



Effective and Safe Trajectory Planning for an Autonomous UAV Using a Decomposition-Coordination Method

Imane Nizar¹ · Adil Jaafar¹ · Zineb Hidila¹ · Mohamed Barki¹ · El Hossein Illoussamen¹ · Mohammed Mestari¹

Received: 30 December 2020 / Accepted: 28 July 2021 / Published online: 27 October 2021
© The Author(s), under exclusive licence to Springer Nature B.V. 2021

Abstract

In this paper, we present a Decomposition Coordination (DC) method applied to solve the problem of safe trajectory planning for autonomous Unmanned Aerial Vehicle (UAV) in a dynamic environment. The purpose of this study is to make the UAV more reactive in the environment and ensure the safety and optimality of the computed trajectory. In this implementation, we begin by selecting a dynamic model of a fixed-arms quadrotor UAV. Then, we define our multi-objective optimization problem, which we convert afterward into a scalar optimization problem (SOP). The SOP is subdivided after that into smaller sub-problems, which will be treated in parallel and in a reasonable time. The DC principle employed in our method allows us to treat non-linearity at the local level. The coordination between the two levels is achieved after that through the Lagrange multipliers. Making use of the DC method, we can compute the optimal trajectory from the UAV's current position to a final target practically in real-time. In this approach, we suppose that the environment is totally supervised by a Ground Control Unit (GCU). To ensure the safety of the trajectory, we consider a wireless communication network over which the UAV may communicate with the GCU and get the necessary information about environmental changes, allowing for successful collision avoidance during the flight until the intended goal is safely attained. The analysis of the DC algorithm's stability and convergence, as well as the simulation results, are provided to demonstrate the advantages of our method and validate its potential.

Keywords Autonomous navigation · Optimal control · Trajectory planning · Unmanned aerial vehicles

1 Introduction

Autonomous unmanned aerial vehicles have raised interest during the last decades amongst the community of researchers [1]. Initially, different types of flying vehicles and robots were created for military applications [2], but they quickly became widely used in various civilian applications [3], given the effectiveness of these robots and their capability to improve the quality of man's life. For example, in the actual context of the global pandemic, drones are extensively used in the fight against Covid-19 in many countries so far, and for many purposes in [4] Euchi, J explains how drones are used to solve logistical problems of

medicines delivery for medical home care, during the time of public isolation.

Looking through recent papers, we find that there are still several open questions that need to be resolved, like the modeling and control of various autonomous UAVs [5], target tracking [6], and UAVs wireless communication networks as in [7]. In this work, we propose a solution to the problem of safe trajectory planning for autonomous quadrotor, which is one of the most challenging topics in this field [8, 9]. Several mathematical methods have been suggested to solve this problem. Most of them are based on Genetic Algorithm (GA) [10], or hybrid Neuro-genetic algorithm, which is an upgraded version of GA where the output of the GA is used to train the Artificial Neural Network (ANN) [11]. Nevertheless, in real-time implementations. These approaches are not the best options, especially for highly complex and/or large-size problems, given the difficulty of their convergence to the Pareto front. To overcome such difficulties, we propose a novel theoretical approach, that falls into the category of global navigation approaches, as described in [12], where the

✉ Imane Nizar
imane.nizar@gmail.com

UAV receives all the needed information from an on-land computer, called in this paper a Ground Control Unit (GCU) [13], which monitors the entire environment, while the UAV flies towards its destination. Using the Decomposition-Coordination (DC) algorithm [14], we compute a first optimal trajectory from the current position, p_i of the UAV to a desired position of our choice p_d .

To ensure safe navigation, the GCU supervises the whole environment, and finds every potential collision. Then, using an obstacle avoidance algorithm, it gives a safe position regarded as an intermediate goal for the UAV to escape the upcoming collision. The intermediate goals are meticulously chosen based on the orientation, speed, and size of the obstacle. The UAV navigates in accordance with the GCU instructions, and computes sub-trajectories from the current position to the next intermediate goal each time an obstacle is detected on the road until the destination is reached efficiently. This paper is an extended version of a previous conference paper [15]. It aims to describe, in a more thorough way, the DC algorithm and the global approach employed in this work.

We start with a description of the quadrotor's dynamic model utilized in this work. It is worth mentioning that the latter was designed based on a number of interesting recent papers [16] and [17].

Then, we define our objective functions that must be optimized in a conflicting situation. We proceed afterward with the resolution of the multi-objective optimization problem using a decomposition-coordination principle that consists of decomposing the problem into separable subproblems, easily processed in parallel, and in almost real-time, allowing the nonlinearity to be treated locally. The coordination is subsequently carried out using the Lagrange multipliers.

In this approach, the nonlinear equality constrained optimization problem is converted into a Scalar Optimization Problem (SOP) with a single objective function, and the SOP is solved through mapping the differential equations into the corresponding difference equations, simulated by discrete-time computing units.

The strength of this method lies in the decomposition and the parallel processing, to ensure the efficiency of solving different nonlinear optimization problems as in previously published works [12] and [14].

This paper is organized as follows: Section 2 is dedicated to the presentation of the quadrotor's dynamic model. In Section 3 we introduce the problem statement and its conversion to an equivalent SOP with a single objective function. Then, the analysis of the problem is given in Section 4, where the DC method is presented meticulously, along with the study of the stability of the algorithm. The principle of our approach for trajectory planning and obstacle avoidance is described in Section 5.

Then, Section 6 is devoted to the results of our simulation on Matlab. Finally, a conclusion is provided in Section 7.

2 The Quadrotor's Dynamic Model

The quadrotor is an under-actuated system that has six degrees of freedom. Three correspond to rotational movement around the x , y , and z axes, the other three correspond to translational movement along those axes. We consider a dynamic model of a fixed-arms quadrotor system, Fig. 1 represents the quadrotor's configuration used in this study: E and B are the earth and body frames, respectively.

As usual, the following assumptions are given to simplify the modeling:

- The quadrotor is a symmetric and rigid body,
- The quadrotor's center of mass coincides with the origin of the body frame,
- The rotation directions of the four rotors are fixed,
- The air resistance and the wind effect are neglected.

Let $[x, y, z]$ be the three axes of translations for the quadrotor with respect to the earth frame E , and $[\phi, \theta, \psi]$ the three angles of rotation around the those axes, where ϕ is the roll angle around the x -axis, θ is pitch angle around the y -axis and ψ the yaw angle around the z -axis. The matrices of rotation around the three axis from the earth frame to the body frame are:

$$R_\phi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & C_\phi & S_\phi \\ 0 & -S_\phi & C_\phi \end{bmatrix}, R_\theta = \begin{bmatrix} C_\theta & 0 & -S_\theta \\ 0 & 1 & 0 \\ S_\theta & 0 & C_\theta \end{bmatrix}, R_\psi = \begin{bmatrix} C_\psi & S_\psi & 0 \\ -S_\psi & C_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

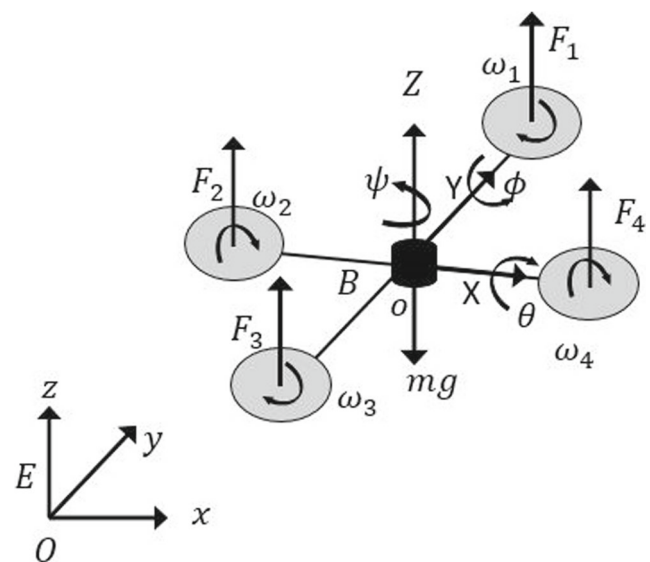


Fig. 1 The quadrotor's structure

Where $C_{(\cdot)}$ and $S_{(\cdot)}$ designate respectively $\cos(\cdot)$ and $\sin(\cdot)$. The transition matrix from the body frame to the earth frame is then represented as:

$$R_t = (R_\phi R_\theta R_\psi)^T = \begin{bmatrix} C_\psi C_\theta & C_\psi S_\theta S_\phi - S_\psi C_\phi & C_\psi S_\theta C_\phi + S_\psi S_\phi \\ S_\psi C_\theta & S_\psi S_\theta S_\phi + C_\psi C_\phi & S_\psi S_\theta C_\phi - C_\psi S_\phi \\ -S_\theta & C_\theta S_\phi & C_\theta C_\phi \end{bmatrix} \tag{1}$$

To describe the translation dynamics of the quadrotor, we use the total thrust force F_{th} , given by the combination of the thrust force of the four motors.

$$F_{th} = \sum_{i=1}^4 F_i = k \cdot \sum_{i=1}^4 \omega_i^2 \tag{2}$$

Where F_i and ω_i are respectively the lift force, and the angular velocity of the i^{th} rotor for $i = 1, \dots, 4$, and k is the lift constant.

The total lift force in the body frame is:

$$\vec{F}_{thB} = [0, 0, F_{th}]^T \tag{3}$$

According to Eqs. 1 and 3, the lift force in the earth frame is:

$$\vec{F}_{thE} = R_t \vec{F}_{thB} \tag{4}$$

Then, we have the gravity force of the quadrotor, described according to Newton’s Second Law in the earth frame as follows:

$$\vec{F}_{gE} = [0, 0, mg]^T \tag{5}$$

Where; m and g are respectively the mass of the quadrotor, and acceleration of gravity, and they are both supposed to be constants.

In compliance with the Newton’s second law, and from the equations above, the translation dynamics of the quadrotor, without considering external disturbances, can be described as:

$$\begin{aligned} \ddot{x} &= \frac{F_{th}(S_\theta C_\phi C_\psi + S_\phi S_\psi)}{m} \\ \ddot{y} &= \frac{F_{th}(S_\theta C_\phi S_\psi + S_\phi C_\psi)}{m} \\ \ddot{z} &= \frac{F_{th}(C_\theta C_\phi - mg)}{m} \end{aligned} \tag{6}$$

For the rotational dynamics of the quadrotor, let $[\omega_x, \omega_y, \omega_z]^T$ be the angular velocity of the quadrotor, about the principal axes in the body-fixed frame B .

According to Euler’s equations for rotations, the resultant torque acting on the quadrotor is described as:

$$\begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \begin{bmatrix} I_x \dot{\omega}_x + \omega_y \omega_z (I_z - I_y) \\ I_y \dot{\omega}_y + \omega_x \omega_z (I_x - I_z) \\ I_z \dot{\omega}_z + \omega_x \omega_y (I_y - I_x) \end{bmatrix} \tag{7}$$

Where $\tau_x, \tau_y,$ and τ_z the three components of the torque in the three directions; $x, y,$ and z . While, $I_x, I_y,$ and I_z are the three rotary inertias, with respect to the axes $x, y,$ and z respectively.

The torques in the three directions $x, y,$ and z are respectively $\tau_\phi, \tau_\theta,$ and τ_ψ , given by:

$$\begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} l(F_3 - F_1) \\ l(F_4 - F_2) \\ D_4 + D_2 - D_3 - D_1 \end{bmatrix} \tag{8}$$

Where $D_i = d\omega_i^2$ is the drag force of the i^{th} rotor, and constant d is the drag coefficient, and l is the quadrotor’s arm length.

Then, $M_{gx}, M_{gy},$ and M_{gz} are the three components of the gyroscopic torque during flight around the axes $x, y,$ and $z,$ respectively, denoted by:

$$\begin{bmatrix} M_{gx} \\ M_{gy} \\ M_{gz} \end{bmatrix} = I_r \begin{bmatrix} \omega_x \varpi \\ \omega_y \varpi \\ 0 \end{bmatrix} \tag{9}$$

Where $\varpi = (\omega_2 + \omega_3 - \omega_1 - \omega_4)$ while I_r represents the rotor inertia.

Using Eqs. 7, 8, and 9, we have the following angular-motion dynamics equations:

$$I_x \dot{\omega}_x = (I_y - I_z)\omega_y \omega_z - I_r \omega_y \varpi + lk(\omega_4^2 - \omega_2^2) \tag{10}$$

$$I_y \dot{\omega}_y = (I_z - I_x)\omega_x \omega_z + I_r \omega_x \varpi + lk(\omega_3^2 - \omega_1^2) \tag{11}$$

$$I_z \dot{\omega}_z = (I_x - I_y)\omega_x \omega_y + d(\omega_4^2 + \omega_2^2 - \omega_3^2 - \omega_1^2) \tag{12}$$

The angular velocity of the quadrotor $[\omega_x, \omega_y, \omega_z]^T$, in the coordinate system B , can be expressed in the coordinate system E as:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & S_\phi t_\theta & C_\phi t_\theta \\ 0 & C_\phi & -S_\phi \\ 0 & \frac{S_\phi}{C_\theta} & \frac{C_\phi}{C_\theta} \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{13}$$

With $t_{(\cdot)} = \tan(\cdot)$.

To simplify the equations, we assume that $[\dot{\phi}, \dot{\theta}, \dot{\psi}]^T = [\omega_x, \omega_y, \omega_z]^T$. This assumption holds true for small angles of movement. Thus, the rotational dynamic model in the inertial frame becomes:

$$\begin{aligned} \ddot{\phi} &= \frac{(I_y - I_z)\dot{\theta}\dot{\psi} - I_r \dot{\theta} \varpi + lk(\omega_4^2 - \omega_2^2)}{I_x} \\ \ddot{\theta} &= \frac{(I_z - I_x)\dot{\phi}\dot{\psi} + I_r \dot{\phi} \varpi + lk(\omega_3^2 - \omega_1^2)}{I_y} \\ \ddot{\psi} &= \frac{(I_x - I_y)\dot{\phi}\dot{\theta} + d(\omega_4^2 - \omega_3^2 + \omega_2^2 - \omega_1^2)}{I_z} \end{aligned} \tag{14}$$

From Eqs. 6 and 14, we obtain the following quadrotor dynamic model:

$$\dot{p} = M_1(p).p + M_2(p).u + C \tag{15}$$

For which the control is represented by the variable: $u = [u_1, u_2, u_3, u_4]^T$.

Where:

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} k(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2) \\ lk(\omega_4^2 - \omega_2^2) \\ lk(\omega_3^2 - \omega_1^2) \\ d(\omega_4^2 + \omega_2^2 - \omega_3^2 - \omega_1^2) \end{bmatrix} \tag{16}$$

While the state variable is expressed by:

$$p = [x, y, z, \dot{x}, \dot{y}, \dot{z}, \phi, \theta, \psi, \dot{\phi}, \dot{\theta}, \dot{\psi}]^T$$

We define state matrix $M_1(p) \in \mathbb{R}^{12 \times 12}$, the control matrix $M_2(p) \in \mathbb{R}^{12 \times 4}$, and $C \in \mathbb{R}^{12}$ the constant matrix below:

$$M_1(p) = \begin{bmatrix} O_{3,3} & I_{3,3} & O_{3,3} & O_{3,3} \\ O_{3,3} & O_{3,3} & O_{3,3} & O_{3,3} \\ O_{3,3} & O_{3,3} & O_{3,3} & I_{3,3} \\ O_{3,3} & O_{3,3} & O_{3,3} & A_1 \end{bmatrix}, M_2(p) = \begin{bmatrix} O_{3,1} & O_{3,3} \\ A_2 & O_{3,3} \\ O_{3,1} & O_{3,3} \\ O_{3,1} & A_3 \end{bmatrix}$$

$$C = [0, 0, 0, 0, 0, -g, 0, 0, 0, 0, 0, 0]^T$$

With $O_{3,3} \in \mathbb{R}^{3 \times 3}$ and $O_{3,1} \in \mathbb{R}^{3 \times 1}$ are null matrices, and $I_{3,3} \in \mathbb{R}^{3 \times 3}$ the identity matrix.

While A_1, A_2 and A_3 are given by:

$$A_1 = \begin{bmatrix} \frac{\dot{\theta}(I_y - I_z)}{I_x} & \frac{-I_x \omega}{I_x} & 0 \\ \frac{I_x \omega}{I_y} & 0 & \frac{\dot{\phi}(I_z - I_x)}{I_y} \\ 0 & \frac{\dot{\phi}(I_x - I_y)}{I_z} & 0 \end{bmatrix}, A_2 = \frac{1}{m} \begin{bmatrix} S_\theta C_\phi C_\psi + S_\phi S_\psi \\ S_\theta C_\phi S_\psi + S_\phi C_\psi \\ C_\theta C_\phi \end{bmatrix}, A_3 = \begin{bmatrix} \frac{1}{I_x} & 0 & 0 \\ 0 & \frac{1}{I_y} & 0 \\ 0 & 0 & \frac{1}{I_z} \end{bmatrix}$$

3 The Statement of the Problem

The aim of this study is to optimize and plan the trajectory of a quadrotor UAV, knowing the initial position p_i , and the arrival position p_d . Using the DC method, we can solve the nonlinear problem of the optimal trajectory. Then, with the assistance of the GCU, which supervises the indoor environment, and provides the necessary instructions at the right time. We can enable the UAV to be more responsive in a dynamic environment and enable it to reach its final destination safely.

In this global approach, we first start by solving the problem of optimal navigation for the quadrotor, using

the DC algorithm. To apply this algorithm properly, we have to change the dynamic model from continuous-time to discrete-time using N steps forward Euler method:

$$p_{k+1} = dt. \left[\left(\frac{1}{dt}.I_{12,12} + M_1(p_k) \right).p_k + M_2(p_k).u_k + C \right] \tag{17}$$

Where $I_{12,12} \in \mathbb{R}^{12 \times 12}$ is the identity matrix.

To simplify the equation, we put $dt = 1$ s. We obtain:

$$p_{k+1} = [I_{12,12} + M_1(p_k)].p_k + M_2(p_k).u_k + C \tag{18}$$

To proceed with the resolution of this problem, we consider the following non-linear discrete-time systems:

$$\begin{cases} p_{k+1} = f(p_k, u_k) & k \in [0, N - 1] \\ p_0 = p_i \text{ given} \\ p_d = p_N \text{ given} \end{cases} \tag{19}$$

Where p_k is the state, and u_k the control at the instant t_k , and t_N is the moment when the UAV reaches its goal $p_d = p_N$.

For any nonlinear problem, we have several objective functions $J_1(y^*) \geq J_1(y)$, $J_2(y^*) \geq J_2(y), \dots, J_n(y^*) \geq J_n(y)$, that have to be satisfied in a conflicting situation.

Of two objective functions, the one with the smaller maximum is then preferable, and the optimum procedures are those with the minimax property of minimizing the maximum loss. This method is a decision rule, used here to compute the smallest value of the maximum values of the objective functions J_i .

We put w_i the weight of the i objective function, where $\sum_{i=0}^n w_i = 1$. (To simplify, the functions to be maximized are converted as follows: $\max J_i(p, u) = -\min(-J_i(p, u))$ so that we assume that all our objective functions are to be minimized).

For our navigation problem, We aim to calculate the optimal control to make the quadrotor reaches its destination, i.e., the sequence of control inputs $u_k^*(k = 0, 1, \dots, N - 1)$ that enabling a desired state p_d to be reached at the time t_N , which minimizes the cost function of control efforts, for this end, we consider the objective function below:

$$J_1(p, u) = \frac{1}{2} \sum_{k=0}^{N-1} \|u_k\|^2 \tag{20}$$

Where u_k is the control input computed at the instant t_k .

The second objective is to reach the desired stated optimally which could be represented by the objective function J_2 :

$$J_2(p, u) = \frac{1}{2} \|p_N - p_d\|^2 \tag{21}$$

Where p_N represent the computed state at the instant t_N ; i.e., the final iteration of the algorithm, and p_d is the desired state.

The scalar optimization problem can be described by:

$$E(p, u) = \max_{1 \leq i \leq n} \{w_i J_i(p, u)\} \tag{22}$$

While, the system bellow describes the multi-objective optimization problem:

$$\begin{cases} \min_{(u_i^*/0 \leq i \leq N-1)} E(p, u) \\ p_{k+1} = f(p_k, u_k) \\ p_0, \text{ given} \\ k \in [0, N - 1] \end{cases} \tag{23}$$

4 The Analysis of Problem

This section is devoted to the mathematical approach applied to solve the problem Eq. 23, defined in the previous section. To proceed with the decomposition of the system Eq. 19, into N static subsystems, allowing us to switch from a nonlinear dynamic system to a set of N static interconnected subsystems, as shown in Fig. 2, where the new variable z_k represents the output of the subsystems k .

$$z_k = f(p_k, u_k) \quad k \in [0, N - 1] \tag{24}$$

$$p_k = z_{k-1} \quad k \in [1, N] \tag{25}$$

The purpose now is to minimize the objective functions Eqs. 20 and 21, under the constraints Eqs. 24 and 25:

$$\begin{cases} \min_{(u_k^*/0 \leq k \leq N-1)} E(p, u) \\ z_k = f(p_k, u_k) & k \in [0, N - 1] \\ p_k = z_{k-1} & k \in [1, N] \\ p_0 = p_i, \text{ given} \\ p_d = p_N, \text{ given} \end{cases} \tag{26}$$

To solve the problem of derivation for the objective function, we construct the ordinary Lagrange function.

$$L = \sum_{k=0}^{N-1} L_k \tag{27}$$

$$L_0 = \frac{1}{N} E(p_0, u_0) + \mu_0^T (f(p_0, u_0) - z_0) \tag{28}$$

$$L_k = \frac{1}{N} E(p_k, u_k) + \mu_k^T (f(p_k, u_k) - z_k) + \beta_k^T (p_k - z_{k-1}) \tag{29}$$

$$for k \in [1, N - 2]$$

$$L_{N-1} = \frac{1}{N} E(p_{N-1}, u_{N-1}) + \mu_{N-1}^T (f(p_{N-1}, u_{N-1}) - p_d) + \beta_{N-1}^T (p_{N-1} - z_{N-2}) \tag{30}$$

The Lagrange multipliers β_k and μ_k are used here to take into consideration the constraints Eqs. 24 and 25. By deriving Ordinary Lagrange function and in accordance with the Karush-Khun-Tucker conditions [14]. We can change the equality constrained optimization problem Eq. 26 into differential equations, considering an equilibrium point $(p_k^*, u_k^*, \mu_k^*, \beta_k^*, z_k^*)$, which satisfies the equations below:

$$\nabla_{p_k} L = \frac{1}{N} \frac{\partial E}{\partial p_k} (p_k^*, u_k^*) + \mu_k^{*T} \frac{\partial f}{\partial p_k} (p_k^*, u_k^*) + \beta_k^{*T} = 0 \tag{31}$$

$$for k \in [1, N - 1]$$

$$\nabla_{u_k} L = \frac{1}{N} \frac{\partial E}{\partial u_k} (p_k^*, u_k^*) + \mu_k^{*T} \frac{\partial f}{\partial p_k} (p_k^*, u_k^*) = 0 \tag{32}$$

$$for k \in [0, N - 1]$$

$$\nabla_{\mu_k} L = f(p_k^*, u_k^*) - z_k^* = 0 \quad for k \in [0, N - 1] \tag{33}$$

$$\nabla_{z_k} L = -\mu_k^{*T} - \beta_{k+1}^{*T} = 0 \quad for k \in [0, N - 2] \tag{34}$$

$$\nabla_{\beta_k} L = p_k^* - z_{k-1}^* = 0 \quad for k \in [1, N - 1] \tag{35}$$

The resolution of the above system Eqs. 31 - 35, is actually equivalent to resolution of our optimization problem Eq. 26.

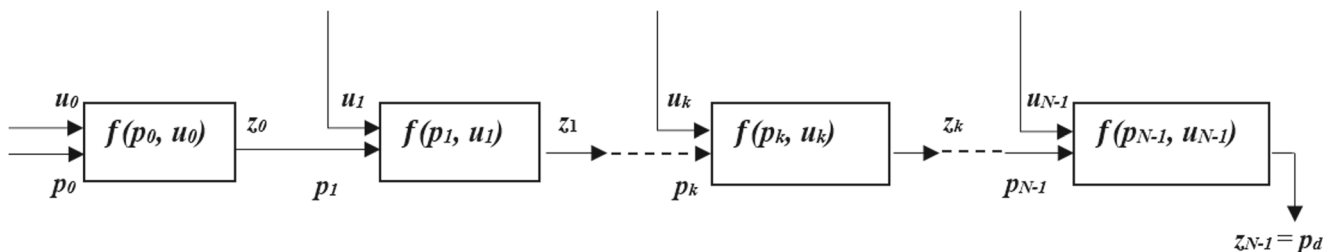


Fig. 2 Overall system made up of N static interconnected sub-system in a serial structure

4.1 The Method of Decomposition-coordination

The principle of the DC method has been neatly presented in previous papers as in [14]. And in this section, we reintroduce this principle, which relies on the subdivision of the system of differential equations Eqs. 31 -35 into two levels. This decomposition is made to gather at the lower level all the sub-problems k that would only involve the unknown variables of indices k ($k \in [1, N - 1]$).

In fact, we decompose the treatment of the system Eqs. 31 - 35 between two levels. An upper level, which treats the Eqs. 34 - 35, responsible for fixing the initial values z_k for $k \in [0, N - 2]$, and β_k for $k \in [1, N - 1]$, then proposing them to the lower level that solves the system Eqs. 31, 32, and 33, so that the decomposition of the problem Eq. 26 is achieved.

The following systems represent our N interconnected sub-problems: *Sub-problem 0*:

$$\begin{cases} p_0 \text{ given by the upper level} \\ \min E(p, u) \\ \text{Subject to } f(p_0, u_0) = z_0 \end{cases}$$

Sub-problem k , for $k \in [1, N - 2]$:

$$\begin{cases} z_k \text{ and } \beta_k \text{ given by the upper level} \\ \min E(p, u) + \beta_k(p_k - z_{k-1}) \\ \text{Subject to } f(p_k, u_k) = z_k \end{cases}$$

Sub-problem $N - 1$:

$$\begin{cases} z_{N-2} \text{ and } \beta_{N-1} \text{ given by the upper level} \\ \min E(p, u) + \beta_{N-1}^T(p_{N-1} - z_{N-2}) \\ \text{Subject to } f(p_{N-1}, u_{N-1}) - p_d = 0. \end{cases}$$

The resolution of each sub-system k corresponds now to the treatment of the system of Eqs. 31, 32, and 33, for each β_k ($k = 1 \dots N - 1$) and z_k ($k = 0 \dots N - 2$) received from the upper level. And using the gradient method, we end up with the differential equations below:

$$\frac{dp_k}{dt} = -\lambda_p \nabla_{p_k} L \tag{36}$$

$$\frac{du_k}{dt} = -\lambda_u \nabla_{u_k} L \tag{37}$$

$$\frac{d\mu_k}{dt} = \lambda_\mu \nabla_{\mu_k} L \tag{38}$$

With $\lambda_p > 0$, $\lambda_u > 0$, and $\lambda_\mu > 0$.

By applying the Forward Euler method, we convert the differential Eqs. 36, 37, and 38 into the following equations in discrete-time. We consider the step of discretization $dt = 1$:

$$p_k^{(i+1)} = p_k^{(i)} - \lambda_p \nabla_{p_k} L, \quad k \in [1, N - 1] \tag{39}$$

$$u_k^{(i+1)} = u_k^{(i)} - \lambda_u \nabla_{u_k} L, \quad k \in [1, N - 1] \tag{40}$$

$$\mu_k^{(i+1)} = \mu_k^{(i)} + \lambda_\mu \nabla_{\mu_k} L, \quad k \in [0, N - 1] \tag{41}$$

To ensure the coordination, the upper level treats in parallel the coordination parameters β_k^i , and z_k^i , which are known within the lower level and used to enable a local resolution of the system of difference equations Eqs. 39 - 41 to find the values of $p_k^*(z_k^i, \beta_k^i)$, $u_k^*(z_k^i, \beta_k^i)$, and $\mu_k^*(z_k^i, \beta_k^i)$ that satisfy the Eqs. 39 - 41. The computed values $p_k^*(z_k^i, \beta_k^i)$ and $\mu_k^*(z_k^i, \beta_k^i)$ are then forwarded to the upper level to solve the system and check if the earliest values of z_k^i and β_k^i were correct, or correct them if necessary.

z_k^i and β_k^i are given by:

$$z_k^{(i+1)} = z_k^{(i)} - \lambda_z \nabla_{z_k} L, \quad k \in [0, N - 2] \tag{42}$$

$$\beta_k^{(i+1)} = \beta_k^{(i)} + \lambda_\beta \nabla_{\beta_k} L, \quad k \in [1, N - 1] \tag{43}$$

With $\lambda_z > 0$, $\lambda_\beta > 0$.

The treatment is repeated in the coordination loop of the algorithm until the coordination between the two levels is obtained, as shown in Fig. 3.

4.2 The Stability Analysis

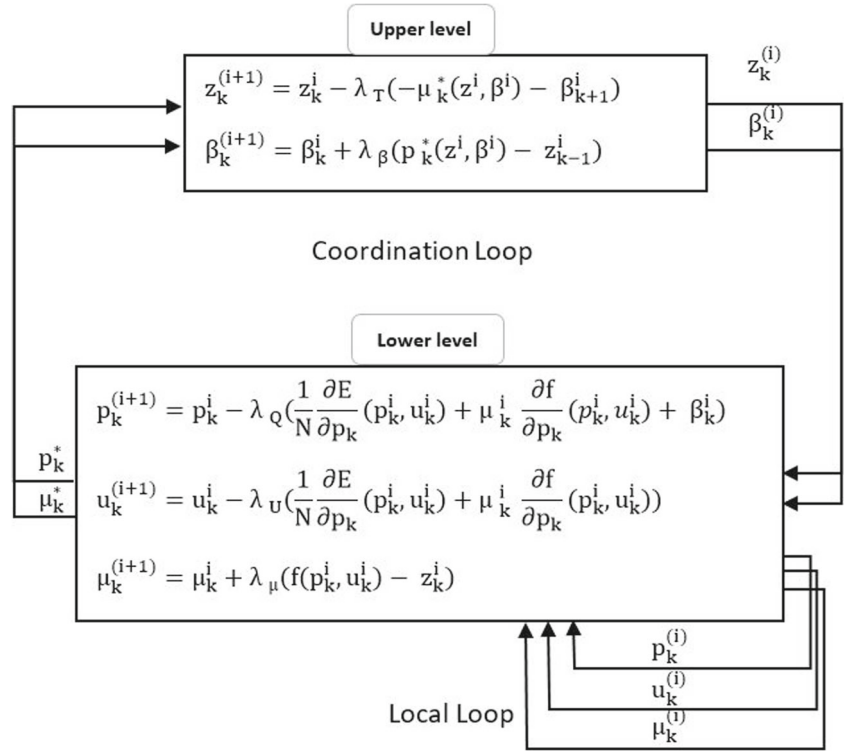
In this part, we discuss the stability and consistency of our DC algorithm. To this end, we will demonstrate that its convergence can be reduced to that of the level of coordination. To simplify the notations, we use the new variable $V_k = (p_k, u_k)^T$, where $V_k^*(z_k^*, \beta_k^*)$, $\mu_k^*(z_k^*, \beta_k^*)$, z_k^* , and β_k^* are the sought solution. The variables $V_k^*(z_k^i, \beta_k^i)$, and $\mu_k^*(z_k^i, \beta_k^i)$ are to be found by the lower level Eqs. 39, 40 and 41, and satisfy locally Eqs. 31, 32, and 33, while the values of z_k^i , and β_k^i (the coordination variables), are to be adjusted in the upper level Eqs. 42 and 43.

Our problem can be presented in a condensed form as follows:

$$\text{Lower level equations} \begin{cases} X_k(V_k, \mu_k, \beta_k) = \begin{pmatrix} \nabla_{p_k} L \\ \nabla_{u_k} L \end{pmatrix} \\ P_k(V_k, z_k) = \nabla_{\mu_k} L \end{cases}$$

$$\text{Upper level equations} \begin{cases} R_k(V_k, z_{k-1}) = \nabla_{\beta_k} L \\ S_k(\mu_k, \beta_{k+1}) = \nabla_{z_k} L \end{cases}$$

Fig. 3 Coordination between the upper and the lower level



We can write the solution sought as:

$$\begin{cases} X_k^* = \left(\frac{1}{N} \frac{\partial E}{\partial p_k} + \beta_k^* + \mu_k^* \frac{\partial f}{\partial p_k} \right) = 0 \\ P_k^* = f(p_k^*, u_k^*) - z_k^* = 0 \\ R_k^* = p_k^* - z_{k-1}^* = 0 \\ S_k^* = -\mu_k^* - \beta_{k+1}^* = 0 \end{cases} \quad (44)$$

For each iteration i at the coordination loop:

$$\begin{cases} X_k^i = 0 \\ P_k^i = 0 \end{cases} \quad (45)$$

And we define the errors at the i^{th} iteration as follows:

$$\begin{cases} e_{V_k}^{(i)} = V_k^*(z_k^{(i)}, \beta_k^{(i)}) - V_k^*(z_k^*, \beta_k^*) \\ e_{\mu_k}^{(i)} = \mu_k^*(z_k^{(i)}, \beta_k^{(i)}) - \mu_k^*(z_k^*, \beta_k^*) \\ e_{T_k}^{(i)} = z_k^{(i)} - z_k^* \\ e_{\beta_k}^{(i)} = \beta_k^{(i)} - \beta_k^* \end{cases}$$

Theorem 1 Let $e_{V_k}^{(i)}$, $e_{\mu_k}^{(i)}$, $e_{z_k}^{(i)}$ and $e_{\beta_k}^{(i)}$ be the errors at the i^{th} iteration of the coordination loop. $e_{V_k}^{(i)}$ and $e_{\mu_k}^{(i)}$ converge to zero, if $e_{z_k}^{(i)}$ and $e_{\beta_k}^{(i)}$ converge to zero.

Proof The linearization of the lower level's equations in the neighborhood of the solution allows us to obtain the following equations:

$$X_k^{(i)} \simeq X_k^* + \frac{\partial X_k^*}{\partial V_k} e_{V_k}^{(i)} + \frac{\partial X_k^*}{\partial \mu_k} e_{\mu_k}^{(i)} + \frac{\partial X_k^*}{\partial \beta_k} e_{\beta_k}^{(i)} \quad (46)$$

$$P_k^{(i)} \simeq P_k^* + \frac{\partial P_k^*}{\partial V_k} e_{V_k}^{(i)} + \frac{\partial P_k^*}{\partial z_k} e_{z_k}^{(i)} \quad (47)$$

Then, considering Eqs. 44 and 45 we can write:

$$\frac{\partial X_k^*}{\partial V_k} e_{V_k}^{(i)} + \frac{\partial X_k^*}{\partial \mu_k} e_{\mu_k}^{(i)} + \frac{\partial X_k^*}{\partial \beta_k} e_{\beta_k}^{(i)} = 0 \quad (48)$$

$$\frac{\partial P_k^*}{\partial V_k} e_{V_k}^{(i)} + \frac{\partial P_k^*}{\partial z_k} e_{z_k}^{(i)} = 0 \quad (49)$$

Which proves that: when $e_{z_k}^{(i)} \rightarrow 0$ and $e_{\beta_k}^{(i)} \rightarrow 0$, we have $e_{V_k}^{(i)} \rightarrow 0$ and $e_{\mu_k}^{(i)} \rightarrow 0$.

By studying the variation of the errors $e_{z_k}^{(i)}$ and $e_{\beta_k}^{(i)}$ at the coordination loop, over the iterations, we can prove the convergence of the algorithm in Fig. 3.

For this purpose, we use the Lyapunov function below to study the convergence:

$$\Phi(i) = \frac{1}{2} \sum_{k=0}^{N-1} \left(e_{\beta_k}^{(i)T} e_{\beta_k}^{(i)} + e_{z_k}^{(i)T} e_{z_k}^{(i)} \right) \geq 0 \quad (50)$$

The change of this Lyapunov function is :

$$\Delta\Phi(i) = \Phi(i+1) - \Phi(i) \quad (51)$$

$$\Delta\Phi(i) = \frac{1}{2} \sum_{k=0}^{N-1} \left(e_{\beta_k}^{(i+1)T} e_{\beta_k}^{(i+1)} - e_{\beta_k}^{(i)T} e_{\beta_k}^{(i)} + e_{z_k}^{(i+1)T} e_{z_k}^{(i+1)} - e_{z_k}^{(i)T} e_{z_k}^{(i)} \right) \tag{52}$$

Then, we develop Eq. 52 to have:

$$\Delta\Phi(i) = \sum_{k=0}^{N-1} \left(e_{z_k}^{(i)T} \Delta e_{z_k}^{(i)} + e_{\beta_k}^{(i)T} \Delta e_{\beta_k}^{(i)} + \frac{1}{2} \left(\Delta e_{z_k}^{(i)T} \Delta e_{z_k}^{(i)} + \Delta e_{\beta_k}^{(i)T} \Delta e_{\beta_k}^{(i)} \right) \right) \tag{53}$$

Where

$$\Delta e_{z_k}^{(i)} = e_{z_k}^{(i+1)} - e_{z_k}^{(i)} = -\lambda S_k^{(i)} \tag{54}$$

$$\Delta e_{\beta_k}^{(i)} = e_{\beta_k}^{(i+1)} - e_{\beta_k}^{(i)} = \lambda R_k^{(i)} \tag{55}$$

And $\lambda = \lambda_z = \lambda_\beta$.

Then, we linearize the equations of the upper level in the neighborhood of the solution to obtain the following equations:

$$S_k^{(i)} \simeq S_k^* + \frac{\partial S_k^*}{\partial \mu_k} e_{\mu_k}^{(i)} + \frac{\partial S_k^*}{\partial \beta_{k+1}} e_{\beta_{k+1}}^{(i)} \tag{56}$$

$$R_k^{(i)} \simeq R_k^* + \frac{\partial R_k^*}{\partial V_k} e_{V_k}^{(i)} + \frac{\partial R_k^*}{\partial z_{k-1}} e_{z_{k-1}}^{(i)} \tag{57}$$

From Eqs, 44, 56, and 57 we have:

$$\Delta e_{z_k}^{(i)} = -\lambda \left(\frac{\partial S_k^*}{\partial \mu_k} e_{\mu_k}^{(i)} + \frac{\partial S_k^*}{\partial \beta_{k+1}} e_{\beta_{k+1}}^{(i)} \right) \tag{58}$$

$$\Delta e_{\beta_k}^{(i)} = \lambda \left(\frac{\partial R_k^*}{\partial V_k} e_{V_k}^{(i)} + \frac{\partial R_k^*}{\partial z_{k-1}} e_{z_{k-1}}^{(i)} \right) \tag{59}$$

And using Eqs. 58 and 59 we can write the change of the function Φ as:

$$\Delta\Phi = \sum_{k=0}^{N-1} \lambda \left[\left(-e_{z_k}^{(i)T} \frac{\partial S_k^*}{\partial \mu_k} e_{\mu_k}^{(i)} - e_{z_k}^{(i)T} \frac{\partial S_k^*}{\partial \beta_{k+1}} e_{\beta_{k+1}}^{(i)} + e_{\beta_k}^{(i)T} \frac{\partial R_k^*}{\partial V_k} e_{V_k}^{(i)} + e_{\beta_k}^{(i)T} \frac{\partial R_k^*}{\partial z_{k-1}} e_{z_{k-1}}^{(i)} \right) + \frac{1}{2} \lambda^2 \left(S_k^{(i)T} S_k^{(i)} + R_k^{(i)T} R_k^{(i)} \right) \right] = A(i)\lambda^2 + B(i)\lambda \tag{60}$$

With

$$A(i) = \frac{1}{2} \sum_{k=0}^{N-1} S_k^{(i)T} S_k^{(i)} + R_k^{(i)T} R_k^{(i)} \geq 0 \tag{61}$$

And

$$B(i) = \sum_{k=0}^{N-1} \left(-e_{z_k}^{(i)T} \frac{\partial S_k^*}{\partial \mu_k} e_{\mu_k}^{(i)} - e_{z_k}^{(i)T} \frac{\partial S_k^*}{\partial \beta_{k+1}} e_{\beta_{k+1}}^{(i)} + e_{\beta_k}^{(i)T} \frac{\partial R_k^*}{\partial V_k} e_{V_k}^{(i)} + e_{\beta_k}^{(i)T} \frac{\partial R_k^*}{\partial z_{k-1}} e_{z_{k-1}}^{(i)} \right) \tag{62}$$

□

Theorem 2 Let as consider λ the adaptive coefficient for the coordination algorithm. The convergence is guaranteed if one matrix of $(\partial X_k^*/\partial V_k)^T$ for $(k = 0, 1, \dots, N - 1)$ is positive definite, and the others are only positive semi-definite and if $A(i) \neq 0$, with λ chosen as: $0 < \lambda < |B(i)/A(i)|$.

Proof Knowing that $(\partial X_k^*/\partial \beta_k)^T = (\partial R_k^*/\partial V_k)$, we can write:

$$B(i) = \sum_{k=0}^{N-1} \left(-e_{z_k}^{(i)T} \frac{\partial S_k^*}{\partial \mu_k} e_{\mu_k}^{(i)} - e_{z_k}^{(i)T} \frac{\partial S_k^*}{\partial \beta_{k+1}} e_{\beta_{k+1}}^{(i)} + e_{\beta_k}^{(i)T} \frac{\partial X_k^{*T}}{\partial \beta_k} e_{V_k}^{(i)} + e_{\beta_k}^{(i)T} \frac{\partial R_k^*}{\partial z_{k-1}} e_{z_{k-1}}^{(i)} \right) \tag{63}$$

We also have $(\partial X_k^*/\partial \mu_k)^T = (\partial P_k^*/\partial V_k)$, and using Eq. 46 we have:

$$e_{\beta_k}^{(i)T} \frac{\partial X_k^{*T}}{\partial \beta_k} = -e_{V_k}^{(i)T} \frac{\partial X_k^{*T}}{\partial V_k} - e_{\mu_k}^{(i)T} \frac{\partial P_k^*}{\partial V_k} \tag{64}$$

Using Eqs. 63 and 64 we can write:

$$B(i) = \sum_{k=0}^{N-1} \left(-e_{z_k}^{(i)T} \frac{\partial S_k^*}{\partial \mu_k} e_{\mu_k}^{(i)} - e_{z_k}^{(i)T} \frac{\partial S_k^*}{\partial \beta_{k+1}} e_{\beta_{k+1}}^{(i)} - e_{V_k}^{(i)T} \frac{\partial X_k^{*T}}{\partial V_k} e_{V_k}^{(i)} - e_{\mu_k}^{(i)T} \frac{\partial P_k^*}{\partial V_k} e_{V_k}^{(i)} + e_{\beta_k}^{(i)T} \frac{\partial R_k^*}{\partial z_{k-1}} e_{z_{k-1}}^{(i)} \right) = \sum_{k=0}^{N-1} \left(-e_{\mu_k}^{(i)T} \left(\frac{\partial S_k^{*T}}{\partial \mu_k} e_{z_k}^{(i)} + \frac{\partial P_k^*}{\partial V_k} e_{V_k}^{(i)} \right) - e_{z_k}^{(i)T} \frac{\partial S_k^*}{\partial \beta_{k+1}} e_{\beta_{k+1}}^{(i)} + e_{\beta_k}^{(i)T} \frac{\partial R_k^*}{\partial z_{k-1}} e_{z_{k-1}}^{(i)} - e_{V_k}^{(i)T} \frac{\partial X_k^{*T}}{\partial V_k} e_{V_k}^{(i)} \right) \tag{65}$$

We remark that $(\partial S_k^*/\partial \mu_k)^T = (\partial P_k^*/\partial z_k) = -I$, and according to Eq. 49 we have:

$$\frac{\partial S_k^{*T}}{\partial \mu_k} e_{z_k}^{(i)} + \frac{\partial P_k^*}{\partial V_k} e_{V_k}^{(i)} = 0 \tag{66}$$

Then we have:

$$B(i) = \sum_{k=0}^{N-1} \left(-e_{z_k}^{(i)T} \frac{\partial S_k^*}{\partial \beta_{k+1}} e_{\beta_{k+1}}^{(i)} + e_{\beta_k}^{(i)T} \frac{\partial R_k^*}{\partial z_{k-1}} e_{z_{k-1}}^{(i)} - e_{V_k}^{(i)T} \frac{\partial X_k^{*T}}{\partial V_k} e_{V_k}^{(i)} \right) \tag{67}$$

Making use of the fact that

$$(\partial S_k^*/\partial \beta_{k+1}) = (\partial R_k^*/\partial z_{k-1}) = -I, \text{ we obtain:}$$

$$B(i) = \sum_{k=0}^{N-1} \left(e_{z_k}^{(i)T} e_{\beta_{k+1}}^{(i)} - e_{\beta_k}^{(i)T} e_{z_{k-1}}^{(i)} - e_{V_k}^{(i)T} \frac{\partial X_k^{*T}}{\partial V_k} e_{V_k}^{(i)} \right) \tag{68}$$

β_k and z_k are not defined respectively for $k = N$ and $k = 1$, thus $e_{\beta_N}^{(i)} = e_{z_{-1}}^{(i)} = 0$.

$$\begin{aligned}
 B(i) &= - \sum_{k=0}^{N-1} e_{V_k}^{(i)T} \frac{\partial X_k^*}{\partial V_k} e_{V_k}^{(i)} + \sum_{k=0}^{N-2} e_{z_k}^{(i)T} e_{\beta_{k+1}}^{(i)} - \sum_{k=1}^{N-1} e_{\beta_k}^{(i)T} e_{z_{k-1}}^{(i)} \\
 &= - \sum_{k=0}^{N-1} e_{V_k}^{(i)T} \frac{\partial X_k^*}{\partial V_k} e_{V_k}^{(i)} + \sum_{k=1}^{N-1} e_{z_{k-1}}^{(i)T} e_{\beta_k}^{(i)} - \sum_{k=1}^{N-1} e_{\beta_k}^{(i)T} e_{z_{k-1}}^{(i)}
 \end{aligned}
 \tag{69}$$

Also we have $e_{z_{k-1}}^{(i)T} e_{\beta_k}^{(i)} = e_{\beta_k}^{(i)T} e_{z_{k-1}}^{(i)}$, which implies :

$$B(i) = - \sum_{k=0}^{N-1} e_{V_k}^{(i)T} \frac{\partial X_k^*}{\partial V_k} e_{V_k}^{(i)}
 \tag{70}$$

For $A(i)$ we have two cases according to Eq. 61, the first is where $A(i) = 0 \Rightarrow \Delta\Phi(i) = \lambda B(i)$

We have then, $\Delta\Phi(i) < 0$, if $B(i) < 0$. And according to Eq. 70, $B(i) < 0$ only if one of the matrices $(\partial X_k^*/\partial V_k)^T$ is definite positive while the others are only positive semi-definite for $k = 0, 1, \dots, N - 1$.

The second case is where $A(i) > 0$, then the equation $\Delta\Phi(i) = 0$ has two distinct roots, 0 and $-(B(i)/A(i))$.

So $\Delta\Phi(i) < 0$ if $B(i) < 0$ (i.e., exactly one matrix of $(\partial X_k^*/\partial V_k)^T$ for $k = 0, 1, \dots, N - 1$ is definite positive and the others are only positive semi-definite according to Eq. 70 with λ chosen as :

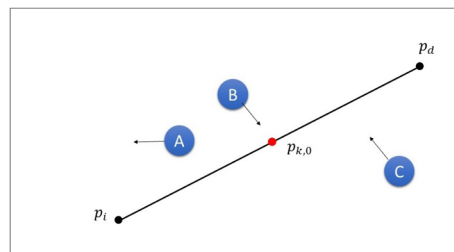
$$0 < \lambda < - \frac{B(i)}{A(i)} = \left| \frac{B(i)}{A(i)} \right|
 \tag{71}$$

The meticulous choice of the Lyapunov function helped to find the ideal values of our adaptive coefficient λ , by enabling a compromise to be reached, between maintaining the stability of the algorithm and increasing the speed of computation, which is not easy to achieve using the Gradient Method, as the coefficient λ is not bounded above by any value. In addition, this algorithm processes repeatedly all sub-problems at the lower level for every iteration of the coordination loop. \square

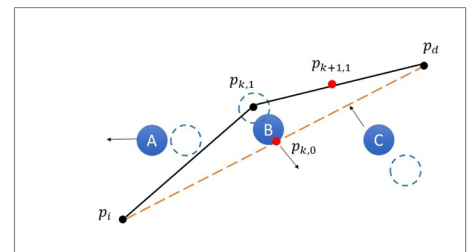
5 The Obstacle Avoidance

Now that we have presented the DC method, used for computing the optimal trajectory of a quadrotor UAV from an initial position $p_i(x_i, y_i, z_i)$ to a goal $p_d(x_d, y_d, z_d)$. This section, focuses on the description of the obstacle avoidance principle, which enables the autonomous UAV to navigate safely. In this approach, we suppose that the UAV is performing its task in an indoor environment. We also assume that the environment is completely supervised by a GCU, which communicates with the UAV via a wireless communication network. Initially, we can choose a desired position p_d (i.e. the system's input) then, the GCU sends instructions to the UAV, so that the later computes the optimal trajectory $T_0 = [p_i, p_1 \dots p_k \dots p_d]$ from its actual position p_i to p_d (see Fig. 4a) using the algorithm of DC. The computed trajectory T_0 is sent back after that to the GCU, to check whether the trajectory is safe or not,

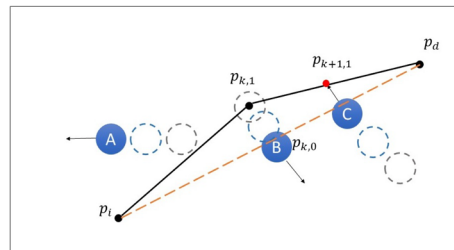
Fig. 4 The obstacle avoidance principle



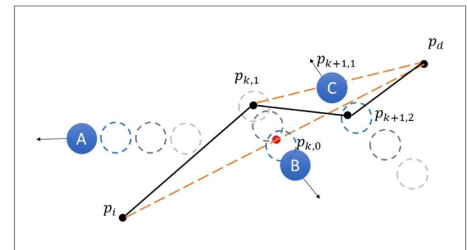
(a) The initial optimal trajectory T_0 computed by the UAV.



(b) The safe trajectory T_1 computed to avoid the first obstacle.



(c) The detection of a second obstacle.



(d) The final trajectory T_2 collision-free leading to the p_d .

Table 1 Values of quadrotor parameters used in this simulation

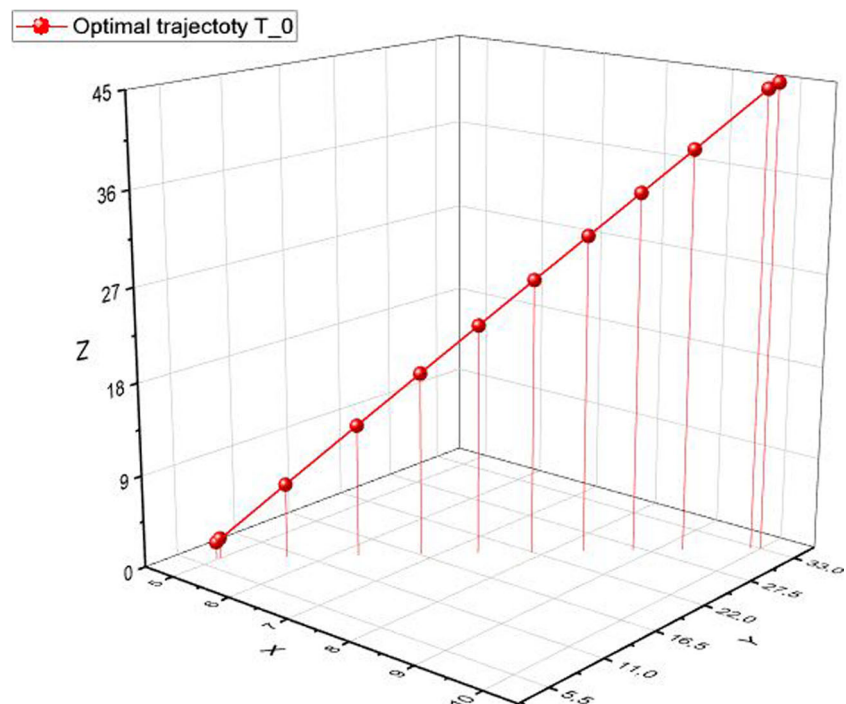
Parameter	Symbol	Value
Drag constant	d	1,31
Lift constant	k	3,74
Length of the quadrotor’s arm	l	0,2 m
The total mass of the UAV	m	0,5 kg
Gravitational acceleration	g	9,8 m/s ²
Moment of inertia along the x axis	I_x	$2e^3 \text{ kgm}^2$
Moment of inertia along the y axis	I_y	$2,9e^3 \text{ kgm}^2$
Moment of inertia along the z axis	I_z	$4,8e^3 \text{ kgm}^2$
Rotor’s inertia	I_r	$2,02e^5 \text{ kgm}^2$

and returns a decision. The UAV can start executing the trajectory and flies towards its goal in case the trajectory is safe. Otherwise, the UAV will receive the coordinates of a new intermediate goal $p_{k,1}$ from the GCU to be reached instead of the p_k the unsafe state in the initial trajectory T_0 . The UAV calculates the new safe trajectory T_1 from p_i to $p_{k,1}$ then, from $p_{k,1}$ to p_d , so that the new trajectory $T_1 = [p_i, p_{1,1} \dots p_{k,1}, p_{k+1,1} \dots p_d]$ allows the UAV to avoid the collision at the state p_k (see Fig. 4b). As soon as another obstacle appears, the GCU will find the next accurate intermediate goal $p_{k+1,2}$, as Fig. 4c shows, the UAV computes the next safe trajectory T_2 . These intermediate goals are supposed to be meticulously chosen, considering to the obstacles’ sizes, speeds and orientations. This process continues until the autonomous quadrotor arrives safely to the final destination p_d , as shown in Fig. 4d.

The pseudo-code below summarizes the obstacle avoidance principle:

- Start: $n = 0$;
- Compute the optimal trajectory $T_n = [p_i, p_{1,n}, \dots p_{k,n} \dots p_d]$ using DC algorithm from the actual position to the state p_d ;
- While p_d is no reached yet:
 - If an obstacle will collide with the UAV:
 - $n = n + 1$;
 - correct the trajectory compute a safe trajectory using DC algorithm from p_i to $p_{k,n}$ (given by the GCU) then to final destination p_d ;

Fig. 5 The optimal trajectory T_0 computed in a collision free environment (case.1), using the DC algorithm from the initial position p_i to the desired position p_d



- Otherwise continue to execute trajectory T_n ;
- Repeat until the state p_d is reached.

6 Simulation and Results

In this section, a simulation on Matlab is presented to show the relevance of the DC method in solving the trajectory planning problem for a small quadrotor UAV. In this simulation, we put $N = 10$. The quadrotor’s parameters used in this simulation are represented in Table 1.

In this simulation, we consider the first case where the environment is mainly safe, i.e. no static or dynamic obstacles are present.

Then, a second case, where the environment is hazardous and the UAV is supposed to avoid collisions by following the GCU’s instructions.

Case 1 We consider the following positions, randomly chosen:

- The initial position: $p_i = [5, 7, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0]^T$
- The desired position: $p_d = [10, 33, 45, 0, 0, 0, 0, 0, 0, 0, 0, 0]^T$

These two positions are the inputs of our system. The outputs of the system is the optimal trajectory $T_0 = [p_i, p_1 \dots p_k \dots p_d]$, and the associated control sequence $U_0 = [u_1, u_2 \dots u_k \dots u_{N-1}]$ computed for each instant t_k for $k \in [0, N - 1]$, where $p_k = [x_k, y_k, z_k, \dot{x}_k, \dot{y}_k, \dot{z}_k, \phi_k, \theta_k, \psi_k, \dot{\phi}_k, \dot{\theta}_k, \dot{\psi}_k]^T$ and $u_k = [u_{1,k}, u_{2,k}, u_{3,k}, u_{4,k}]^T$.

As shown in Fig. 5, the generated trajectory T_0 is obviously the most optimal (straight line), which confirms the efficiency of our algorithm.

The Fig. 6 illustrates the variation of the associated control $u_k = (u_{1,k}, u_{2,k}, u_{3,k}, u_{4,k})$. Along with these results, we can represent the variation of the four angular velocities $(\omega_1, \omega_2, \omega_3, \omega_4)$ of the UAV’s rotors (case 1) by solving the system Eq. 16, taking into consideration the following assumptions:

- The rotors 1 and 3 are rotating counterclockwise (i.e., the angular velocities ω_1 and ω_3 are positive).
- The rotors 2 and 4 are rotating clockwise (i.e., the angular velocities ω_2 and ω_4 are negative).

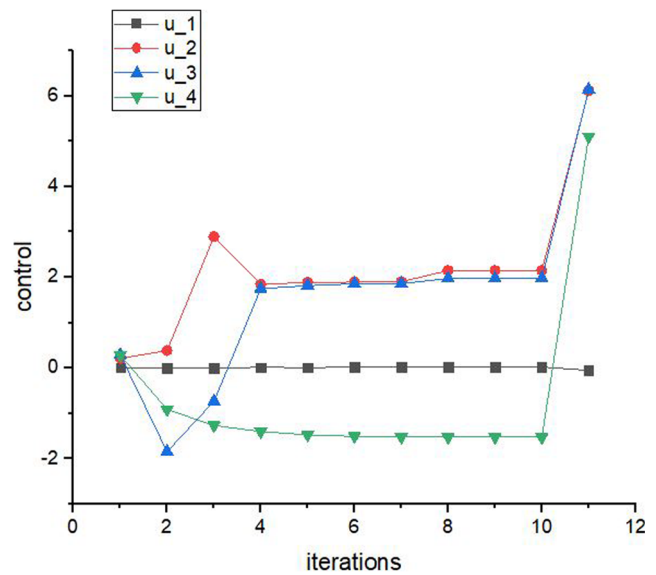


Fig. 6 The variation of the control $u_k = (u_{1,k}, u_{2,k}, u_{3,k}, u_{4,k})$ associated to the optimal trajectory T_0 , computed in a collision free environment (case.1)

We obtain the equations below, allowing as to represent the variation of angular velocities ω_i for $(i = 1, 2, 3, 4)$:

$$\omega_1 = +\sqrt{\frac{1}{4}(u_1/k - 2u_3/lk - u_4/d)}$$

$$\omega_2 = -\sqrt{\frac{1}{4}(u_1/k - 2u_2/lk + u_4/d)}$$

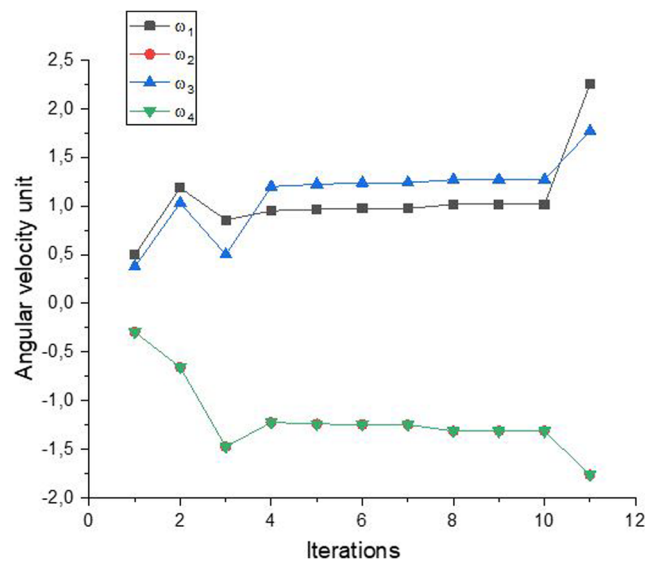
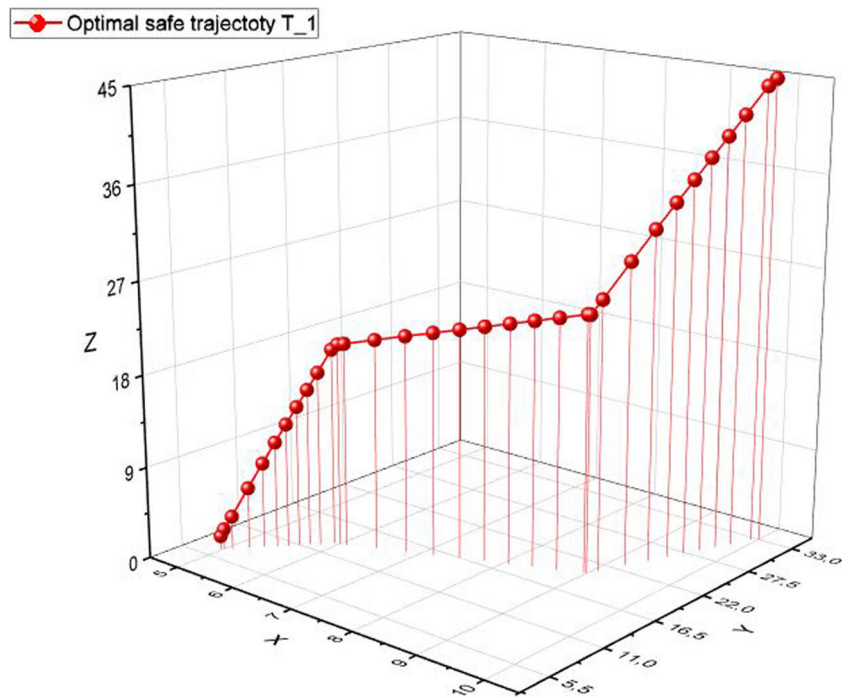


Fig. 7 The variation of the four angular velocities $(\omega_{1,k}, \omega_{2,k}, \omega_{3,k}, \omega_{4,k})$ at each state p_k for the optimal trajectory T_0 , computed in a collision free environment (case.1)

Fig. 8 The safe optimal trajectory T_1 computed according to the GCU instructions enabling the UAV to avoid the collisions (case. 2), where $T_1 = [T_{d,1}, T_{d,2}, T_d]$



$$\omega_3 = +\sqrt{\frac{1}{4}(u_1/k + 2u_3/lk - u_4/d)}$$

$$\omega_4 = -\sqrt{\frac{1}{4}(u_1/k + 2u_2/lk + u_4/d)}$$

The Fig. 7, shows the variation of the angular velocities for (case 1).

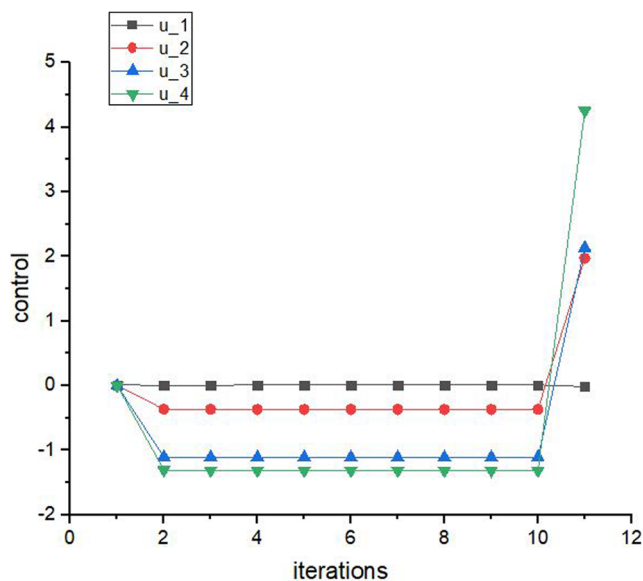


Fig. 9 The variation of the control $u_k = (u_{1,k}, u_{2,k}, u_{3,k}, u_{4,k})$ associated to the first optimal sub-trajectory $T_{d,1}$ (case. 2) (i.e. from p_i to $p_{d,1}$)

Case 2 In this case, we use the same inputs p_i and p_d used in Case 1, and we suppose that the environment is dynamic.

This time, the computed trajectory T_0 Fig. 5 is sent to the GCU, The latter checks whether there are any obstacles. We consider some virtual obstacles that UAV can avoid following a set of safe sub-trajectories calculated according to the instructions of GCU.

To avoid these virtual collisions, the GCU provides a first intermediate goal $p_{d,1}$, so that the UAV can calculate its first safe sub-trajectory $T_{d,1}$ from p_i to $p_{d,1}$. To ensure safe

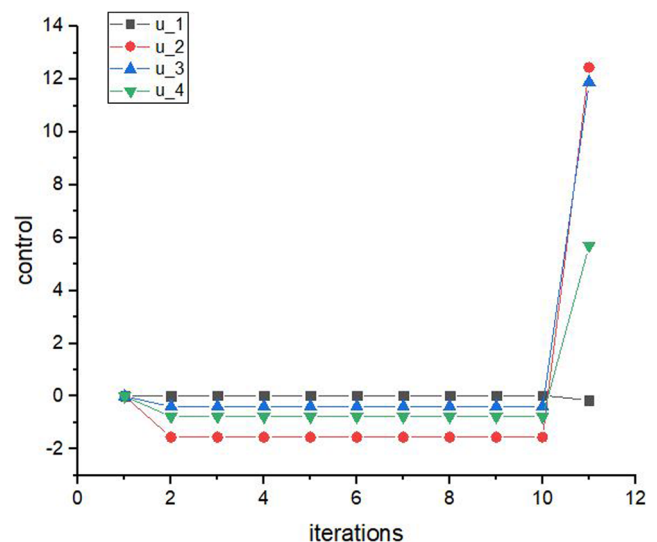


Fig. 10 The variation of the control $u_k = (u_{1,k}, u_{2,k}, u_{3,k}, u_{4,k})$ associated to the second optimal sub-trajectory $T_{d,2}$ (case. 2) (i.e. from $p_{d,1}$ to $p_{d,2}$)

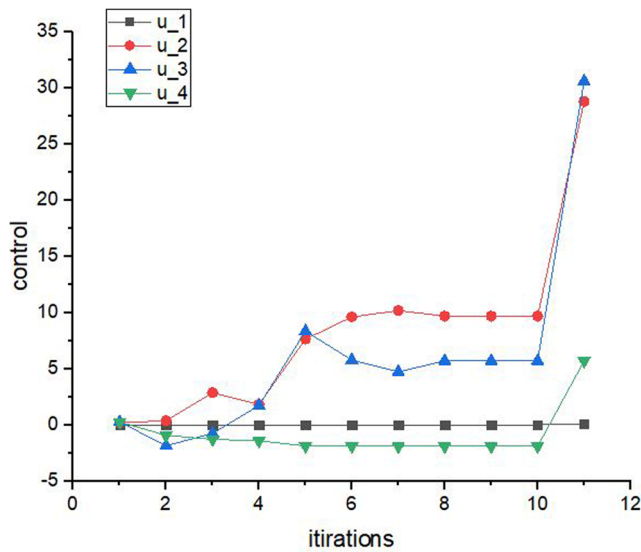


Fig. 11 The variation of the control $u_k = (u_{1,k}, u_{2,k}, u_{3,k}, u_{4,k})$ associated to the third optimal sub-trajectory T_d (case. 2) (i.e. from $p_{d,2}$ to p_d)

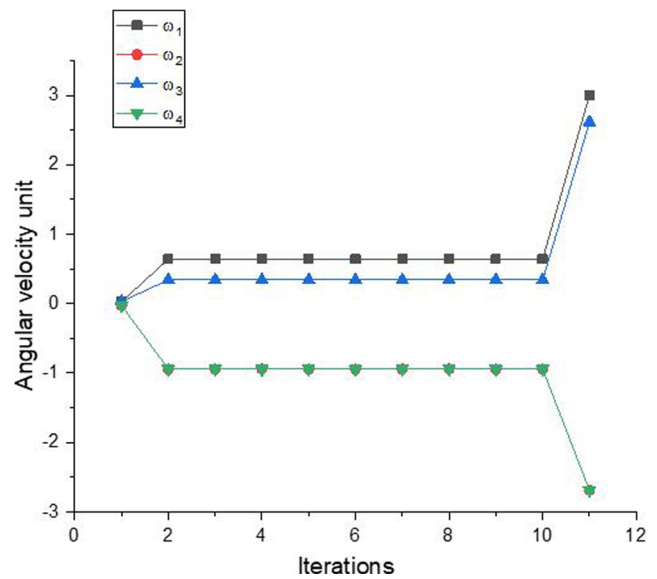


Fig. 13 The variation of the four angular velocities ($\omega_{1,k}, \omega_{2,k}, \omega_{3,k}, \omega_{4,k}$) at each state p_k for the second optimal sub-trajectory $T_{d,2}$, computed to avoid collisions (case.2)

and smooth navigation, the GCU monitors continuously the indoor environment, and gives a second goal $p_{d,2}$ allowing the UAV to avert more collisions, and compute the next safe sub-trajectory $T_{d,2}$ from $p_{d,1}$ to $p_{d,2}$.

After all of the obstacles have been successfully avoided, the GCU transmits information to the UAV, instructing it to compute the last sub-trajectory T_d toward the final goal p_d .

We put:

- The first intermediate goal: $p_{d,1} = [6, 13, 20, 0, 0, 0, 0, 0, 0]^T$.
- The second intermediate goal: $p_{d,2} = [9, 20, 25, 0, 0, 0, 0, 0, 0]^T$.

The Fig. 8 shows the safe trajectory $T_1 = [T_{d,1}, T_{d,2}, T_d]$, computed by the UAV in accordance with the GCU instructions. The Figs. 9, 10 and 11 depict the variation of

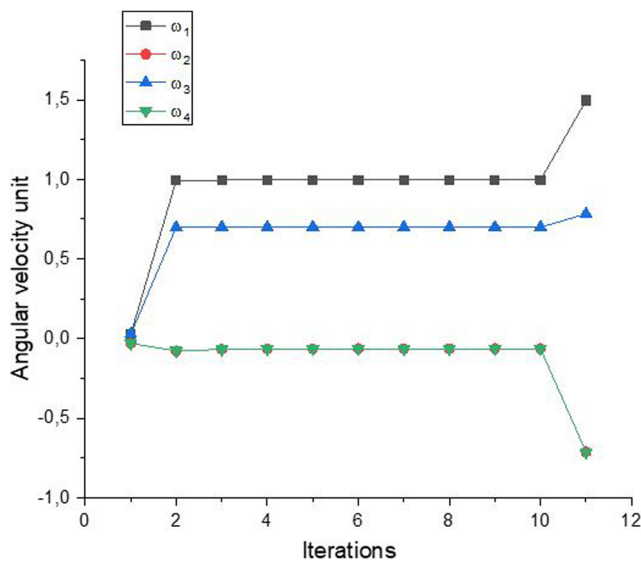


Fig. 12 The variation of the four angular velocities ($\omega_{1,k}, \omega_{2,k}, \omega_{3,k}, \omega_{4,k}$) at each state p_k for the first optimal sub-trajectory $T_{d,1}$, computed to avoid collisions (case.2)

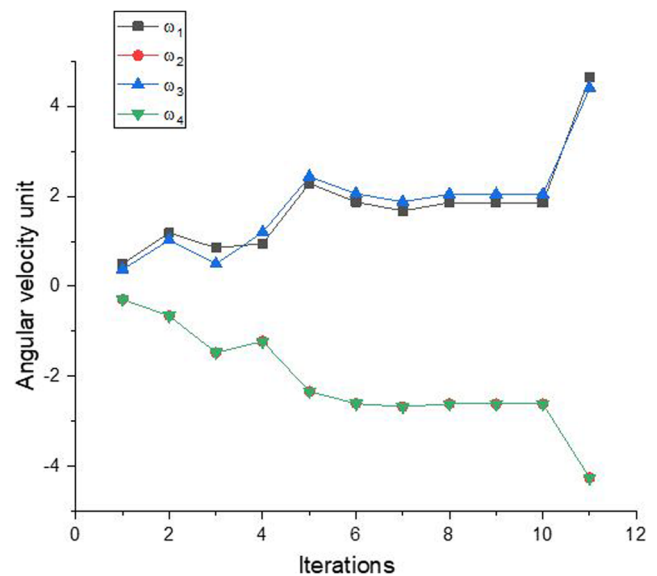


Fig. 14 The variation of the four angular velocities ($\omega_{1,k}, \omega_{2,k}, \omega_{3,k}, \omega_{4,k}$) at each state p_k for the third optimal sub-trajectory T_d , computed to avoid collisions (case.2)

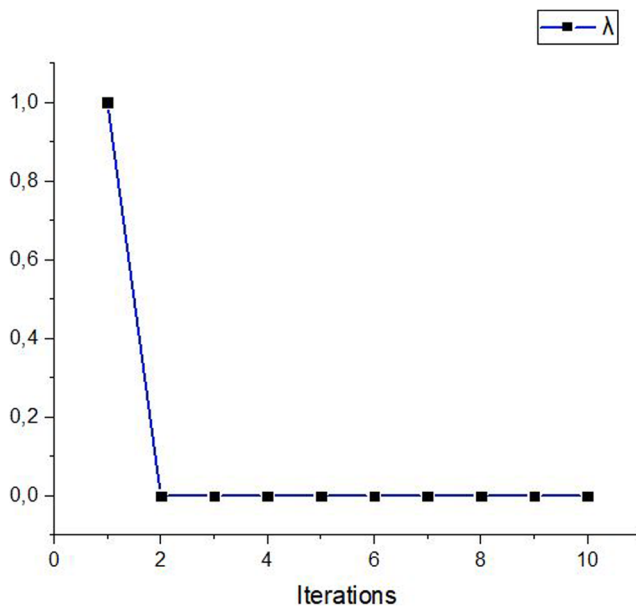


Fig. 15 The evolution of λ the adaptive coefficient, adjusted according to the condition $0 < \lambda < |B(i)/A(i)|$

the associated control u_k of the sub-trajectories $T_{d,1}$, $T_{d,2}$, and T_d respectively. In the Figs. 12, 13 and 14, we represent the variation of the angular velocities of the four rotors, ω_i (for $i = 1, 2, 3, 4$), respectively for the sub-trajectories $T_{d,1}$, $T_{d,2}$ and T_d .

The algorithm generates also the values of the adaptive coefficient λ , adjusted according to the condition $0 < \lambda < |B(i)/A(i)|$, as the Fig. 15 shows, the coefficient λ converges to 0 from the second iteration which allows a faster computation. This adaptive coefficient λ converges the same way each time (i.e. for all the sub-trajectories).

7 Conclusion

Optimal control of nonlinear systems presents several challenges, owing mostly to the complexity of the models and the rules of command that we intend to develop. The optimal navigation problem studied in this work shows the potential of the DC approach, and how it may be profitable for the management of different nonlinear dynamic systems requiring a great amount of computations. Several methods based on genetic algorithms have been utilized to address nonlinear optimization problems. However, these approaches have a significant drawback so that they may fail to solve nonlinear problems when the problem's constraints become too difficult to satisfy, or when the objective space is non-convex, causing the genetic algorithm to converge to the optimal Pareto front with difficulty. Furthermore, these algorithms could find any bound-optimal Pareto front of

the problem, which is not always the best. In light of what we've seen so far, our approach based on the decomposition-coordination technique, looks to be a very efficient way to solve this type of problem, since the resolution at the lower level entails sub-problems involving a limited number of variables, which is much easier than processing an overall non-linear problem. Moreover, because the sub-problems are separable, they may be handled in parallel simultaneously, making the DC approach easily extensible to analog neural networks. To guarantee safe navigation, the GCU monitors the environment and provides the required directives, allowing the UAV to calculate safe sub-trajectories until it arrives at its target. The strength of this approach lies on the overall planning of trajectory, giving the UAV enough freedom to avoid eventual collisions. Our future work will focus on implementing the DC method on a neural network, which will be supported by an experimental study of navigation in a real environment.

Acknowledgements The authors would like to thank the project H2020-MSCARISE-2017 research and innovation program under the Marie Marie Skłodowska-CuriCurie grant agreement No. 777720 for their support.

Author Contributions The contributions in relation with this submission:

- Imane Nizar: [12] and [15]
- Adil Jaafar: no contributions in relation with this submission
- Zineb Hidila: no contributions in relation with this submission
- Mohamed Barki: no contributions in relation with this submission
- El Hossein Illoussamen: [12] and [15]
- Mohammed Mestari : [12], [14], [15] and [18]

Funding No funding was received for this work.

Availability of data and materials All data generated or analyzed during this study are included in this article.

Declarations

Ethics approval Not applicable (no procedures involving human participants were performed in this study).

Consent to participate Not applicable (no participants).

Consent for Publication All authors give their consent for the publication of this work.

Conflict of Interests All authors declare that they have no competing interests.

References

1. Dalamagkidis, K., Valavanis, K.P., Piegl, L.A.: Aviation history and unmanned flight. In: On Integrating Unmanned Aircraft Systems into the National Airspace System, pp. 11–42. Springer, Dordrecht (2012)

2. Springer, P.J.: Military robots and drones: a reference handbook. ABC-CLIO, Santa Barbara, Calif (2013)
3. Faiçal, B.S., Costa, F.G., Pessin, G., Ueyama, J., Freitas, H., Colombo, A., Fini, P.H., Villas, L., Osório, F.S., Vargas, P.A., Braun, T.: The use of unmanned aerial vehicles and wireless sensor networks for spraying pesticides. *J. Syst. Archit.* **60**, 393–404 (2014). <https://doi.org/10.1016/j.sysarc.2014.01.004>
4. Euchi, J.: Do drones have a realistic place in a pandemic fight for delivering medical supplies in healthcare systems problems. *Chinese J. Aeron.* <https://doi.org/10.1016/j.cja.2020.06.006> (2020)
5. Zhang, R., Zhang, J., Yu, H.: Review of modeling and control in UAV autonomous maneuvering flight. In: 2018 IEEE International Conference on Mechatronics and Automation (ICMA), pp. 1920–1925. IEEE, Changchun (2018)
6. Chuang, H.-M., He, D., Namiki, A.: Autonomous target tracking of UAV using high-speed visual feedback. *Appl. Sci.* **9**, 4552 (2019). <https://doi.org/10.3390/app9214552>
7. Saad, W., Bennis, M., Mozaffari, M., Lin, X.: *Wireless Communications and Networking for Unmanned Aerial Vehicles*. Cambridge University Press, Cambridge (2020)
8. Primatesta, S., Guglieri, G., Rizzo, A.: A risk-aware path planning strategy for UAVs in urban environments. *J Intell Robot Syst.* **95**, 629–643 (2019). <https://doi.org/10.1007/s10846-018-0924-3>
9. Delamer, J.-A., Watanabe, Y., P. Carvalho Chanel, C.: Solving path planning problems in urban environments based on a priori sensors availabilities and execution error propagation. In: AIAA Scitech 2019 Forum. American Institute of Aeronautics and Astronautics, San Diego, California (2019)
10. Galvez, R.L., Dadios, E.P., Bandala, A.A.: Path planning for quadrotor UAV using genetic algorithm. In: 2014 International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM). pp. 1–6 (2014)
11. Gautam, S.A., Verma, N.: Path planning for unmanned aerial vehicle based on genetic algorithm and artificial neural network in 3D. In: 2014 International Conference on Data Mining and Intelligent Computing (ICDMIC). pp. 1–5 (2014)
12. Nizar, I., Illoussamen, Y., El Ouarak, H., Hossein Illoussamen, E., Grana (Graña), M., Mestari, M.: Safe and optimal navigation for autonomous multi-rotor aerial vehicle in a dynamic known environment by a decomposition-coordination method. *Cognit. Syst. Res.* **63**, 42–54 (2020). <https://doi.org/10.1016/j.cogsys.2020.05.003>
13. Sandino, J., Vanegas, F., Maire, F., Caccetta, P., Sanderson, C., Gonzalez, F.: UAV framework for autonomous onboard navigation and people/object detection in cluttered indoor environments. *Remote Sensing.* **12**, 3386 (2020). <https://doi.org/10.3390/rs12203386>
14. Mestari, M., Benzirar, M., Saber, N., Khouil, M.: Solving nonlinear equality constrained multiobjective optimization problems using neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **26**, 2500–2520 (2015). <https://doi.org/10.1109/TNNLS.2015.2388511>
15. Nizar, I., Illoussamen, Y., Illoussamen, E.H., Mestari, M., Hamlich, M., Bellatreche, L., Mondal, A., Ordonez, C. (eds.): *Safe and optimal path planning for autonomous UAV using a decomposition-coordination method*. Springer International Publishing, Cham (2020)
16. Huang, T., Huang, D., Wang, Z., Dai, X., Shah, A.: Generic adaptive sliding mode control for a Quadrotor UAV system subject to severe parametric uncertainties and fully unknown external disturbance. *Int. J. Control Autom. Syst.* <https://doi.org/10.1007/s12555-019-0853-3> (2020)
17. Chen, F., Jiang, R., Zhang, K., Jiang, B., Tao, G.: Robust backstepping sliding mode control and observer-based fault estimation for a Quadrotor UAV. *IEEE Trans. Ind. Electron.* 1–1. <https://doi.org/10.1109/TIE.2016.2552151> (2016)
18. Khouil, M., Sanou, I., Mestari, M., Aitelmahjoub, A.: Planification of an optimal path for a mobile robot using neural networks. *ams.* **10**, 637–652 (2016). <https://doi.org/10.12988/ams.2016.510653>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Imane Nizar received the Engineering degree in industrial logistics from the École Normale Supérieure de l'Enseignement Technique (ENSET) Mohammed VI, Casablanca, Morocco, in 2015. She is currently preparing a thesis about optimal and safe navigation of autonomous UAVs using a decomposition-coordination algorithm. Her current research interests include optimal control of nonlinear systems, trajectory planning, obstacle avoidance, and autonomous UAV.

Adil Jaafar received the master's degree in Distributed Information Systems in 2015 from the École Normale Supérieure de l'Enseignement Technique Mohammed VI/Morocco at University of Hassan II Casablanca. He is currently preparing a thesis titled cooperative distributed control for multi-agent systems, event of a fleet of Drones or Robots moving in an open and cluttered environment. His current research interests include multi-agent systems and neural network NECMOP implementation.


Zineb Hidila received a Ms degree in engineering and optimization of transport and logistics systems in faculty of sciences - Hassan II University-Casablanca Morocco in 2015. Currently pursuing a Ph.D. program concerning airspace optimization and geometrical computation. The current research interests are metaheuristics, neural networks implementations, and geographical information systems.

Mohamed Barki received an Engineering degree in computer science from the Ecole Nationale de l'Industrie Minérale (ENIM) Rabat Morocco, in 1999. He is currently preparing a thesis in the field of artificial intelligence.

El Hossein Illoussamen received his Ph.D. from the University Mohamed V of Rabat, in 1996. He is currently a full Professor in the department of mathematics and informatics at ENSET Mohammed VI, University Hassan II of Casablanca. His research interests include various domains of functional analysis, particularly in automatic continuity problems of operators. Since 2012, he is the head of the functional analysis and optimization team within the laboratory of signals, distributed systems, and artificial intelligence.

Mohammed Mestari received the M.A. degree from the cole Normale Supérieure de l'Enseignement Technique (ENSET), Mohammedia, Morocco, in 1991, and the Ph.D. degrees in applied mathematics and artificial intelligence from the Faculty of Science Ben M'Sick, Hassan II University, Casablanca, Morocco, in 1997 and 2000, respectively. He is currently a Professor of artificial intelligence at ENSET, and the Head of the Artificial Intelligence Research Team with the Laboratory of Signals, Distributed Systems, and Artificial Intelligence. He is Co-founder of the International Neural Network Society Morocco Chapter (INNS Morocco Chapter), Co-founder of the IEEE Computational Intelligence Morocco Chapter, and Member of the IEEE Transactions on Neural Network and Learning Systems (IEEE TNNLS). His current research interests include neural networks for image classification, data-driven approach, neural networks hardware implementation, high-speed and/or low-power techniques and systems for neural networks, and theoretical issues directly related to hardware implementation of techniques based on the principle of decomposition coordination for optimal control and trajectory planning for Unmanned Aerial Vehicles (UAVs) and mobile robots.

Affiliations

Imane Nizar¹  · Adil Jaafar¹ · Zineb Hidila¹ · Mohamed Barki¹ · El Hossein Illoussamen¹ · Mohammed Mestari¹

Adil Jaafar
jaafar.adil@gmail.com

Zineb Hidila
zineb.hidila@gmail.com

Mohamed Barki
mohamedbarki7@gmail.com

El Hossein Illoussamen
illous@hotmail.com

Mohammed Mestari
mestari@enset-media.ac.ma

¹ Laboratory SSDIA, École Normale Supérieure de l'Enseignement Technique (ENSET) Mohammedia 20800, University Hessian II, Casablanca, Morocco