevation data
Output: Series of polyhedrons modeling elevation
for $g \in \mathcal{G}$ do
  $alt(g) \leftarrow$ Find the maximum altitude of grid cell g
for $g \in \mathcal{G}$ do
    for $\omega \in \Omega$ do
      if $alt(g) \geqslant \omega$ then add grid cell g to L^{ω}
for $l \in L$ do
  for $g \in L^{\omega}$ do
    if g has not been marked as visited then
      $\mathcal{L} = \{\}; \mathcal{L} \leftarrow g$
      while $\exists\ g' \in \bar{\mathbf{N}}_{\mathcal{L}} | g' \in L^{\omega}$
        $\mathcal{L} \leftarrow g'$; mark g' as visited
      Form $\mathcal{L}$ as one polyhedron; $O \leftarrow \mathcal{L}$

The mega polyhedrons constructed by using Algorithms 2 and 3, may have large number of vertices. This causes an increase in the number of inequalities. These inequailities are required to define polyhedrons which leads to computational deficiencies in the mathematical model. To avoid such cases, we have used polygon simplification algorithm proposed by Douglas and Peucker (1973). By applying this algorithm, we are able to reduce the number of vertices while maintaining the shape of the polyhedrons.

References

Adiyatov, O., Varol, H.A., 2013. Rapidly-exploring random tree based memory efficient motion planning. In: 2013 IEEE International Conference on Mechatronics and Automation (ICMA). IEEE, pp. 354–359.

Ahuja, N., Chuang, J.-H., 1997. Shape representation using a generalized potential field model. IEEE Trans. Pattern Anal. Mach. Intell. 19 (2), 169–176.

Barnhart, C., Smith, B., 2012. Quantitative Problem Solving Methods in the Airline Industry. Springer.

Bellingham, J.S., 2002. Coordination and Control of UAV Fleets using Mixed-integer Linear Programming (Ph.D. thesis). Citeseer.

Bircher, A., Kamel, M., Alexis, K., Oleynikova, H., Siegwart, R., 2016. Receding horizon path planning for 3d exploration and surface inspection. Auton. Robots 1–16.

Blanco, M., Borndörfer, R., Hoang, N.-D., Kaier, A., Schienle, A., Schlechte, T., Schlobach, S., 2016. Solving time dependent shortest path problems on airway networks using super-optimal wind. OASIcs-OpenAccess Series in Informatics, vol. 54 Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Bonami, P., Olivares, A., Soler, M., Staffetti, E., 2013. Multiphase mixed-integer optimal control approach to aircraft trajectory optimization. J. Guid., Control, Dyn. 36 (5), 1267–1277.

Borrelli, F., Subramanian, D., Raghunathan, A.U., Biegler, L.T., 2006. Milp and nlp techniques for centralized trajectory planning of multiple unmanned air vehicles. In: American Control Conference, 2006. IEEE, pp. 6.

Boyle, D.P., Chamitoff, G.E., 1999. Autonomous maneuver tracking for self-piloted vehicles. J. Guid., Control, Dyn. 22 (1), 58–67.

Canny, J., Reif, J., 1987. New lower bound techniques for robot motion planning problems. In: 28th Annual Symposium on Foundations of Computer Science, 1987. IEEE, pp. 49–60.

Chamseddine, A., Zhang, Y., Rabbath, C.A., Join, C., Theilliol, D., 2012. Flatness-based trajectory planning/replanning for a quadrotor unmanned aerial vehicle. IEEE Trans. Aerospace Electron. Syst. 48 (4), 2832–2848.

Chen, Y.-b., Luo, G.-c., Mei, Y.-s., Yu, J.-q., Su, X.-l., 2016. Uav path planning using artificial potential field method updated by optimal control theory. Int. J. Syst. Sci. 47 (6), 1407–1420.

Cheng, B., 1962. The Law of International Air Transport, vol. 47 Stevens.

Choset, H., Burdick, J., 2000. Sensor-based exploration: the hierarchical generalized voronoi graph. Int. J. Robot. Res. 19 (2), 96–125.

Connolly, C.I., Burns, J.B., Weiss, R., 1990. Path planning using laplace's equation. In: Proceedings., 1990 IEEE International Conference on Robotics and Automation, 1990. IEEE, pp. 2102–2106.

Darbari, V., Gupta, S., Verma, O.P., 2017. Dynamic motion planning for aerial surveillance on a fixed-wing uav. In: 2017 International Conference on Unmanned Aircraft Systems (ICUAS). IEEE, pp. 488–497.

Dever, C., Mettler, B., Feron, E., Popovic, J., McConley, M., 2004. Trajectory interpolation for parametrized maneuvering and flexible motion planning of autonomous vehicles. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, pp. 5143.

Dijkstra, E.W., 1959. A note on two problems in connexion with graphs. Numer. Math. 1 (1), 269–271.

Ding, Z., Zhang, J., Li, C., Liu, Y., 2015. A real-time path planning method for emergent threats. In: 2015 IEEE International Conference on Information and Automation. IEEE, pp. 3014–3019.

Djokic, J., Lorenz, B., Fricke, H., 2010. Air traffic control complexity as workload driver. Transport. Res. Part C: Emerg. Technol. 18 (6), 930–936.

Douglas, D.H., Peucker, T.K., 1973. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. Cartographica: Int. J. Geograph. Inform. Geovisual. 10 (2), 112–122.

Ferguson, D., Likhachev, M., Stentz, A., 2005. A guide to heuristic-based path planning. In: Proceedings of the International Workshop on Planning under Uncertainty for Autonomous Systems, International Conference on Automated Planning and Scheduling (ICAPS), pp. 9–18.

Fernández, J., Cánovas, L., Pelegrın, B., 2000. Algorithms for the decomposition of a polygon into convex polygons. Eur. J. Oper. Res. 121 (2), 330–342.

Frazzoli, E., Dahleh, M.A., Feron, E., 2000. A hybrid control architecture for aggressive maneuvering of autonomous aerial vehicles. In: System Theory. Springer, pp. 325–343.

Frazzoli, E., Dahleh, M.A., Feron, E., 2002. Real-time motion planning for agile autonomous vehicles. J. Guid., Control, Dyn. 25 (1), 116–129.

Gammell, J.D., Srinivasa, S.S., Barfoot, T.D., 2014. Informed rrt∗: optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. In: Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on. IEEE, pp. 2997–3004.

Geem, Z.W., Kim, J.H., Loganathan, G.V., 2001. A new heuristic optimization algorithm: harmony search. Simulation 76 (2), 60–68.

Girardet, B., Lapasset, L., Delahaye, D., Rabut, C., 2014. Wind-optimal path planning: application to aircraft trajectories. In: Control Automation Robotics & Vision (ICARCV), 2014 13th International Conference on. IEEE, pp. 1403–1408.

Goerzen, C., Kong, Z., Mettler, B., 2010. A survey of motion planning algorithms from the perspective of autonomous uav guidance. J. Intell. Rob. Syst. 57 (1–4), 65.

Holden, J., Goel, N., 2016. Fast-forwarding to a future of on-demand urban air transportation. San Francisco, CA.

Howlet, J.K., Schulein, G., Mansur, M.H., 2004. A practical approach to obstacle field route planning for unmanned rotorcraft.

Jardin, M.R., Bryson, A.E., 2001. Neighboring optimal aircraft guidance in winds. J. Guid., Control, Dyn. 24 (4), 710–715.

Jardin, M.R., Bryson, A.E., 2012. Methods for computing minimum-time paths in strong winds. J. Guid., Control, Dyn. 35 (1), 165–171.

Judd, K., McLain, T., 2001. Spline based path planning for unmanned air vehicles. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, pp. 4238.

Karney, C.F., 2011. Geodesics on an ellipsoid of revolution. arXiv preprint arXiv:1102.1215.

Khuswendi, T., Hindersah, H., Adiprawita, W., 2011. Uav path planning using potential field and modified receding horizon a∗ 3d algorithm. In: Electrical Engineering and Informatics (ICEEI), 2011 International Conference on. IEEE, pp. 1–6.

Kong, Z., Korukanti, V., Mettler, B., 2009. Mapping 3d guidance performance using approximate optimal cost-to-go function. In: AIAA Guidance, Navigation, and Control Conference, pp. 6017.

Kreimeier, M., Stumpf, E., Gottschalk, D., 2016. Economical assessment of air mobility on demand concepts with focus on Germany. In: 16th AIAA Aviation Technology, Integration, and Operations Conference, pp. 3304.

Kuwata, Y., 2003. Real-time Trajectory Design for Unmanned Aerial Vehicles using Receding Horizon Control (Ph.D. thesis). Massachusetts Institute of Technology.

LaValle, S.M., 1998. Rapidly-exploring random trees: A new tool for path planning.

LaValle, S.M., 2006. Planning Algorithms. Cambridge University Press.

LaValle, S.M., Kuffner Jr., J.J., 2001. Randomized kinodynamic planning. Int. J. Robot. Res. 20 (5), 378–400.

Marchidan, A., Bakolas, E., 2015. Numerical techniques for minimum-time routing on sphere with realistic winds. J. Guid., Control, Dyn. 39 (1), 188–193.

Mettler, B., Dadkhah, N., Kong, Z., 2010. Agile autonomous guidance using spatial value functions. Control Eng. Pract. 18 (7), 773–788. Special Issue on Aerial Robotics http://www.sciencedirect.com/science/article/pii/S0967066110000730 .

Mettler, B., Kong, Z., 2008. Receding horizon trajectory optimization with a finite-state value function approximation. In: American Control Conference, 2008. IEEE, pp. 3810–3816.

Miller, B., Stepanyan, K., Miller, A., Andreev, M., 2011. 3d path planning in a threat environment. In: 2011 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC). IEEE, pp. 6864–6869.

Nelson, G.D., Rae, A., 2016. An economic geography of the united states: from commutes to megaregions. PloS One 11 (11), e0166083.

Prats, X., Puig, V., Quevedo, J., Nejjari, F., 2010. Multi-objective optimisation for aircraft departure trajectories minimising noise annoyance. Transport. Res. Part C: Emerg. Technol. 18 (6), 975–989.

Qian, X., Mao, J., Chen, C.-H., Chen, S., Yang, C., 2017. Coordinated multi-aircraft 4d trajectories planning considering buffer safety distance and fuel consumption optimization via pure-strategy game. Transport. Res. Part C: Emerg. Technol. 81, 18–35.

Redding, J., Amin, J., Boskovic, J., Kang, Y., Hedrick, K., Howlett, J., Poll, S., 2007. A real-time obstacle detection and reactive path planning system for autonomous small-scale helicopters. In: AIAA Guidance, Navigation and Control Conference and Exhibit, pp. 6413.

Reif, J.H., 1979. Complexity of the mover's problem and generalizations. In: Foundations of Computer Science, 1979., 20th Annual Symposium on. IEEE, pp. 421–427.

Richards, A., Bellingham, J., Tillerson, M., How, J., 2002. Coordination and control of multiple uavs. In: AIAA Guidance, Navigation, and Control Conference,

Monterey, CA.
Richards, A., How, J.P., 2002. Aircraft trajectory planning with collision avoidance using mixed integer linear programming. In: American Control Conference, 2002. Proceedings of the 2002, vol. 3. IEEE, pp. 1936–1941.
Sch, F., la Cour-Harbo, A., Bisgaard, M., et al., 2012. Generating approximative minimum length paths in 3d for uavs. In: Intelligent Vehicles Symposium (IV), 2012 IEEE. IEEE, pp. 229–233.
Scherer, S., Singh, S., Chamberlain, L., Saripalli, S., 2007. Flying fast and low among obstacles. In: Robotics and Automation, 2007 IEEE International Conference on. IEEE, pp. 2023–2029.
Schøler, F., 2012. 3D Path Planning for Autonomous Aerial Vehicles in Constrained Spaces (Ph.D. thesis). Section of Automation & Control, Department of Electronic Systems, Aalborg University.
Sinopoli, B., Micheli, M., Donato, G., Koo, T.-J., 2001. Vision based navigation for an unmanned aerial vehicle. In: Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on, vol. 2. IEEE, pp. 1757–1764.
Soler, M., Zou, B., Hansen, M., 2014. Flight trajectory design in the presence of contrails: application of a multiphase mixed-integer optimal control approach. Transport. Res. Part C: Emerg. Technol. 48, 172–194.
Suzuki, S., Komatsu, Y., Yonezawa, S., Masui, K., Tomita, H., 2005. Online four-dimensional flight trajectory search and its flight testing. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, pp. 6475.
Tang, B., Mukherjee, A., 2000. System and method for generating optimal flight plans for airline operations control. US Patent 6,134,500.
Tsenkov, P., Howlett, J., Whalley, M., Schulein, G., Takahashi, M., Rhinehart, M., Mettler, B., 2008. A system for 3d autonomous rotorcraft navigation in urban environments. In: AIAA Guidance, Navigation and Control Conference and Exhibit, pp. 7412.
Valente, J., Del Cerro, J., Barrientos, A., Sanz, D., 2013. Aerial coverage optimization in precision agriculture management: a musical harmony inspired approach. Comput. Electron. Agric. 99, 153–159.
Vandapel, N., Kuffner, J., Amidi, O., 2005. Planning 3-d path networks in unstructured environments. In: Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on. IEEE, pp. 4624–4629.
Vascik, P.D., 2017. Systems-level Analysis of On Demand Mobility for Aviation (Ph.D. thesis). Massachusetts Institute of Technology.
Wasiuk, D., Lowenberg, M., Shallcross, D., 2015. An aircraft performance model implementation for the estimation of global and regional commercial aviation fuel burn and emissions. Transport. Res. Part D: Transp. Environ. 35.
Wein, R., Van den Berg, J.P., Halperin, D., 2007. The visibility–Voronoi complex and its applications. Comput. Geomet. 36 (1), 66–87.
Wolek, A., Woolsey, C.A., 2017. Model-Based Path Planning. Springer International Publishing, Cham pp. 183–206.
Yang, L., Qi, J., Song, D., Xiao, J., Han, J., Xia, Y., 2016. Survey of robot 3d path planning algorithms. J. Control Sci. Eng. 2016, 5.