Numerical Algorithms
https://doi.org/10.1007/s11075-021-01167-w

ORIGINAL PAPER

A minimum-time obstacle-avoidance path planning algorithm for unmanned aerial vehicles

Arturo De Marinis[1] · **Felice Iavernaro**[1] · **Francesca Mazzia**[2]

Received: 17 March 2021 / Accepted: 2 July 2021 / Published online: 12 October 2021

Abstract
In this article, we present a new strategy to determine an unmanned aerial vehicle trajectory that minimizes its flight time in presence of avoidance areas and obstacles. The method combines classical results from optimal control theory, i.e. the Euler-Lagrange Theorem and the Pontryagin Minimum Principle, with a continuation technique that dynamically adapts the solution curve to the presence of obstacles. We initially consider the two-dimensional path planning problem and then move to the three-dimensional one, and include numerical illustrations for both cases to show the efficiency of our approach.

Keywords Path planning · Minimum-time trajectory · Obstacle avoidance · Pontryagin minimum principle · Continuation technique · UAV

Mathematics Subject Classification (2010) 65L10 · 65L04 · 49M25

1 Introduction

Unmanned aerial vehicle (UAV) path planning is a current research topic having the purpose of making the UAV capable of performing a given mission autonomously. This topic has recently attracted the attention of many researchers due to the increasing number of potential civilian and military UAV applications, e.g. environmental

✉ Felice Iavernaro
felice.iavernaro@uniba.it

Arturo De Marinis
a.demarinis18@studenti.uniba.it

Francesca Mazzia
francesca.mazzia@uniba.it

[1] Dipartimento di Matematica, Università di Bari, Bari, Italy

[2] Dipartimento di Informatica, Università di Bari, Bari, Italy

monitoring, monitoring of areas affected by natural disasters such as earthquakes, floods, and fires, search and rescue operations in emergency situations, and remote sensing to create maps identifying areas of interest. Modern mission planning tools require the optimization of different UAV performances, according to the given mission specifications, e.g. the flight time or the energy consumption in terms of fuel or battery power energy. A common issue in almost every path generation strategy is the presence of interdicted areas or obstacles in urban or orographic environments. Therefore, obstacle avoidance methods turn out to be crucial in UAV path planning (some details may be found in [1–3]). Recent techniques include cell decomposition, roadmap, artificial potential field, potential flow and optimal control methods, among others.

In cell decomposition methods, the operational area is partitioned into non-overlapping similar shaped small regions called cells. The trajectory is generated connecting by straight lines the cell centres from the starting point to the arrival point using search algorithms like A*, PSO (Particle Swarm Optimization), RRT (Rapidly-exploring Random Tree) [4–7]. The cells occupied by obstacles are excluded in the trajectory generation process. In roadmap methods, a network of straight lines connecting the starting point to the arrival point without passing through any obstacle (a roadmap) is generated. Then a search algorithm is employed to find the path which best fits the criteria required by the given mission [7, 8]. In artificial potential field methods, the arrival point is treated as an attractive potential and the obstacles are treated as repulsive potentials. An artificial potential force is then computed and applied to the UAV as a control input. Therefore, the UAV is attracted towards the arrival point and repelled by the obstacles [9–11]. In potential flow methods, obstacle avoidance path planning is based on the concept that a uniform fluid flow creates a natural path around an obstacle [12].

Optimal control methods form an important framework to address obstacle avoidance path planning problems. They are divided into two main categories: direct methods and indirect methods [13–15]. We will recall the ideas behind them in the next section. In this paper, we focus our attention on the indirect approach which relies on two results — the Euler-Lagrange Theorem and the Pontryagin Minimum Principle [13, 16] — stating necessary conditions for the optimal solution. Using these conditions, a Hamiltonian boundary value problem (BVP) is built, and among its solutions there is the optimal one. In more detail, the problem of our interest is the determination of a UAV trajectory that minimizes the associated flight time in presence of avoidance areas and obstacles. An example addressing a similar problem via the indirect approach is [17], where the constrained optimal control equations have been cast as an unconstrained system formulated in such a way as to preserve the information on the constraints. However, the numerical solutions presented, obtained by means of the MATLAB code `bvp4c`, do not completely satisfy the constraints.

It should be noticed that the general purpose solvers available in most numerical computing environments often lack robustness and efficiency in handling the Hamiltonian BVPs emerging from the above procedure. A further delicate aspect concerns the choice of the initial guess to be used during the solution of the nonlinear problems arising from the discretization of the BVP, especially in presence of multiple local solutions. These drawbacks have somehow discouraged the study of indirect

methods in favour of direct ones. Our approach with the indirect method is motivated by a consolidated experience in the computer simulation of Hamiltonian dynamical systems and in the numerical solution of BVPs [18–25]. We propose the use of the code `bvptwp`, available in MATLAB [26], Fortran and R [27–30], to efficiently solve BVPs arising from the application of indirect methods. Furthermore, we employ a new homotopy continuation technique to overcome the problem of selecting a suitable initial profile to get the optimal trajectory. The combination of these tools allows us to efficiently deal with several involved scenarios.

With this premise, the paper is organized as follows. In Section 2, we recall those concepts of optimal control theory that will be exploited later in deriving the Hamiltonian BVP. In Section 3, we study the two-dimensional path planning problem, the method we have realized to solve it, and the corresponding simulation results. Section 4 deals with a generalization to the three-dimensional path planning problem. Finally, in Section 5, we have summarized our findings.

2 Background

2.1 Optimal control problem definition

Let us consider a controlled dynamical system given by the set of ordinary differential equations

$$\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t), \mathbf{u}(t)), \tag{1a}$$

coupled with separated boundary conditions

$$\mathbf{x}(t_0) = \mathbf{x}_0, \tag{1b}$$

$$\boldsymbol{\Psi}(t_f, \mathbf{x}(t_f)) = 0, \tag{1c}$$

where

- t is the time variable; t_0 and t_f are the initial and final times respectively — notice that t_f may be unknown since, for example, it is what we want to minimize;
- $\mathbf{x}(t) \in \mathbb{R}^n$ is the vector of the state variables; $\mathbf{x}_0$ and $\mathbf{x}(t_f)$ are the system initial and final states respectively — notice that $\mathbf{x}(t_f)$ may be implicitly defined, since $\boldsymbol{\Psi}$ in (1c) may be a nonlinear function;
- $\mathbf{u}(t) \in \mathbb{R}^m$ is the vector of the control variables, i.e. time functions provided as input to the system to control its evolution over time, hereafter simply referred to as the control;
- $\mathbf{f} : [t_0, t_f] \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is a suitably regular function defining the vector field of the controlled dynamical system.

The control $\mathbf{u}(t)$ influences the performance of the dynamical system by means of the functional

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

called *cost functional* or *performance index*. This functional consists of the sum of two terms: the first one solely depends on the system final state, while the second one

takes into account the overall system evolution over time. The integrand function L in the second term is called Lagrangian function. An optimal control problem is an optimization problem which consists in determining the control $\mathbf{u}^*(t)$ that minimizes or maximizes the performance index. Without loss of generality, we can always think of an optimal control problem as a minimization problem:

$$\min_{\mathbf{u}} \phi(t_f, \mathbf{x}(t_f)) + \int_{t_0}^{t_f} L(t, \mathbf{x}(t), \mathbf{u}(t))\mathrm{d}t,$$

where the state variables $\mathbf{x}(t)$, for a given control $\mathbf{u}(t)$, must satisfy (1a)–(1c).

2.2 Methods for solving optimal control problems

The best known methods in the literature to solve an optimal control problem are divided into two categories: direct and indirect methods.

The basic idea of direct methods consists in transforming the original problem into a standard nonlinear programming problem. This is achieved in two sequential phases, according to the *discretize then optimize* paradigm. First, the problem is discretized by introducing a temporal mesh and approximating the solution $\mathbf{x}(t)$ and the control $\mathbf{u}(t)$ by means of predetermined piecewise polynomial functions. Then, appropriate collocation conditions are imposed on them in order to obtain a solution that accurately approximates the continuous model. The collocation conditions define a finite set of equality constraints which, together with the cost functional, constitute a nonlinear programming problem. For more information on direct methods, we recommend a paper by Matthew Kelly [15], that covers all of the basics required to understand and implement direct collocation methods and collects a number of interesting references. Recent results about convergence of direct methods can be found in [31]. Interesting comparisons between direct and indirect methods are presented in [32, 33].

Indirect methods, on the other hand, follow the *optimize then discretize* paradigm. They are based on the variational calculus and build a Hamiltonian BVP, whose set of solutions includes the optimal one. Below, we recall the resolution procedure.

Let us introduce the Hamiltonian function

$$H(t, \mathbf{x}(t), \mathbf{u}(t), \boldsymbol{\lambda}(t)) = L(t, \mathbf{x}(t), \mathbf{u}(t)) + \boldsymbol{\lambda}(t) \cdot \mathbf{f}(t, \mathbf{x}(t), \mathbf{u}(t)),$$

where $\boldsymbol{\lambda}(t) = (\lambda_1(t), \lambda_2(t), \ldots, \lambda_n(t))$ is the vector of the so-called costate variables. Let us denote by $H_{\mathbf{x}}$, $H_{\mathbf{u}}$ the gradient of H with respect to $\mathbf{x}$ and $\mathbf{u}$, respectively, and by $\mathbf{u}^*(t)$ the optimal control and by $\mathbf{x}^*(t)$ the corresponding state. Then the Euler-Lagrange theorem [16, pp. 45–56] states that, if $\mathbf{f}$, L, ϕ and $\boldsymbol{\Psi}$ are sufficiently smooth, then $\boldsymbol{\lambda}(t)$ is the solution of the adjoint differential equation

$$\dot{\boldsymbol{\lambda}}(t) = -H_{\mathbf{x}}(t, \mathbf{x}^*(t), \mathbf{u}^*(t), \boldsymbol{\lambda}(t)), \tag{3}$$

and $\mathbf{u}^*(t)$ and $\mathbf{x}^*(t)$ satisfy the algebraic equation system

$$H_{\mathbf{u}}(t, \mathbf{x}^*(t), \mathbf{u}^*(t), \boldsymbol{\lambda}(t)) = 0, \tag{4}$$

from which we derive the optimal control $\mathbf{u}^*(t)$, and the transversality condition

$$H(t_f)\mathrm{d}t_f - \boldsymbol{\lambda}(t_f) \cdot \mathrm{d}\mathbf{x}_f + \mathrm{d}\phi_f = 0, \tag{5}$$

that yields the boundary conditions for the costate variables. Here, to simplify the notation, we have set

$$H(t_f) = H(t_f, \mathbf{x}^*(t_f), \mathbf{u}^*(t_f), \boldsymbol{\lambda}(t_f)), \quad \text{and} \quad \phi_f = \phi(t_f, \mathbf{x}^*(t_f)).$$

Equation (4) admits, in general, multiple solutions but, exploiting the Pontryagin Minimum Principle [16, pp. 95–101], the optimal control $\mathbf{u}^*(t)$ must minimize the Hamiltonian function, evaluated at $\mathbf{x}^*(t)$, among all the admissible controls $\mathbf{u}(t)$:

$$H(t, \mathbf{x}^*(t), \mathbf{u}^*(t), \boldsymbol{\lambda}(t)) \leq H(t, \mathbf{x}^*(t), \mathbf{u}(t), \boldsymbol{\lambda}(t)).$$

Equations (3) and (4) are called Euler-Lagrange equations. Once the optimal control $\mathbf{u}^*(t)$ is determined from (4), its expression is plugged into (1a) and (3) to obtain a system of ordinary differential equations for the state and the costate variables

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{f}(t, \mathbf{x}(t), \mathbf{u}^*(t)), \\ \dot{\boldsymbol{\lambda}}(t) &= -H_{\mathbf{x}}(t, \mathbf{x}(t), \mathbf{u}^*(t), \boldsymbol{\lambda}(t)), \end{aligned}$$

with the boundary conditions given by (1b), (1c) and (5)

$$\begin{aligned} &\mathbf{x}(t_0) = \mathbf{x}_0, \quad \boldsymbol{\Psi}(t_f, \mathbf{x}(t_f)) = 0, \\ &H(t_f)\mathrm{d}t_f - \boldsymbol{\lambda}(t_f) \cdot \mathrm{d}\mathbf{x}_f + \mathrm{d}\phi_f = 0. \end{aligned}$$

Thus, we have got a Hamiltonian BVP that we can solve numerically. In our application context, the above BVP will be integrated with a set of constraints representing interdicted areas that the solution trajectory should not cross. A penalty function approach is then exploited to incorporate these additional requirements inside the original cost functional (2). As we will see in the next section, these penalty functions add singularities in the problem, so that its numerical solution has to be handled with care. Hereafter, we give a brief description of the code we have considered for this purpose.

2.3 Numerical solution of the boundary value problem

It is well known that the two most involved implementation issues in devising general purpose codes for BVPs are the mesh selection strategy and the efficient solution of the nonlinear system arising from the discretization of the continuous problem by means of a suitable numerical method. The presence of singularities makes such aspects even more relevant, since they heavily influence the behaviour of the resulting procedure.

For our numerical simulations, we have considered the MATLAB environment [34] and the code `bvptwp`, which is a MATLAB translation of the Fortran codes `twpbvpc`, `twpbvplc` and `acdcc` [26, 30, 35]. The code `bvptwp` allows the user to choose between two techniques, one of which gets information about the conditioning of the problem (see [21, 25] for a complete description). This particular feature is exploited in the mesh selection strategy, which is able to adapt the mesh points in order to cope with specific conditioning issues that may emerge during the solution of the BVP, especially when the trajectory gets close to a singular point. The code incorporates two deferred correction schemes, the former based on Mono-Implicit Runge-Kutta (MIRK) methods and the latter on Lobatto formulae. In our

simulations, we have used the implementation based on Lobatto formulae, since they are more efficient for stiff problems and are more appropriate for problems with a Hamiltonian structure. The MATLAB and Fortran release of the codes are freely available at the url [36] together with many test examples (see also [28]). An interface of the Fortran codes in the R environment is also available [29].

For comparison purposes we have also considered the MATLAB built-in functions `bvp4c` and `bvp5c`. These latter are very efficient for the solution of smooth problems, but suffer from lack of robustness when applied to singularly perturbed problems (see Example 1 in Section 3.4).

Finally, it is worth mentioning that a BVP can admit more than one solution. In this case, these solutions represent local minima for the original problem, among which the global minimum is hidden. Since the solution calculated by a numerical method depends on the initial guess, particular attention must be paid to the choice of the latter. We give a detailed description of this aspect in the next section.

3 2D path planning problem

Let us turn to the problem of our interest, i.e. to determine a UAV trajectory that minimizes the flight time in presence of avoidance areas. In general, the UAV motion is a three-dimensional motion. However, some salient flight phases occur at a constant altitude, i.e. in a plane. Therefore, we first consider the two-dimensional case, and later we will move to the three-dimensional one.

3.1 Problem formulation

The problem we intend to solve is the determination of the two-dimensional trajectory that minimizes the UAV flight time from a starting point (x_0, y_0) to an end point (x_f, y_f) in presence of avoidance areas. As a working assumption, we neglect the rigid body structure of the UAV, which we represent as a material point, corresponding to its centre of mass.[1] Furthermore, we assume that the UAV motion occurs with constant velocity in modulus V. Without loss of generality, we suppose that the initial time is $t_0 = 0$. Thus, the UAV motion is described by the ordinary differential equations

$$\begin{aligned} \dot{x} &= V\cos\theta, \\ \dot{y} &= V\sin\theta, \end{aligned}$$

with boundary conditions

$$\begin{aligned} x(0) &= x_0, \quad y(0) = y_0, \\ x(t_f) &= x_f, \quad y(t_f) = y_f, \end{aligned}$$

where

[1] This assumption is appropriate, for example, when the UAV has a symmetrical structure with respect to its centre of mass.

- $x = x(t)$, $y = y(t)$, respectively the abscissa and the ordinate of the UAV in an appropriate Cartesian reference, are the state variables;
- $\theta = \theta(t) \in [-\pi, \pi]$, the yaw angle (Fig. 1), which describes the UAV rotation around the vertical axis passing through its centre of mass, is the control variable.

Hence, we do not consider neither the pitch angle, which describes the UAV rotation around the transverse axis passing through its centre of mass, the motion being two-dimensional, nor the roll angle, which describes the UAV rotation around the longitudinal axis passing through its centre of mass, since we are neglecting the rigid body structure of the UAV. For simplicity, we will not introduce constraints on the control variable.

The performance index to be minimized is the flight time (observe that the final time t_f dependence on the control is not explicit)

$$J(\theta) = \int_0^{t_f(\theta)} \mathrm{d}t = t_f(\theta).$$

Regarding the avoidance areas, since we are working in the two-dimensional space $\mathbb{R}^2$, we enclose them in ellipses in order to obtain sufficiently regular constraints. Therefore, the set of constraints with which we approximate the avoidance areas is

$$g_i(x, y) \geq 0, \quad i = 1, \ldots, p,$$

where

$$g_i(x, y) = \frac{(x - x_i)^2}{a_i^2} + \frac{(y - y_i)^2}{b_i^2} - 1,$$

and x_i, y_i, a_i and b_i are, respectively, the centre coordinates and the axis lengths of the i-th ellipse. Furthermore, we also consider ellipses whose first axis is

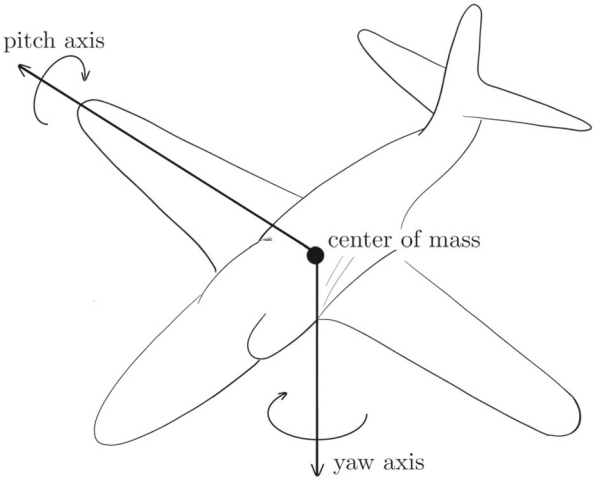

Fig. 1 Yaw and pitch angles governing the motion of a UAV (the roll angle is not illustrated, since the UAV is assimilated to a material point)

rotated counterclockwise by an angle α with respect to the x-axis applying the transformation

$$\begin{aligned} x' &= x\cos\alpha + y\sin\alpha, \\ y' &= -x\sin\alpha + y\cos\alpha. \end{aligned}$$

In conclusion, the problem of our interest is the following:

$$\min_{\theta} \int_0^{t_f(\theta)} \mathrm{d}t,$$

where the state variables $x(t)$ and $y(t)$, given a control $\theta(t)$, must satisfy the equations

$$\begin{aligned} &\dot{x} = V\cos\theta, \\ &\dot{y} = V\sin\theta, \\ &x(0) = x_0, \quad y(0) = y_0, \\ &x(t_f) = x_f, \quad y(t_f) = y_f, \end{aligned}$$

and the additional constraints

$$g_i(x, y) \geq 0, \quad i = 1, \ldots, p. \tag{6}$$

This problem is a constrained optimal control problem because of the additional algebraic constraints (6), so we cannot solve it directly using the Euler-Lagrange Theorem and the Pontryagin Minimum Principle. Let us frame an auxiliary unconstrained optimal control problem formulated in such a way as to preserve the information on the additional constraints, with which we approximate the above-mentioned problem. After that, we shall devise a continuation technique to solve the auxiliary problem.

3.2 Auxiliary problem formulation

For each constraint, let us define the function

$$h_i(x, y; k_i) = \frac{k_i}{g_i(x, y)}, \quad i = 1, \ldots, p,$$

where $k_i > 0$ is a parameter. The function h_i has a singularity on the boundary of the i-th ellipse, is positive outside the ellipse and negative inside it, and is close to zero outside a neighbourhood of the ellipse. We can adjust the size of this neighbourhood by tuning the parameter k_i. Then, let us define the function

$$P(x, y; k_1, \ldots, k_p) = \sum_{i=1}^{p} h_i(x, y; k_i),$$

and the augmented Lagrangian function

$$L(x, y; k_1, \ldots, k_p) = 1 + P(x, y; k_1, \ldots, k_p),$$

and let us consider the unconstrained optimal control problem

$$\min_{\theta} \int_0^{t_f(\theta)} \big(1 + P(x(t), y(t); k_1, \ldots, k_p)\big)\mathrm{d}t, \tag{7}$$

where the state variables $x(t)$ and $y(t)$, given a control $\theta(t)$, must satisfy the equations

$$\begin{aligned} &\dot{x} = V\cos\theta, \\ &\dot{y} = V\sin\theta, \\ &x(0) = x_0, \quad y(0) = y_0, \\ &x(t_f) = x_f, \quad y(t_f) = y_f. \end{aligned} \tag{8}$$

Hereafter, to keep the notation simple, we will omit the explicit dependence of h_i, $i = 1, \ldots, p$, and P on the parameters $k_1, \ldots, k_p$. The new cost functional (7) approximates the flight time where $P(x, y) \approx 0$, i.e. away from the ellipses, is far greater than the flight time in a narrow neighbourhood of the ellipses, outside them, where $P(x, y) \gg 0$, and has singularities on the boundary of each ellipse.

Apart from its regularity features outside the interdicted areas, the reason for choosing the penalty function P is elucidated in the example in Fig. 2. The left picture shows the level sets of the function $P(x, y)$ corresponding to an ellipse of centre $(0, 0)$, axes of length 2, 1, whose first axis is rotated counterclockwise by an angle of $\pi/4$ radians with respect to the x-axis, and to a circumference of centre $(2.5, 1)$ and radius 0.5, with $k_1 = k_2 = 10^{-3}$. We see that $P(x, y)$ assumes small (positive) values outside a narrow neighbourhood of the two ellipses, while it diverges to infinity when (x, y) approaches the boundary of each ellipse. This latter aspect is better visible in the right plot of Fig. 2, that shows the graph of the function P along the straight line joining the centres of the two ellipses parameterized with respect to the abscissa ($x = t, y = t/2.5, t \in [0, 1]$) and confined to the segment QR lying in the fly zone. By virtue of (7), the two asymptotes act as barriers to prevent that the UAV may cross the ellipses while, at the same time, the trajectory may approach the ellipses closely as far as $|P(x, y)| \ll 1$.

In conclusion, when we apply the Euler-Lagrange Theorem and the Pontryagin Minimum Principle to minimize the modified cost functional (7) with respect to the control, we get an optimal control which makes the UAV trajectory to possibly approach an ellipse very closely without touching it, so that the cost functional essentially returns the flight time, provided that the parameters k_i are chosen small enough.

However, when solving the BVP numerically, the above argument partially evaporates due to the discrete nature of the numerical solution. In fact, while a continuous trajectory $(x(t), y(t))$ crossing an ellipse would necessarily encounter a singular point of the cost functional, the same argument does not apply for the discrete orbit $(x_j, y_j) \approx (x(t_j), y(t_j))$ unless the stepsize of integration is chosen sufficiently small, which would dramatically affect the efficiency of the overall procedure. In order to prevent this situation from occurring, we have implemented a continuation technique, which generates a preliminary trajectory considering null axes for each ellipse. Subsequently, the axes are gradually increased until they reach their final values and, at each step, the solution obtained at the previous step is readapted in order to satisfy the constraints at the current step. A formal description of this continuation technique is reported below. Hereafter we apply the Euler-Lagrange Theorem,

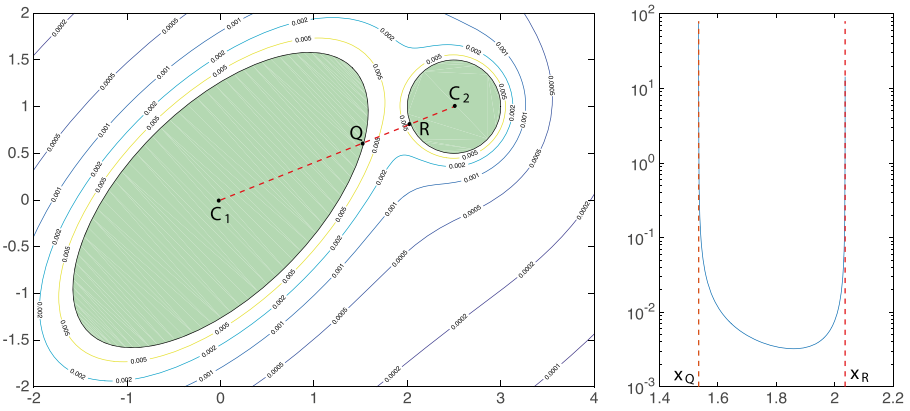

Fig. 2 Level sets of the function $P(x, y)$ corresponding to an ellipse of centre $(0, 0)$, axes of length 2, 1, whose first axis is rotated counterclockwise by an angle of $\pi/4$ radians with respect to the x-axis, and to a circumference of centre $(2.5, 1)$ and radius 0.5, with $k_1 = k_2 = 10^{-3}$

[16, pp. 45–56], and the Pontryagin Minimum Principle [16, pp. 95–101], to solve problem (7) and (8). Defining the Hamiltonian function

$$H = 1 + P(x, y) + \lambda_1 V \cos\theta + \lambda_2 V \sin\theta,$$

the Euler-Lagrange equations become

$$\begin{aligned} \dot{\lambda_1} &= -H_x = -P_x(x, y), \\ \dot{\lambda_2} &= -H_y = -P_y(x, y), \\ H_\theta &= -\lambda_1 V \sin\theta + \lambda_2 V \cos\theta = 0. \end{aligned} \tag{9}$$

The optimal control is not uniquely defined by (9), since

$$\cos\theta = \pm\frac{\lambda_1}{\sqrt{\lambda_1^2 + \lambda_2^2}}, \quad \sin\theta = \pm\frac{\lambda_2}{\sqrt{\lambda_1^2 + \lambda_2^2}},$$

both satisfy (9). Therefore, according to the Pontryagin Minimum Principle, we choose the optimal control corresponding to the negative solution. The transversality condition reads

$$H(t_f)\mathrm{d}t_f - \lambda_1(t_f)\mathrm{d}x_f - \lambda_2(t_f)dy_f + \mathrm{d}\phi_f = 0.$$

Since the final state is given and $\phi = 0$, we have that $dx_f = dy_f = d\phi_f = 0$, and the transversality condition reduces to $H(t_f)\mathrm{d}t_f = 0$. It has to be satisfied for all $\mathrm{d}t_f$, thus we arrive at the additional boundary condition $H(t_f) = 0$ for the costate variables.

Let us recall that $t_f = t_f(\theta)$ is unknown: indeed, it is the performance index to be minimized. By setting $s = t/t_f$, the integration interval becomes $[0, 1]$, t_f becomes a further state variable and we obtain the additional equation $\dot{t_f} = 0$.

Therefore, the state and the costate variables are the solution of the system of ordinary differential equations

$$\begin{aligned}
\dot{x} &= V\left(-\frac{\lambda_1}{\sqrt{\lambda_1^2+\lambda_2^2}}\right)t_f,\\
\dot{y} &= V\left(-\frac{\lambda_2}{\sqrt{\lambda_1^2+\lambda_2^2}}\right)t_f,\\
\dot{\lambda_1} &= (-P_x(x,y))\,t_f,\\
\dot{\lambda_2} &= \big(-P_y(x,y)\big)\,t_f,\\
\dot{t_f} &= 0,
\end{aligned} \tag{10}$$

coupled with the boundary conditions

$$x(0) = x_0, \quad y(0) = y_0, \quad x(1) = x_f, \quad y(1) = y_f, \tag{11}$$

and

$$\begin{aligned}
H(1) &= 1 + P(x_f, y_f) + \lambda_1(1)\left(-\frac{V\lambda_1(1)}{\sqrt{\lambda_1(1)^2+\lambda_2(1)^2}}\right)\\
&\quad +\lambda_2(1)\left(-\frac{V\lambda_2(1)}{\sqrt{\lambda_1(1)^2+\lambda_2(1)^2}}\right) = 0.
\end{aligned} \tag{12}$$

3.3 The continuation technique

For all $i = 1, \ldots, p$, let N_i be a positive integer, $N = \max\limits_{1\le i\le p} N_i$, and define the sequence of constraint functions

$$g_i^{(n)}(x, y; N_i) = \frac{(x-x_i)^2}{a_i^2} + \frac{(y-y_i)^2}{b_i^2} - \frac{n}{N_i}, \quad n = 0, \ldots, N_i.$$

The equation $g_i^{(n)}(x, y; N_i) = 0$ represents an ellipse of centre (x_i, y_i) and axes $a_i\sqrt{n/N_i}$, $b_i\sqrt{n/N_i}$. When $n = 0$, each ellipse shrinks to its centre, while its axes increase with n until, for $n = N_i$, they match the values a_i and b_i, yielding the original shape. Let us define the functions

$$h_i^{(n)}(x, y; k_i, N_i) = \frac{k_i}{g_i^{(n)}(x, y; N_i)}, \quad n = 0, \ldots, N_i,$$

and

$$P^{(n)}(x, y; k_1, \ldots, k_p, N_1, \ldots, N_p) = \sum_{i=1}^{p} h_i^{(n)}(x, y; k_i, N_i), \quad n = 0, \ldots, N,$$

where, after noticing that $g_i^{(N_i)}(x, y; N_i) = g_i(x, y)$ and $h_i^{(N_i)}(x, y; k_i, N_i) = h_i(x, y)$, we simply set $g_i^{(n)}(x, y; N_i) = g_i(x, y)$ for $n > N_i$, so $h_i^{(n)}(x, y; k_i, N_i) = h_i(x, y)$ and $P^{(N)}(x, y; k_1, \ldots, k_p, N_1, \ldots, N_p) = P(x, y)$. In the sequel, to

simplify the notation, we will omit the explicit dependence of $g_i^{(n)}$, $h_i^{(n)}$, $i = 1, \ldots, p$, and $P^{(n)}$, $n = 0, \ldots, N$, on the parameters $k_1, \ldots, k_p, N_1, \ldots, N_p$.

The continuation technique consists in solving the BVPs (10)–(12) with $P(x, y)$ replaced by $P^{(n)}(x, y)$, sequentially for $n = 0, \ldots, N$. The solution obtained after solving the n-th problem is used in the code as initial guess to solve the subsequent BVP. At the very first step ($n = 0$), the initial guess is just the straight line joining (x_0, y_0) and (x_f, y_f), i.e. the optimal trajectory in absence of constraints. In principle, it may happen that an obstacle centre lies on the initial guess or is very close to it. In this case, the singularity introduced in that point prevents the continuation from starting since $P^{(0)}(x, y)$ is Infinity. There are several ways to overcome this issue. We have chosen to slightly move the obstacle centre along the line orthogonal to the initial guess, and to increase its axes in such a way that the new adjusted obstacle contains the original one.

The above procedure guarantees a fast convergence of the nonlinear scheme solver embedded in the code towards the local solution which is closest to the initial profile given in input. From a geometrical viewpoint, the continuation procedure defines a sequence of homotopic solution curves in the phase plane, originating from the degenerate case where all the constraints reduce to points. As long as the ellipses expand, the singularities introduced in the functions $P^{(n)}(x, y)$ adapt the corresponding trajectories in order to keep them outside the ellipses.

Once the solution of the last BVP ($n = N$) has been determined, the state variables $x(t)$ and $y(t)$ give the UAV trajectory, and the costate variables $\lambda_1(t)$ and $\lambda_2(t)$ give the optimal control $\theta(t) \in [-\pi, \pi]$ via the relations

$$\cos\theta(t) = -\frac{\lambda_1(t)}{\sqrt{\lambda_1(t)^2 + \lambda_2(t)^2}}, \quad \sin\theta(t) = -\frac{\lambda_2(t)}{\sqrt{\lambda_1(t)^2 + \lambda_2(t)^2}}.$$

The values of the parameters k_i and N_i are empirical and based on the numerical simulations, though we have employed a dynamic selection strategy to speed up the overall procedure, according to the following scheme:

initially, set $k_i = 0.01$ and $N_i = 1$, $i = 1, \ldots, p$;
n = 0;
while $n \leq N$, **do**:
    solve the n-th problem and get the solution trajectory $\left(x_j^{(n)}, y_j^{(n)}\right)$[2];
    for $i = 1, \ldots, p$, **do**:
        while $n < N_i$ and $\min_j g_i^{(n+1)}\left(x_j^{(n)}, y_j^{(n)}\right) < 0.02$, **do**:
            $N_i \leftarrow N_i + 1$;
            if N_i is a multiple of 10, **then**:
                $k_i \leftarrow k_i + 0.0045$;
                solve the n-th problem again;
            end if
        end while
    end for
    $n \leftarrow n + 1$.
end while

[2] The index j indicates the element of the discrete trajectory.

Notice that, at each continuation step n, the parameters k_i and N_i are increased until the minimum of $g_i^{(n+1)}(x, y)$ computed along the n-th solution trajectory $\left(x_j^{(n)}, y_j^{(n)}\right)$ is greater than or equal to 0.02, for all $i = 1, \ldots, p$. This check assures that the n-th solution trajectory is outside and not so close to each ellipse at step $n + 1$. This is needed because, if the n-th solution trajectory crosses some ellipses at step $n + 1$, then it crosses the singularities introduced along them, making the continuation get stuck. Furthermore, every time k_i is updated, the n-th problem has to be solved again since the functions $h_i^{(n)}(x, y)$ and $P^{(n)}(x, y)$ depend on k_i. So k_i is increased whenever N_i becomes a multiple of 10, in order to prevent the execution time from increasing too much.

Finally, let us notice that the continuation technique described so far applies to all the obstacles simultaneously. As an alternative, the continuation can be applied to an obstacle at a time, starting from the one with the largest axes to the one with the smallest axes decreasingly. To achieve this, we first define the functions

$$P_i^{(n)}(x, y) = \sum_{j=1}^{i-1} h_j(x, y) + h_i^{(n)}(x, y), \quad i = 1, \ldots, p,\ n = 0, \ldots, N_i,$$

where $P_1^{(n)}(x, y) = h_1^{(n)}(x, y)$, $n = 0, \ldots, N_1$, and $P_p^{(N_p)}(x, y) = P(x, y)$, and then, after sorting the obstacles as mentioned above, set up the following scheme:

initially, set $k_i = 0.01$ and $N_i = 1$, $i = 1, \ldots, p$;
for $i = 1, \ldots, p$, **do**:
    $n = 0$;
    while $n \leq N_i$, **do**:
        solve the n-th problem and get the solution trajectory $\left(x_j^{(i,n)}, y_j^{(i,n)}\right)$;
        while $n < N_i$ and $\min_j g_i^{(n+1)}\left(x_j^{(i,n)}, y_j^{(i,n)}\right) < 0.02$, **do**:
            $N_i \leftarrow N_i + 1$;
            if N_i is a multiple of 10, **then**:
                $k_i \leftarrow k_i + 0.0045$;
                solve the n-th problem again;
            end if
        end while
        $n \leftarrow n + 1$.
    end while
end for

Therefore, this variant consists in solving the BVPs (10)–(12) with $P(x, y)$ replaced by $P_i^{(n)}(x, y)$, sequentially for $i = 1, \ldots, p$, and sequentially for $n = 0, \ldots, N_i$. The solution obtained after solving the n-th problem in the i-th iteration is used in the code as initial guess to solve

- the subsequent BVP at step $n + 1$, if $n < N_i$;
- the first problem in the $(i + 1)$-th iteration, if $n = N_i$.

The initial guess for $i = 1$ and $n = 0$, as well as the parameter initialization and tuning, is as before.

3.4 Simulation results

In this subsection, we report two examples showing the results of the two-dimensional numerical simulations we have carried out. For the first example, we compare the results given by the two continuation strategies, in terms of the corresponding flight time, as well as the performance of the code `bvptwp` with that of the codes `bvp4c` and `bvp5c`. As comparison criteria, we have considered, for the computed solution of the last BVP in the continuation process, the number of points in the final output mesh (nMeshPoints) and the total number of function evaluations defining the ordinary differential equation system (nODEevals) and the boundary conditions (nBCevals). In addition, we have reported the execution time expressed in seconds (time) for the overall continuation. All computations have been carried out on an Intel i7 quad-core CPU with 16GB of memory, running MATLAB R2020b.

For both examples, we assume that the UAV must move from the point $(x_0, y_0) = (0, 0)$ to the point $(x_f, y_f) = (10, 10)$ with constant velocity in modulus $V = 1$, while the avoidance areas are disjoined and defined as follows.

– Example 1: Three circumferences having radius 1 and centres $(2.2, 1.8)$, $(3.8, 4.2)$ and $(6.2, 3.8)$, respectively; an ellipse having centre $(7.4, 7.8)$ and axes of length 2.25 and 1.5, whose first axis is rotated counterclockwise by an angle of $\pi/4$ radians with respect to the x-axis. The final parameter values of the continuation strategies for the codes `bvptwp`, `bvp4c` and `bvp5c` are reported in Table 1. The corresponding trajectory and the yaw angle that realizes it, expressed in radians, are shown in Figs. 3 and 4 for the two continuation strategies. Table 2 summarizes the performance of the code `bvptwp` in terms of the output parameters listed at the beginning of this subsection, and compares its behaviour with that of the MATLAB built-in codes `bvp4c` and `bvp5c`, showing its efficiency and robustness.

  This example has been suitably conjectured in such a way that the two continuation strategies could lead to two different local minima of the underlying

Table 1 Example 1: Final parameter values of the two continuation strategies

First continuation strategy									
Code	k_1	k_2	k_3	k_4	N_1	N_2	N_3	N_4	Flight time
`bvptwp`	0.01	0.01	0.01	0.01	9	8	1	6	15.1876
`bvp4c/bvp5c`	0.0145	0.01	0.01	0.01	10	8	1	6	15.2053
Second continuation strategy									
Code	k_1	k_2	k_3	k_4	N_1	N_2	N_3	N_4	Flight time
`bvptwp`	0.01	0.01	0.01	0.01	8	1	6	5	14.9766
`bvp4c/bvp5c`	0.01	0.01	0.01	0.01	8	1	6	5	14.9766

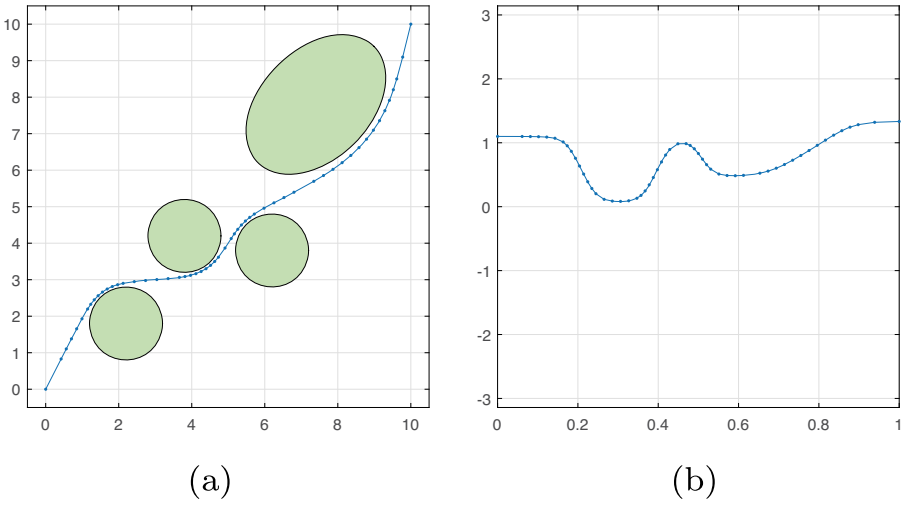

Fig. 3 2D Example 1, first continuation strategy: optimal trajectory (a) and yaw angle that realizes it (b)

optimization problem, which is not the case for general problems. In the following example, the first continuation strategy even fails to converge, which makes the second strategy more robust, despite a bit more expensive.

– Example 2: Three circumferences having radius 1 and centres $(1.8, 1.4)$, $(2.8, 3.8)$ and $(4.6, 5.4)$, respectively; an ellipse having centre $(7.6, 7.2)$ and axes of length 1.75 and 3.5, whose first axis is rotated counterclockwise by an angle of $\pi/4$ radians with respect to the x-axis. Using the second continuation strategy and the code `bvptwp`, the final parameter values are $(k_1, k_2, k_3, k_4) =$

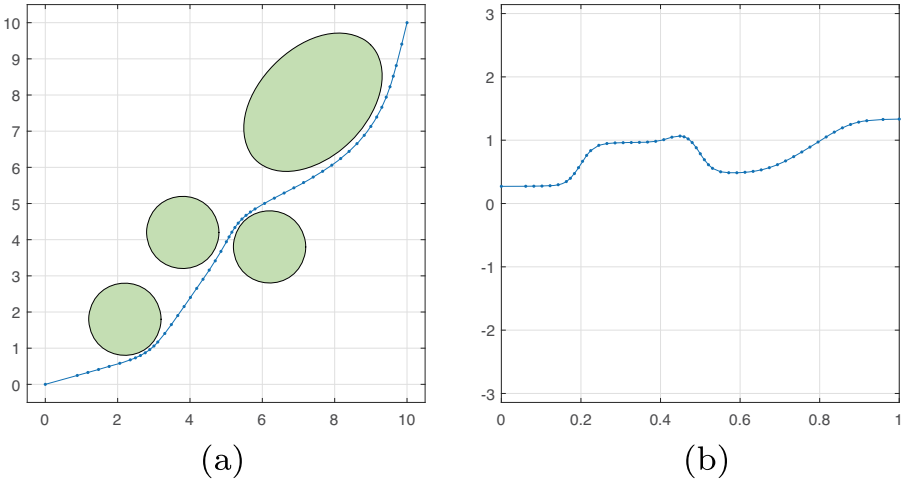

Fig. 4 2D Example 1, second continuation strategy: optimal trajectory (a) and yaw angle that realizes it (b)

Table 2 Example 1: Comparison of the codes' performance

Strategy	Code	nMeshPoints	nODEevals	nBCevals	Time
	`bvptwp`	55	1252	10	3.844797
1	`bvp4c`	179795	7415318	9	549.997793
	`bvp5c`	6201	468757	13	81.166330
	`bvptwp`	53	1206	10	6.248556
2	`bvp4c`	265868	8773065	11	477.555046
	`bvp5c`	11857	663555	17	96.182822

$(0.01, 0.01, 0.01, 0.0145)$ and $(N_1, N_2, N_3, N_4) = (1, 2, 1, 17)$, which yield an optimal trajectory with associated flight time 16.1760 (see Fig. 5).

4 3D path planning problem

The three-dimensional problem is conceptually similar to the two-dimensional one, the ordinary differential equations governing the UAV motion now being

$$\begin{aligned} \dot{x} &= V \cos\theta \cos\gamma, \\ \dot{y} &= V \sin\theta \cos\gamma, \\ \dot{z} &= V \sin\gamma, \end{aligned}$$

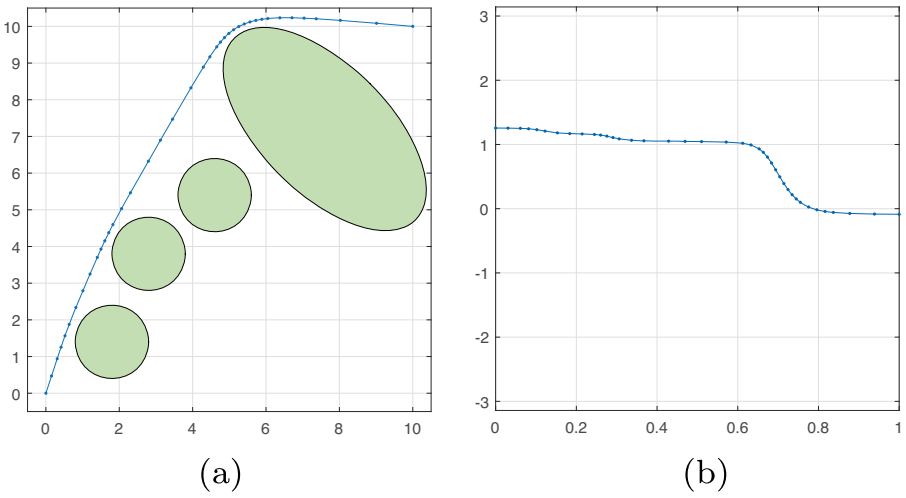

Fig. 5 Example 2, second continuation strategy: optimal trajectory (a) and yaw angle that realizes it (b)

with the boundary conditions

$$x(0) = x_0, \quad y(0) = y_0, \quad z(0) = z_0,$$
$$x(t_f) = x_f, \quad y(t_f) = y_f, \quad z(t_f) = z_f,$$

where

- $x = x(t)$, $y = y(t)$, $z = z(t)$, respectively the abscissa, ordinate and altitude of the UAV in an appropriate Cartesian reference, are the state variables;
- $\theta = \theta(t) \in [-\pi, \pi]$, $\gamma = \gamma(t) \in [-\pi/2, \pi/2]$, respectively the yaw and pitch angles, are the control variables (see Fig. 1).

We still do not consider the roll angle since we are neglecting the rigid body structure of the UAV, and we will not introduce constraints on the control variables. The performance index to be minimized is again the flight time

$$J(\theta, \gamma) = \int_0^{t_f(\theta,\gamma)} \mathrm{d}t = t_f(\theta, \gamma),$$

that now is a function of the pitch angle, too.

Regarding the avoidance areas, since we are working in the three-dimensional space $\mathbb{R}^3$, we enclose them in cylinders or ellipsoids in order to obtain sufficiently regular constraints. Therefore, the set of constraints with which we approximate the avoidance areas is

$$g_i(x, y, z) \geq 0, \quad i = 1, \ldots, p,$$
$$h_j(x, y, z) \geq 0, \quad j = 1, \ldots, q,$$

where

$$g_i(x, y, z) = \frac{(x - x_i)^2}{a_i^2} + \frac{(y - y_i)^2}{b_i^2} - 1,$$

and x_i, y_i, a_i and b_i are, respectively, the axis coordinates and the axis lengths of the i-th cylinder,

$$h_j(x, y, z) = \frac{(x - x_j)^2}{a_j^2} + \frac{(y - y_j)^2}{b_j^2} + \frac{(z - z_j)^2}{c_j^2} - 1,$$

and x_j, y_j, z_j, a_j, b_j and c_j are, respectively, the centre coordinates and the axis lengths of the j-th ellipsoid. Cylinders are suitable for areas to avoid no matter what the altitude is, while ellipsoids are suitable for finite height avoidance areas.

Thus, the optimal control problem in the three-dimensional case is the following:

$$\min_{\theta,\gamma} \int_0^{t_f(\theta,\gamma)} \mathrm{d}t,$$

where the state variables $x(t)$ and $y(t)$, given a control $(\theta(t), \gamma(t))$, must satisfy the equations

$$\begin{aligned}
&\dot{x} = V\cos\theta\cos\gamma,\\
&\dot{y} = V\sin\theta\cos\gamma,\\
&\dot{z} = V\sin\gamma,\\
&x(0) = x_0, \quad y(0) = y_0, \quad z(0) = z_0,\\
&x(t_f) = x_f, \quad y(t_f) = y_f, \quad z(t_f) = z_f,
\end{aligned}$$

and the additional constraints

$$\begin{aligned}
g_i(x, y, z) &\geq 0, \quad i = 1, \ldots, p,\\
h_j(x, y, z) &\geq 0, \quad j = 1, \ldots, q.
\end{aligned}$$

The formulation of the auxiliary unconstrained optimal control problem described in the previous section is now adapted in order to tackle the three-dimensional problem. According to the Euler-Lagrange Theorem, to determine the optimal controls θ and γ, we consider the Hamiltonian function

$$H = 1 + P(x, y, z) + \lambda_1 V\cos\theta\cos\gamma + \lambda_2 V\sin\theta\cos\gamma + \lambda_3 V\sin\gamma$$

and compute its stationary points with respect to the controls

$$\begin{aligned}
H_\theta &= -\lambda_1 V\sin\theta\cos\gamma + \lambda_2 V\cos\theta\cos\gamma = 0,\\
H_\gamma &= -\lambda_1 V\cos\theta\sin\gamma - \lambda_2 V\sin\theta\sin\gamma + \lambda_3 V\cos\gamma = 0,
\end{aligned}$$

from which we have that

$$\cos\theta = \pm\frac{\lambda_1}{\sqrt{\lambda_1^2 + \lambda_2^2}}, \quad \sin\theta = \pm\frac{\lambda_2}{\sqrt{\lambda_1^2 + \lambda_2^2}},$$

and

$$\cos\gamma = \pm\frac{\sqrt{\lambda_1^2 + \lambda_2^2}}{\sqrt{\lambda_1^2 + \lambda_2^2 + \lambda_3^2}}, \quad \sin\gamma = \pm\frac{\lambda_3}{\sqrt{\lambda_1^2 + \lambda_2^2 + \lambda_3^2}}.$$

We choose the negative solution for the yaw angle θ in order to be consistent with the two-dimensional case, the positive and negative ones for the $\cos\gamma$ and $\sin\gamma$ respectively such that the Pontryagin Minimum Principle [16, pp. 95–101] is satisfied. The continuation technique remains identical to the two-dimensional case.

Table 3 Example 3: Final parameter values of the two continuation strategies

First continuation strategy											
Code	k_1	k_2	k_3	k_4	k_5	N_1	N_2	N_3	N_4	N_5	Flight time
`bvptwp`	0.0145	0.01	0.01	0.01	0.01	10	1	7	8	9	15.3833
Second continuation strategy											
Code	k_1	k_2	k_3	k_4	k_5	N_1	N_2	N_3	N_4	N_5	Flight time
`bvptwp`	0.01	0.01	0.01	0.01	0.0145	8	1	4	6	10	15.2962

4.1 Simulation results

Among the three-dimensional numerical simulations we have carried out, we here report an example, simulating an urban environment, where the two continuation strategies behave differently, again emphasizing that for general and less involved problems, the two approaches lead to the same results. Finally, we consider a scenario reproducing an orographic environment by means of a set of ellipsoids.

– Example 3: We assume that the UAV must move from the point $(x_0, y_0, z_0) = (0, 0, 0)$ to the point $(x_f, y_f, z_f) = (10, 10, 1.5)$ with constant velocity in modulus $V = 1$, while the avoidance areas are four circular cylinders having radius 1 and axes of equations $x = 1.5,\ y = 1.2$, $x = 2.1,\ y = 3.6$, $x = 7.9,\ y = 6.4$ and $x = 8.5,\ y = 8.8$, respectively, and a circular cylinder having radius 1.5 and axis of equations $x = 4.5,\ y = 5.5$. The final parameter values and the resulting

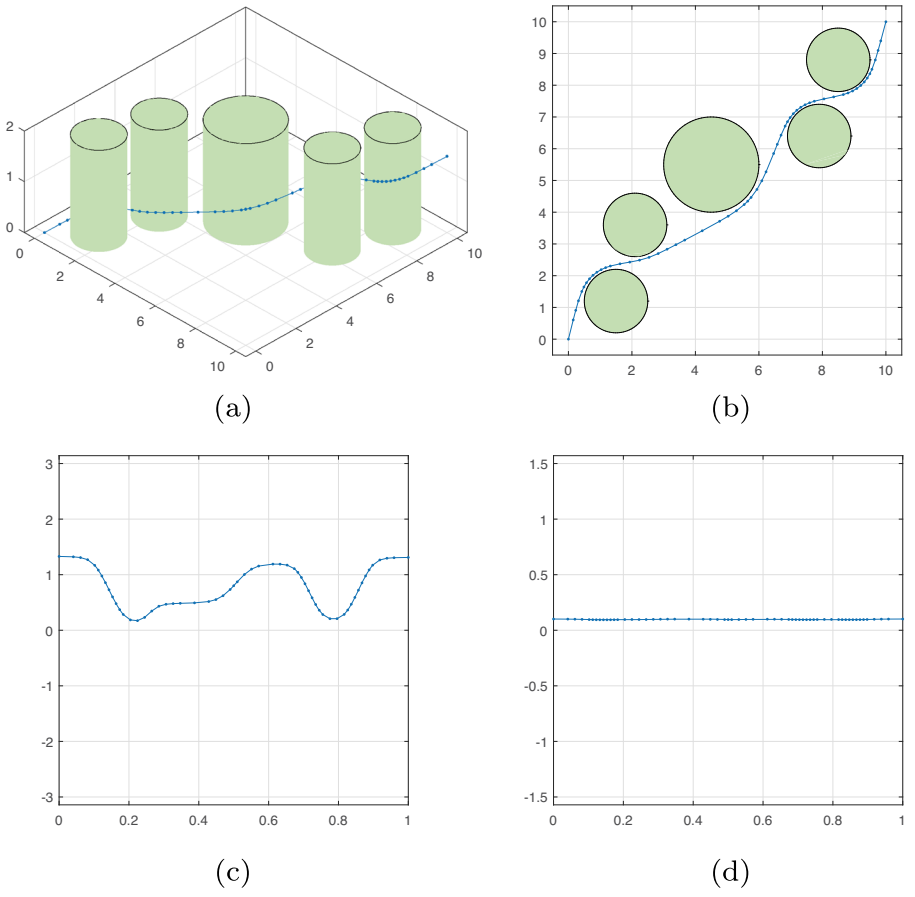

Fig. 6 Example 3, first continuation strategy: optimal trajectory (a), view from above of it (b), and yaw and pitch angles that realize it (c, d)

flight time for the two continuation strategies are reported in Table 3. The corresponding trajectory, a view from above of it, and the yaw and pitch angles that realize it, expressed in radians, are shown in Figs. 6 and 7.

– Example 4: We assume that the UAV must move from the point $(x_0, y_0, z_0) = (0, 0, 0)$ to the point $(x_f, y_f, z_f) = (6, 6, 0)$ with constant velocity in modulus $V = 1$, while the areas to avoid are three ellipsoids having centres $(2, 3.5, -0.5)$, $(3, 2, -0.5)$ and $(5.1, 4.4, -0.5)$, and axis lengths 1, 1, 4, 2.5, 1.25, 2 and 1, 1, 2 respectively (notice that the first two ellipsoids intersect). For this problem, the first continuation strategy fails to converge, so we have considered an extension of the second continuation technique to the three-dimensional case. The corresponding output parameters are $(k_1, k_2, k_3) = (0.01, 0.01, 0.01)$ and $(N_1, N_2, N_3) = (8, 8, 6)$, which yield an optimal trajectory with associated flight time $t_f = 8.9816$ (see Fig. 8).

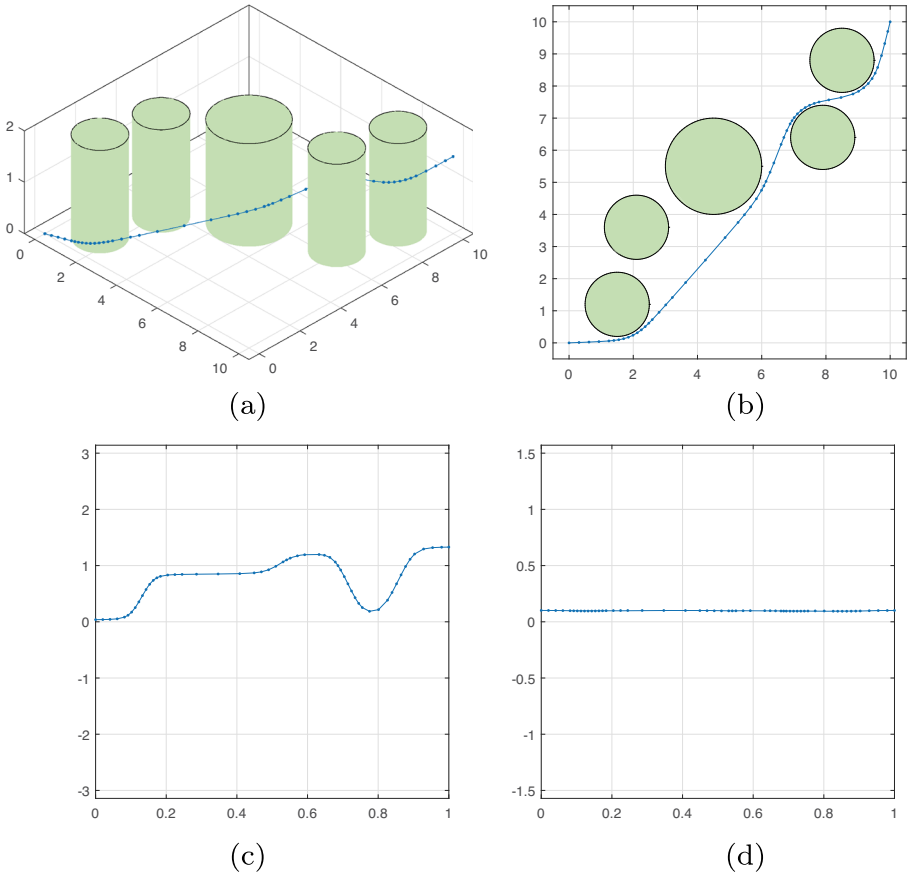

Fig. 7 Example 3, second continuation strategy: optimal trajectory (a), view from above of it (b), and yaw and pitch angles that realize it (c, d)

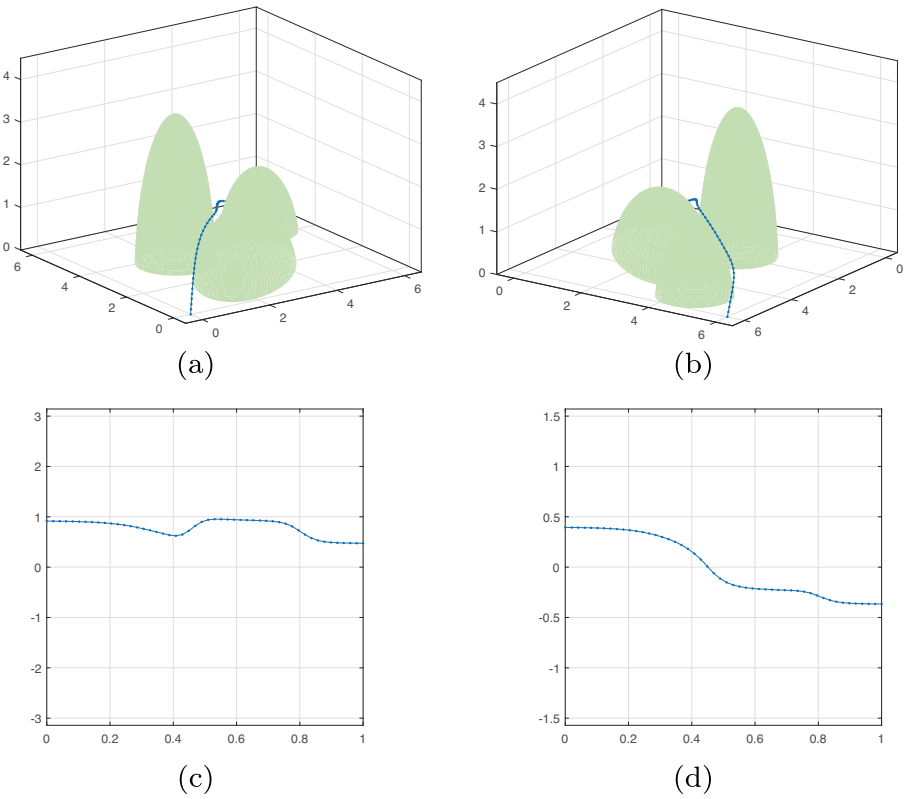

Fig. 8 Example 4: Front and rear views of the optimal trajectory (a, b), and yaw and pitch angles that realize it (c, d)

5 Conclusions

We have considered the problem of determining a UAV trajectory that minimizes the flight time in presence of avoidance areas and obstacles. Incorporating these constraints in the cost functional, with the aid of the Euler-Lagrange Theorem and the Pontryagin Minimum Principle, we have derived a boundary value problem with a Hamiltonian structure. The use of the code `bvptwp` combined with a suitable continuation strategy to manage the additional constraints has shown very efficient in comparison to the standard solvers available in MATLAB.

Acknowledgements The authors thank Cristian Brutto, Giuseppe Colucci and Roberto Lorusso for the support and useful discussions during the preparation of the paper. The last two authors are members of the INdAM Research group GNCS.

Funding Open access funding provided by Università degli Studi di Bari Aldo Moro within the CRUI-CARE Agreement. The research of Felice Iavernaro and Francesca Mazzia has been funded by the PON "Ricerca e Innovazione 2014-2020", project "RPASInAir: Integrazione dei Sistemi Aeromobili a Pilotaggio Remoto nello spazio aereo non segregato per servizi", n. ARS01_00820.

References

1. Aggarwal, S., Kumar, N.: Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges. Comput. Commun. **149**, 270–299 (2020)
2. Radmanesh, M., Kumar, M., Guentert, P.H., Sarim, M.: Overview of path-planning and obstacle avoidance algorithms for UAVs: A comparative study. Unmanned Syst. **6**(02), 95–118 (2018)
3. Yang, L., Qi, J., Xiao, J., Yong, X.: A literature review of UAV 3D path planning. Proceeding of the 11th World Congress on Intelligent Control and Automation, pp. 2376–2381. https://doi.org/10.1109/WCICA.2014.7053093 (2014)
4. Kang, M., Liu, Y., Ren, Y., Zhao, Y., Zheng, Z.: An empirical study on robustness of UAV path planning algorithms considering position uncertainty. 2017 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE). IEEE, pp. 1–6 (2017)
5. Samaniego, F., Sanchis, J., García-Nieto, S., Simarro, R.: UAV motion planning and obstacle avoidance based on adaptive 3D cell decomposition: Continuous space vs discrete space. 2017 IEEE Second Ecuador Technical Chapters Meeting (ETCM). IEEE, pp 1–6 (2017)
6. Siegwart, R., Nourbakhsh, I.R., Scaramuzza, D.: Autonomous mobile robots. A Bradford Book:15 (2011)
7. Tsourdos, A., White, B., Shanmugavel, M.: Cooperative path planning of unmanned aerial vehicles. vol. 32. Wiley (2010)
8. Jang, D.-S., Chae, H.-J., Choi, H.-L.: Optimal control-based UAV path planning with dynamically-constrained TSP with neighborhoods. 2017 17th International Conference on Control, Automation and Systems (ICCAS). IEEE, pp. 373–378 (2017)
9. Bai, W., Wu, X., Xie, Y., Wang, Y., Zhao, H., Chen, K., Li, Y., Hao, Y.: A cooperative route planning method for multi-UAVs based-on the fusion of artificial potential field and B-spline interpolation. 2018 37th Chinese Control Conference (CCC). IEEE, pp. 6733–6738 (2018)
10. Budiyanto, A., Cahyadi, A., Adji, T.B., Wahyunggoro, O.: UAV obstacle avoidance using potential field under dynamic environment. 2015 International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC). IEEE, pp. 187–192 (2015)
11. Dai, J., Wang, Y., Wang, C., Ying, J., Zhai, J.: Research on hierarchical potential field method of path planning for UAVs. 2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC). IEEE, pp. 529–535 (2018)
12. Yao, P., Wang, H., Su, Z.: UAV feasible path planning based on disturbed fluid and trajectory propagation. Chin. J. Aeronaut. **28**(4), 1163–1177 (2015)
13. Gerdts, M.: Optimal control of ODEs and DAEs. Walter de Gruyter (2011)
14. Von Stryk, O., Bulirsch, R.: Direct and indirect methods for trajectory optimization. Ann. Oper. Res. **37**(1), 357–373 (1992)
15. Kelly, M.: An introduction to trajectory optimization: How to do your own direct collocation. SIAM Rev. **59**(4), 849–904 (2017)
16. Longuski, J.M., Guzmán, J.J., Prussing, J.E.: Optimal control with aerospace applications. Springer (2014)
17. Miller, B., Stepanyan, K., Miller, A., Andreev, M.: 3D path planning in a threat environment. 2011 50th IEEE Conference on Decision and Control and European Control Conference. IEEE, pp 6864–6869 (2011)
18. Amodio, P., Brugnano, L., Iavernaro, F.: Energy-conserving methods for Hamiltonian boundary value problems and applications in astrodynamics. Adv. Comput. Math. **41**(4), 881–905 (2015)

19. Amodio, P., Iavernaro, F.: Symmetric boundary value methods for second order initial and boundary value problems. Mediterr. J. Math. **3**(3-4), 383–398 (2006)
20. Brugnano, L., Iavernaro, F.: Line integral methods for conservative problems, vol. 13. CRC Press (2016)
21. Capper, S., Cash, J., Mazzia, F.: On the development of effective algorithms for the numerical solution of singularly perturbed two-point boundary value problems. Int. J. Comput. Sci. Math. **1**(1), 42–57 (2007)
22. Mazzia, F., Sestini, A., Trigiante, D.: The continuous extension of the B-spline linear multistep methods for BVPs on non-uniform meshes. Appl. Numer. Math. **59**(3-4), 723–738 (2009)
23. Cash, J.R., Mazzia, F.: Efficient global methods for the numerical solution of nonlinear systems of two point boundary value problems. Recent. Adv. Comput. Appl. Math., 23–39. https://doi.org/10.1007/978-90-481-9981-5_2 (2011)
24. Manni, C., Mazzia, F., Sestini, A., Speleers, H.: BS2 methods for semi-linear second order boundary value problems. Appl. Math. Comput. **255**, 147–156 (2015). https://doi.org/10.1016/j.amc.2014.08.046
25. Cash, J.R., Mazzia, F.: Conditioning and hybrid mesh selection algorithms for two-point boundary value problems. Scalable Comput. **10**(4), 347–361 (2009)
26. Cash, J.R., Hollevoet, D., Mazzia, F., Nagy, A.M.: Algorithm 927: the MATHLAB code bvptwp.m for the numerical solution of two point boundary value problems. ACM Trans. Math. Softw. (TOMS) **39**(2), 1–12 (2013)
27. Soetaert, K., Cash, J., Mazzia, F.: Solving differential equations in R. Springer Science & Business Media (2012)
28. Mazzia, F., Cash, J.R.: A Fortran test set for boundary value problem solvers. AIP Conf. Proc. **1648**(1), 020009 (2015). https://doi.org/10.1063/1.4912313
29. Mazzia, F., Cash, J.R., Soetaert, K.: Solving boundary value problems in the open source software R: Package bvpsolve. Opuscula Math. **34**(2), 387–403 (2014)
30. Cash, J.R., Mazzia, F.: Hybrid mesh selection algorithms based on conditioning for two-point boundary value problems. J. Numer. Anal. Ind. Appl. Math. **1**(1), 81–90 (2006)
31. Martens, B., Gerdts, M.: Convergence analysis for approximations of optimal control problems subject to higher index differential-algebraic equations and mixed control-state constraints. SIAM J. Control Optim. **58**(1), 1–33 (2020). https://doi.org/10.1137/18M1219382
32. Biral, F., Bertolazzi, E., Bosetti, P.: Notes on numerical methods for solving optimal control problems. IEEJ J. Industry Appl. **5**(2), 154–166 (2016). https://doi.org/10.1541/ieejjia.5.154
33. Dal Bianco, N., Bertolazzi, E., Biral, F., Massaro, M.: Comparison of direct and indirect methods for minimum lap time optimal control problems. Veh. Syst. Dyn. **57**(5), 665–696 (2019). https://doi.org/10.1080/00423114.2018.1480048
34. MathWorks, T.: MATHLAB release 2020b. http://www.mathworks.com/
35. Cash, J.R., Mazzia, F.: A new mesh selection algorithm, based on conditioning, for two-point boundary value codes. J. Comput. Appl. Math. **184**(2), 362–381 (2005)
36. Mazzia, F., Cash, J.R., et al: Testset for BVP solvers. https://archimede.dm.uniba.it/~bvpsolvers/testsetbvpsolvers/