Robotica (2018) volume 36, pp. 1363–1385. © Cambridge University Press 2018
doi:10.1017/S026357471800036X

Constrained generalized predictive control for obstacle avoidance in a quadcopter

José Luis Mendoza-Soto†*, Luis Alvarez-Icaza‡ and H. Rodríguez-Cortés§

†División de Ingeniería Eléctrica, Facultad de Ingeniería,
Universidad Nacional Autónoma de México,
Av. Universidad 3000, Ciudad Universitaria, Coyoacán, Ciudad de México 04510, México
‡Instituto de Ingeniería, Universidad Nacional Autónoma de México,
Av. Universidad 3000, Ciudad Universitaria, Coyoacán, Ciudad de México 04510, México.
E-mail: alvar@pumas.iingen.unam.mx
§Sección de Mecatrónica, Departamento de Ingeniería Eléctrica,
Centro de Investigación y de Estudios Avanzados del Instituto
Politécnico Nacional, Av. Instituto Politécnico Nacional 2508,
Col. San Pedro Zacatenco, Gustavo A. Madero, Ciudad de México 07360, México.
E-mail: hrodriguez@cinvestav.mx

(Accepted April 28, 2018. First published online: June 6, 2018)

SUMMARY
This work proposes a strategy for position control and obstacle avoidance in a quadcopter based on constrained generalized predictive control and geometric attitude control. The approach allows real-time trajectory tracking using optimal control actions and avoids collisions with static obstacles whose position is known. An experimental validation of the proposed controller is presented.

KEYWORDS: Generalized predictive control; Quadcopter; Unmanned aerial vehicle; Obstacle avoidance.

1. Introduction

Unmanned aerial vehicles (UAVs) have had a boom in their production over the last few years due to their large number of applications. The quadcopter is a type of UAV with the capacity to perform stationary flight; therefore, this vehicle found applications in areas such as entertainment, surveillance, package delivery and even military use. Quadcopters have characteristics that represent a challenge for control design such as sub actuation and nonlinearities. A review of control approaches for quadcopter vehicles is performed, for instance in ref. [22]. Algorithms analyzed in this review range from linear to intelligent control techniques. The study describes advantages and disadvantages of applying the different algorithms to quadcopters, concerning robustness, optimality, trajectory performance, and computational simplicity among other characteristics. In novel applications, one of the functions that a quadcopter must fulfill is trajectory tracking in the presence of obstacles. Model-based predictive control (MPC) is an approach that allows the incorporation of constraints on the control objective; thus, it has the capability to meet the avoiding collisions objective.

The computational burden increases with the level of detail of the dynamic model used for MPC synthesis; therefore, implementation of obstacle avoidance in an aerial vehicle by employing MPC must use efficient computational strategies and tailored dynamic models. Hence, different approaches are considered to simplify the nonlinear quadrotor dynamic model and to implement MPC in real time. The authors of ref. [11] compensate, using a state observer and a disturbances estimator, the nonlinear part of the quadrotor dynamics for being able to approximate the closed-loop dynamics to a

* Corresponding author. E-mail: eemsj03@yahoo.com.mx

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

linear system where the generalized predictive control (GPC) technique is applied. The authors present numerical simulations to evaluate the proposed control scheme without constraints. The work in ref. [2] simplifies the quadcopter dynamics by decoupling the longitudinal dynamics of the rotational dynamics and considering that the Euler angles are small. Then, the authors propose an MPC scheme for the resulting dynamics. The proposed strategy is experimentally tested on a quadcopter Quanser Qball-X4.

Other works based on approximate linearization and MPC are given in refs. [3] and [15]. In ref. [3], a decoupling between rotational and translational dynamics is considered. Perturbation terms are included in the linear model, and the MPC approach is applied to obtain a robust MPC. The algorithm proposed in ref. [15] consists of linearizing the quadcopter model at each sample step around operation points produced by an initial stable control sequence. Then, the unconstrained MPC problem is solved with the sequential linear quadratic (SLQ) optimization algorithm. The last two papers present experimental results. The work presented in ref. [4] performs the reduction of the quadcopter dynamics, as a result of applying a high gain controller to the rotational dynamics. Then, a linear model is obtained with the aid of an input-output linearizing feedback controller. Finally, the authors use an unconstrained MPC algorithm to command the resulting linear system. Experimental results showing the performance of the controller are presented. The linear MPC operates at 16 Hz, while the high gain controller operates at 200 Hz on a PX4 autopilot.

Optimization methods are also base for trajectory planning algorithms. In refs. [19], [18] and [16], a mixed integer linear programming (MILP) approach is used to plan obstacle-free vehicle trajectories. A trajectory planning scheme for optimal fuel consumption in a vehicle is considered in ref. [19]. In refs. [18] and [16], an optimal control strategy to perform maneuvers and command the relative position of several vehicles without collisions is presented. The results of these works are validated using numerical simulations so that the effects of non-modeled dynamics, noise and real-time issues are not considered. In ref. [21], a discrete-time non-linear MPC approach with constraints is used to land a quadrotor on a mobile inclined platform. Experimental results with a Pioneer mobile robot and a Parrot ARDrone quadrotor verify the proposed control algorithm.

The mixed integer quadratic programming (MIQP) approach is applied in works given in refs. [13] and [5]. In ref. [13], experiments for collision avoidance in a group of quadcopters using a trajectory planner based on waypoints are presented; the authors use a control based on an error metric defined in the space $so(3)$. A combination of trajectory planning and MPC is presented in ref. [5], where a two-layer hybrid hierarchical MPC control approach is proposed for trajectory tracking. The MPC approach in conjunction with MIQP method is based on the linear approximation of the quadrotor dynamics around hover. Numerical simulations validate the proposed scheme. In ref. [14], it is proposed a trajectory planner for the quadcopter translational dynamics using an optimal control approach assuming that the rotational dynamics is faster than the translational dynamics. The trajectory planner minimizes a jerk-based cost function with the desired start and end states being the boundary conditions. Experimental results show the operation of the proposed approach. In ref. [8], an MPC-based trajectory planner is presented to control a quadcopter system using the I/O linearization technique and a proportional control. Experiments are performed with the Pelican quadcopter from Ascending Technologies (AscTec) endowed with an Intel Core i7 processor. Other MPC approaches applied to the control of group of quadcopters are presented in ref. [6] with a hierarchical MPC strategy and in ref. [17], where a nonlinear MPC strategy is proposed to determine a control sequence close to the optimal control sequence for cooperative vehicles using a nonlinear kinematic model for each vehicle.

The issue of stability in MPC is still an open problem; however, partial results under the absence of constraints, infinite prediction horizons or imposing terminal constraints are available. The main approaches for stability analysis of MPC are surveyed in refs. [12] and [7]. The imposition of final constraints assures stability in any prediction horizon; this is verified taking the cost function as a Lyapunov function and showing that it is monotonically decreasing. Under the infinite horizons condition, it is possible to show that the cost function is monotonically decreasing, guaranteeing stability. One way to test stability using a finite horizon without constraints is through the utilization of the fake algebraic Riccati technique; stability can be assured as in the Linear quadratic regulator (LQR) technique. It is also possible to ensure stability by obtaining the Youla parameterization for the closed-loop system. In ref. [7], the approach called Stable generalized predictive control (SGPC) ensures that the associated cost function is monotonically decreasing.

This work proposes a strategy to track trajectories and to avoid static obstacles whose position is known in a quadcopter using a GPC with a MIQP algorithm, and a geometric attitude controller. It

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

is assumed that linear constraints can model the position of obstacles. The high computational load in the optimization algorithms is reduced taking advantage of the cascade interconnection between the translational and rotational quadrotor dynamics and using two control loops. The first control loop solves the trajectory tracking problem combining a GPC translational controller and a geometric attitude controller. The second control loop takes care of the constraints on the desired trajectory. This second control loop is designed from a linear model obtained by identifying offline the closed-loop translational dynamics; moreover, to solve the resulting constrained GPC problem employs MIQP. The proposed scheme is tested experimentally in a quadcopter equipped with an attitude and heading reference system (AHRS) and a Texas Instruments digital signal processor (DSP). An Optitrack positioning system determines the quadrotor position. The GPC algorithms are calculated on a personal computer because the available memory on the onboard DSP is not enough. The attitude control is performed on the onboard DSP.

The main contributions of this work are as follows. It proposes to exploit the cascade structure of the quadrotor dynamics to separate a linear translational dynamics. Then, the GPC controller solves the constrained trajectory tracking problem for the linear translational dynamics, and the geometric attitude controller takes care of the interconnection terms. The literature, to the best of authors knowledge, neither reports the application of the GPC method to solve the constrained trajectory control problem for quadrotors nor any strategy that shows the use of the MIQP algorithm in the GPC technique for collision avoidance. Additionally, it suggests a scheme to perform a real-time experimental validation of the proposed controller using a personal computer and a homemade autopilot due to the computational load. During the experimental flights the constraints model obstacles that the quadrotor must avoid. The experimental results show that the proposed controller achieves the control objective, and provide information to analyze the influence of some controller parameters on the closed-loop behavior.

The organization of this article is as follows. After a short introduction to the concept of GPC, in Section 3, the model of a quadcopter helicopter is described. Section 4 shows the developments to synthesize the proposed controller. Section 5 presents experimental results. Finally, Section 6 presents the conclusions of this work.

2. Generalized Predictive Control (GPC)

In recent years, model-based predictive control (MPC) has been widely used in industry to solve regulation and trajectory tracking problems due to its ability to handle perturbations, constraints, parametric uncertainty and delays. The MPC approach utilized in this work, for the two translational control loops, is the GPC. The GPC technique has some advantages over traditional Non-linear model-based predictive control (NMPC); for example, in the case of absence of constraints, it is possible to obtain an analytical solution, it can deal with unstable or non-minimal-phase systems, and it has an additional adjustment parameter called the control horizon.[7] Also, GPC allows obtaining predictions with optimal criteria when solving a Diophantine equation and it uses an input–output approach with the CARIMA model; thus, it avoids measurements of the outputs time derivatives.

The GPC approach uses a CARIMA (Auto Regressive Integrated Average Mobile Controller) model with n outputs and m inputs[7] of the form

$$\mathbf{A}\left(z^{-1}\right)\mathbf{y}\left(t\right) = \mathbf{B}\left(z^{-1}\right)\mathbf{u}\left(t-1\right) + \frac{1}{\Delta}\mathbf{C}\left(z^{-1}\right)\mathbf{e}\left(t\right) \tag{1}$$

where $\mathbf{A}(z^{-1})$ and $\mathbf{C}(z^{-1})$ are monotonic polynomial matrices of order $n \times n$ and $\mathbf{B}(z^{-1})$ is a polynomial matrix of order $n \times m$. The operator Δ is defined as $\Delta = 1 - z^{-1}$. The variables $\mathbf{y}(t)$, $\mathbf{u}(t)$ and $\mathbf{e}(t)$ are output, control and noise vectors of $n \times 1$, $m \times 1$ and $n \times 1$ dimensions, respectively. It is common to consider for simplicity $\mathbf{e}(t)$ as white noise. The objective of the GPC is to find the control sequence that minimizes the following quadratic criterion with finite horizon:

$$J_c\left(N_p, N_u\right) = \sum_{j=1}^{N_p} \left\|\hat{\mathbf{y}}\left(t+j|t\right) - \mathbf{w}\left(t+j\right)\right\|^2_{\mathbf{R}_\mathbf{y}} + \sum_{j=1}^{N_u} \left\|\Delta\mathbf{u}\left(t+j-1\right)\right\|^2_{\mathbf{Q}_\mathbf{u}} \tag{2}$$

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

where $\hat{\mathbf{y}}(t+j|t)$ is an optimal output prediction j steps ahead of the output of the system calculated at time t. It is possible to compute the optimal prediction if the past input and output vectors, and the future sequence reference are known. N_p and N_u are the prediction and control horizons, respectively. $\mathbf{w}(t+j)$ is the vector of the sequence reference for the output vector. $\mathbf{R_y}$ and $\mathbf{Q_u}$ are positive define weighting matrices and $\|\cdot\|$ the Euclidean vector norm.

In the simplest GPC case, the matrix that models the noise is defined as $\mathbf{C}(z^{-1}) = \mathbf{I}_{n\times n}$. Then, the prediction for the output takes the following form:

$$\mathbf{y}(t+j) = \mathbf{F}_j\left(z^{-1}\right)\mathbf{y}(t) + \mathbf{E}_j\left(z^{-1}\right)\mathbf{e}(t+j) + \mathbf{E}_j\left(z^{-1}\right)\mathbf{B}\left(z^{-1}\right)\Delta\mathbf{u}(t+j-1) \tag{3}$$

where $\mathbf{E}_j(z^{-1})$ and $\mathbf{F}_j(z^{-1})$ are recursively obtained as solutions of the following Diophantine matrix equation:

$$\mathbf{I}_{n\times n} = \mathbf{E}_j\left(z^{-1}\right)\tilde{\mathbf{A}}\left(z^{-1}\right) + z^{-j}\mathbf{F}_j\left(z^{-1}\right) \tag{4}$$

Defining $\mathbf{E}_j(z^{-1})\mathbf{B}(z^{-1}) = \mathbf{G}_j(z^{-1}) + z^{-j}\mathbf{G}_{jp}(z^{-1})$ with $\deg(\mathbf{G}_j(z^{-1})) < j$, the prediction equation can be written as

$$\hat{\mathbf{y}}(t+j|t) = \mathbf{G}_j\left(z^{-1}\right)\Delta\mathbf{u}(t+j-1) + \mathbf{G}_{jp}\left(z^{-1}\right)\Delta\mathbf{u}(t-1) + \mathbf{F}_j\left(z^{-1}\right)\mathbf{y}(t) \tag{5}$$

Eq. (5) can be expressed in compact form as

$$\mathbf{y} = \mathbf{G}\mathbf{u} + \mathbf{f} \tag{6}$$

with $\mathbf{f}_j = \mathbf{G}_{jp}(z^{-1})\Delta\mathbf{u}(t-1) + \mathbf{F}_j(z^{-1})\mathbf{y}(t)$. The terms that appear evaluated ahead on time are neglected.

Note that Eq. (2) can be written as

$$J_c = (\mathbf{G}\mathbf{u} + \mathbf{f} - \mathbf{w})^\top \bar{\mathbf{R}}_\mathbf{y}(\mathbf{G}\mathbf{u} + \mathbf{f} - \mathbf{w}) + \mathbf{u}^\top\bar{\mathbf{Q}}_\mathbf{u}\mathbf{u} \tag{7}$$

where $\bar{\mathbf{R}}_\mathbf{y} = \mathrm{diag}(\mathbf{R_y}, \ldots, \mathbf{R_y})$ and $\bar{\mathbf{Q}}_\mathbf{u} = \mathrm{diag}(\mathbf{Q_u}, \ldots, \mathbf{Q_u})$. If there are no constraints, the optimal control can be expressed as

$$\Delta\mathbf{u}(t) = \mathbf{K}(\mathbf{w} - \mathbf{f}) \tag{8}$$

where $\mathbf{K} = (\mathbf{G}^\top\bar{\mathbf{R}}\,\mathbf{G} + \bar{\mathbf{Q}}_\mathbf{u})^{-1}\mathbf{G}^\top\bar{\mathbf{R}}_\mathbf{y}$. Because the GPC strategy uses the receding horizon approach, only the component $\Delta\mathbf{u}(t)$ at time t is necessary, and the other components of $\Delta\mathbf{u}(t)$ are discarded.

In the case of a control problem with constraints, it is necessary to solve a quadratic programming problem at each sample step to get the optimal control sequence. There is not an analytical expression for the control policy. The constraints can define limits on the control signals, the desired trajectory, the maximal overshoot, among other characteristics of the response of the system. In this case, it is a problem of quadratic optimization of the cost function (2) subject to constraints of the form

$$\mathbf{R}_r\mathbf{u} \leq \mathbf{c} \tag{9}$$

with

$$\mathbf{R}_r = \begin{bmatrix} \mathbf{I}_{N\times N} \\ -\mathbf{I}_{N\times N} \\ \mathbf{G} \\ -\mathbf{G} \end{bmatrix}; \qquad \mathbf{c} = \begin{bmatrix} \mathbf{L}\bar{\mathbf{u}} \\ -\mathbf{L}\underline{\mathbf{u}} \\ \mathbf{L}\bar{\mathbf{y}} - \mathbf{f} \\ -\mathbf{L}\underline{\mathbf{y}} + \mathbf{f} \end{bmatrix} \tag{10}$$

where $N = N_p$, $\mathbf{L}$ is a matrix of order $(N\times n)\times m$ formed by N identity matrices of order $m\times m$. The variables with upper and lower bars represent the upper and lower limits of the inputs and outputs, respectively.

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

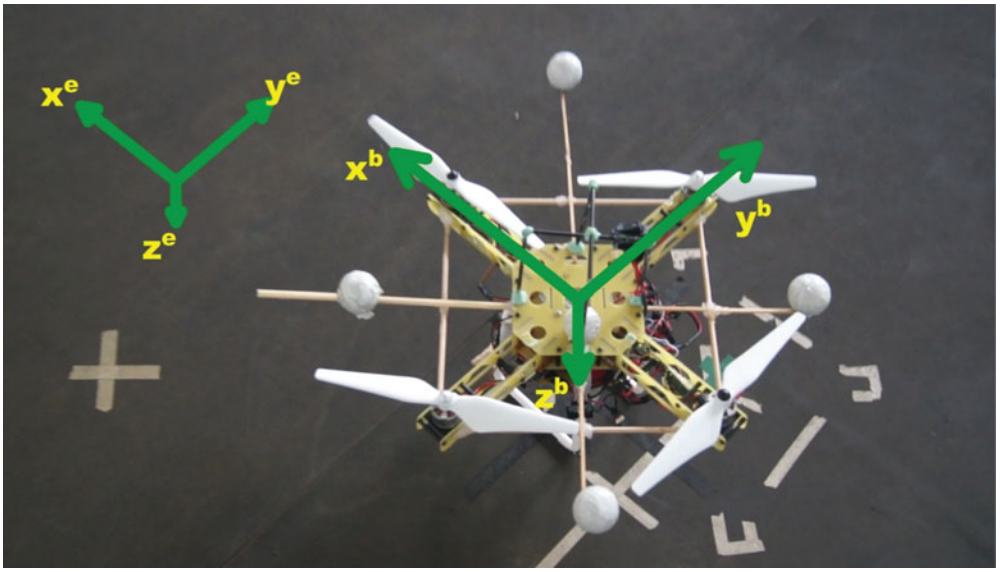


Fig. 1. Quadrotor vehicle. The triads $x^b y^b z^b$ and $x^e y^e z^e$ represent the body and inertial coordinate frames, respectively.

3. Model

The quadcopter model is defined according to the reference frames shown in Fig. 1, where (x^b, y^b, z^b) and (x^e, y^e, z^e) are the inertial and body reference frames, respectively. The quadcopter dynamics[20] is given by

$$\mathbf{J}\dot{\boldsymbol{\Omega}} + \boldsymbol{\Omega} \times \mathbf{J}\boldsymbol{\Omega} = \mathbf{M_e} \tag{11}$$

$$\dot{\mathbf{R}} = \mathbf{R}\hat{\boldsymbol{\Omega}} \tag{12}$$

$$\ddot{\mathbf{x}} = g\mathbf{e_3} - \frac{1}{m_a} T_T \mathbf{R}\mathbf{e_3} \tag{13}$$

where $\boldsymbol{\Omega} = [\,p \quad q \quad r\,]^\top$ is the vector of angular velocity around the axes x^b, y^b and z^b, and $\mathbf{x} = [\,x \quad y \quad z\,]^\top$ is the translational position vector. The map $\hat{\cdot} : \mathbb{R}^3 \to \mathbf{so}(3)$ is defined as $\hat{x}\, y = x \times y$, where $so(3)$ is the space of 3×3 antisymmetric matrices. Additionally, $\mathbf{M_e}$ is the vector of external moments acting on the quadrotor frame defined as

$$\mathbf{M_e} = \begin{bmatrix} (T_2 - T_4)\, l \\ (T_1 - T_3)\, l \\ -Q_1 + Q_2 - Q_3 + Q_4 \end{bmatrix} \tag{14}$$

where T_i and Q_i with $i = 1, \ldots, 4$ are the thrust and the reactive torque, respectively, produced by the rotors and l defines the length of the quadcopter arms with respect to its gravity center. $\mathbf{R}$ is the rotation matrix from the inertial reference frame (x^e, y^e, z^e) to the body reference frame (x^b, y^b, z^b), $\mathbf{e_3}$ the unit vector in the direction of the z^b-axis, m_a the mass of the quadcopter, g the gravity constant and T_T the total thrust given by

$$T_T = \sum_{i=1}^{4} T_i \tag{15}$$

The control objective is set as follows. Consider the dynamic system given by Eqs. (11)–(13). Design a control law to generate and track an optimal obstacle-free trajectory.

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

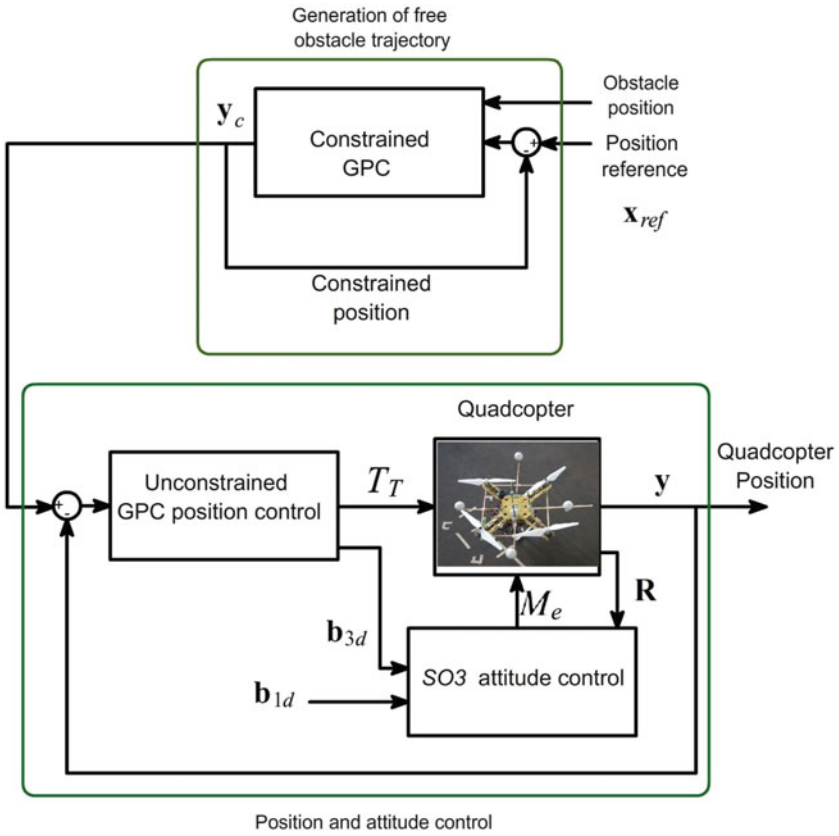


Fig. 2. Control architecture.

4. Control Design

The control strategy proposed in this article uses two control loops (see Fig. 2). The first control loop is responsible for generating an obstacle-free trajectory $\mathbf{y_c}$ to achieve the desired position target $\mathbf{x}_{ref}$. This control loop computes an optimal trajectory over a time horizon before the quadrotor reaches the target position; thus, this control loop requires a sufficient enough sampling period to run the optimization algorithm. It should be noted that the obstacle is static and its position is known. The linear model used for the design of this first control loop is obtained offline from step input tests of the closed-loop quadcopter dynamics under the second control loop.

The second control loop uses the unconstrained GPC technique to follow the reference trajectory generated in the first control loop. The model used for the control design is obtained by considering that the rotational dynamics of the quadcopter is faster than the translational dynamics; thus, if the rotational dynamics reaches their reference values quickly, the translational dynamics has a linear structure according to the analysis presented in ref. [9]. The rotational dynamics controller uses the nonlinear error metrics in the attitude configuration space $so(3)$ as proposed in ref. [9]. The construction of the reference rotation matrix depends on the translational control actions and the desired yaw reference.

4.1. Constrained GPC trajectory generator

The combination of a unconstrained GPC controller and a geometric attitude controller stabilizes the full quadrotor dynamics. To generate the model used to construct the constrained references, the closed-loop system depicted in Fig. 3 is experimentally identified. The input of this model is defined as $\mathbf{u_a} = [\,u_{ax} \quad u_{ay} \quad u_{az}\,]^\top$ and the output as $\mathbf{y_c} = [\,x_c \quad y_c \quad z_c\,]^\top$. $\mathbf{u_a}$ and $\mathbf{y_c}$ are the desired quadrotor position reference and the real-time quadrotor position, respectively, expressed in the x^e, y^e and z^e axes. Figure 4 shows the step responses of the quadrotor translational positions x, y and z. There is a change on the sign for the trajectory along the z-axis with respect to the coordinate frame shown in Fig. 1, to display a graph with positive values. From the step responses, using the peak

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

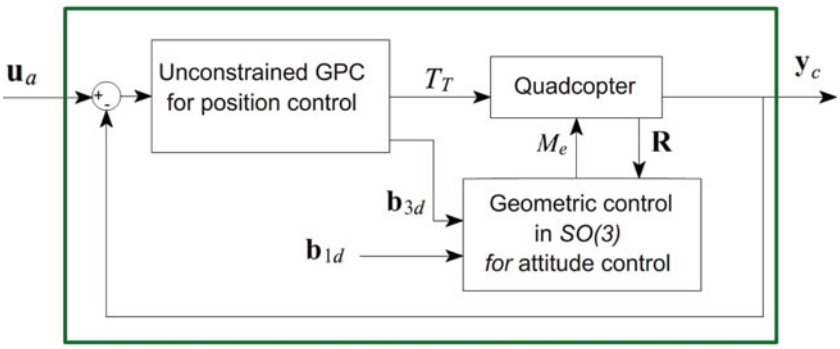


Fig. 3. Scheme to identify a linear model.

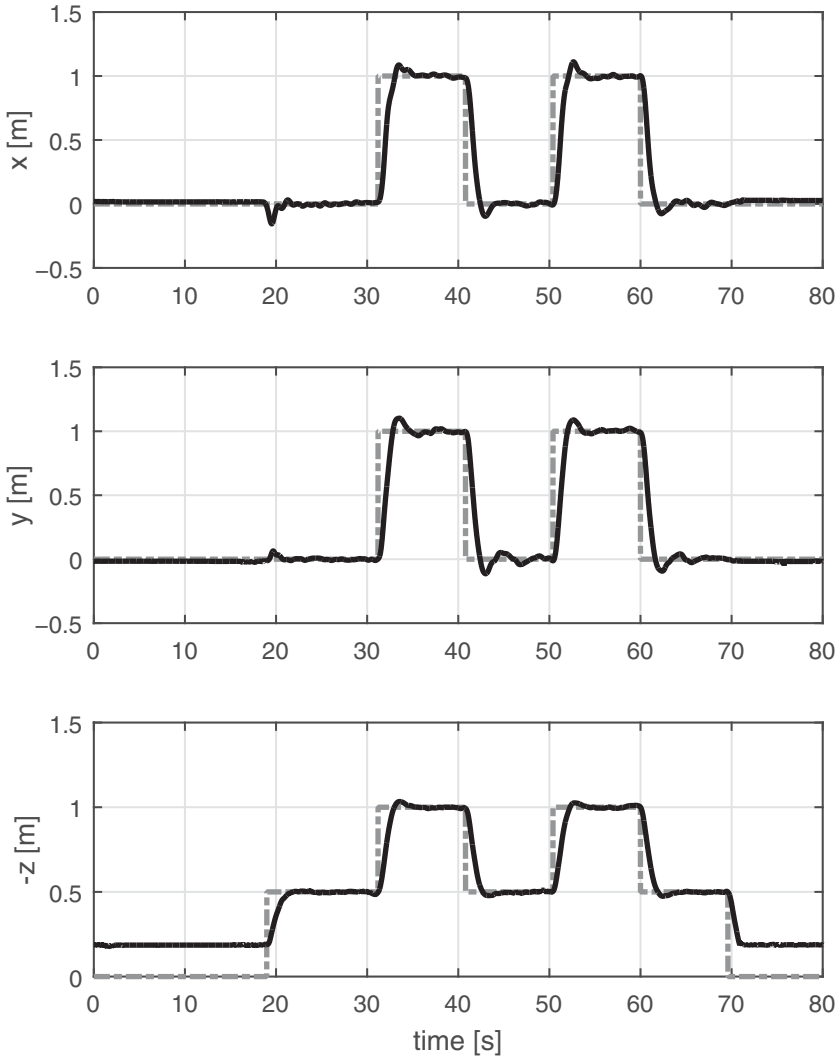


Fig. 4. Quadrotor time responses to the input step. Quadrotor response on the x-axis (top), y-axis (middle) and z-axis (bottom).

time t_p and the maximum overshot M_p, three second-order linear systems capture the closed-loop quadcopter translational dynamics. Table I presents M_p, t_p, the natural frequency ω_n and the relative damping factor ζ for each second-order system. Table II shows the transfer functions for each second-order system. The discretization of the transfer functions employs the zero-order hold method with

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

Table I. Identificated second-order parameters.

	t_p[s]	$\%M_p$	ω_n[rad/s]	ζ
x dynamics	2.27	8.6	1.756 0	0.615 49
y dynamics	2.29	10.2	1.695 8	0.587 83
z dynamics	2.35	6.6	1.767 8	0.654 30

Table II. Transfer functions of second closed loop.

	$F(s) = \frac{\omega_n^2}{s^2+2\zeta\omega_n s+\omega_n^2}$
x dynamics	$\frac{3.083\,5}{s^2+2.161\,6s+3.083\,5}$
y dynamics	$\frac{2.875\,7}{s^2+2.064\,5s+2.875\,7}$
z dynamics	$\frac{3.125\,1}{s^2+2.297\,9s+3.125\,1}$

Table III. Discrete transfer functions of second closed loop.

	$F(z^{-1})$
x dynamics	$\frac{0.03506z^{-1}+0.03124z^{-2}}{1-1.641z^{-1}+0.7076z^{-2}}$
y dynamics	$\frac{0.03287z^{-1}+0.02944z^{-2}}{1-1.656z^{-1}+0.7187z^{-2}}$
z dynamics	$\frac{0.03529z^{-1}+0.03121z^{-2}}{1-1.626z^{-1}+0.6923z^{-2}}$

a sampling period of 0.16 s. The sampling time was chosen smaller than the time constant of the identified linear dynamics and large enough to compute the optimization algorithms.

The trajectory constraints along the x, y and z axes are included as described in Section 2. The prediction of the output signals $x_c(t)$, $y_c(t)$ and $z_c(t)$ is obtained by solving Eqs. (3)–(5) with the discrete transfer functions shown in Table III. The constrained trajectories are generated by minimizing the following quadratic cost function:

$$J_{c1}\left(N_{p_1}, N_{u_1}\right) = \sum_{j=1}^{N_{p1}} \left\|\hat{\mathbf{y}}_{\mathbf{c}}\left(t+j|t\right) - \mathbf{x}_{ref}\left(t+j\right)\right\|^2_{\mathbf{R}_{\mathbf{y_1}}} + \sum_{j=1}^{N_{u_1}} \left\|\Delta\mathbf{u}_{\mathbf{a}}\left(t+j-1\right)\right\|^2_{\mathbf{Q}_{\mathbf{u_1}}} \tag{16}$$

subject to

$$\underline{\mathbf{y}}_{\mathbf{c}} < \mathbf{y}_{\mathbf{c}} < \overline{\mathbf{y}}_{\mathbf{c}} \tag{17}$$

where $\mathbf{x_{ref}} = [\,x_{ref} \quad y_{ref} \quad z_{ref}\,]^\top$ is the reference vector for the position. A quadratic programming algorithm computes the optimal control sequence with the cost function (16) subject to constraints (17).

4.2. MIQP-GPC formulation for obstacle avoidance

This section describes how to integrate obstacle avoidance constraints using the mixed quadratic integer programming method and the GPC strategy. The trajectory to avoid collisions with static obstacles is generated by adding constraints that model obstacles (see ref. [19]). Obstacles can be modeled as convex polygons. For simplicity, a single obstacle is modeled although any number of obstacles can be modeled. The polygon that models the obstacle is established from its vertices defined by x_{min}, x_{max}, y_{min}, y_{max}, z_{min}, z_{max}. The evasion of the modeled obstacle is fulfilled when the position

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

of the quadcopter complies with the constraints

$$\begin{aligned} x_{max} \leq x \leq x_{min} \\ y_{max} \leq y \leq y_{min} \\ z_{max} \leq z \leq z_{min} \end{aligned} \tag{18}$$

which can be expressed as

$$\begin{aligned} x \leq x_{min} \\ -x \leq -x_{max} \\ y \leq y_{min} \\ -y \leq -y_{max} \\ z \leq z_{min} \\ -z \leq -z_{max} \end{aligned} \tag{19}$$

To develop an algorithm that fulfils the constraints in (19), binary slack variables, a sufficiently large value $d > 0$ and a new constraint are added to relax constraints. Then, the set of constraints becomes

$$\begin{aligned} x \leq x_{min} + a_1 d \\ y \leq y_{min} + a_2 d \\ z \leq z_{min} + a_3 d \\ -x \leq -x_{max} + a_4 d \\ -y \leq -y_{max} + a_5 d \\ -z \leq -z_{max} + a_6 d \\ \sum_{i=1}^{6} a_i \leq 5 \text{ with } a_i \in \{0, 1\} \end{aligned} \tag{20}$$

As a result, it is guaranteed that there is always at least one original constraint that is not violated, while the other constraints are relaxed with the value d. The constraints (20) can be expressed in matrix form as

$$\begin{aligned} \mathbf{y_c} \leq \mathbf{y_{min}} + \mathbf{a_a}\mathbf{d_a} \\ -\mathbf{y_c} \leq -\mathbf{y_{max}} + \mathbf{a_b}\mathbf{d_a} \\ \sum_{i=1}^{6} a_i \leq 5 \text{ with } a_i \in \{0, 1\} \end{aligned} \tag{21}$$

where $\mathbf{y_{min}} = [\, x_{min} \;\; y_{min} \;\; z_{min} \,]^\top$, $\mathbf{y_{max}} = [\, x_{max} \;\; y_{max} \;\; z_{max} \,]^\top$, $\mathbf{a_a} = [\, a_1 \;\; a_2 \;\; a_3 \,]^\top$, $\mathbf{a_b} = [\, a_4 \;\; a_5 \;\; a_6 \,]^\top$ and $\mathbf{d_a} = \text{diag}(d)$. Using the GPC approach, the prediction of the output signal is

$$\mathbf{y_c} = \mathbf{G}\mathbf{u_a} + \mathbf{f_a}$$

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

where $\mathbf{Gu_a}$ is the forced response, $\mathbf{f_a}$ is the free response of the prediction output signal. Then, it is possible to write the constraints (21) as

$$\begin{aligned} \mathbf{Gu_a} - \mathbf{a_a d_a} &\leq \mathbf{y}_{\min} - \mathbf{f_a} \\ -\mathbf{Gu_a} - \mathbf{a_b d_a} &\leq -\mathbf{y}_{\max} + \mathbf{f_a} \\ \sum_{i=1}^{6} a_i &\leq 5 \text{ with } a_i \in \{0, 1\} \end{aligned} \tag{22}$$

and the constraints (22) have the form of (9), as follows:

$$\begin{bmatrix} \mathbf{G} & -\mathbf{d_a} & 0 \\ -\mathbf{G} & 0 & -\mathbf{d_a} \\ 0 & \mathbf{b} & \mathbf{b} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{a_a} \\ \mathbf{a_b} \end{bmatrix} \leq \begin{bmatrix} \mathbf{y_{min}} - \mathbf{f_a} \\ -\mathbf{Y_{max}} + \mathbf{f_a} \\ \mathbf{c_a} \end{bmatrix} \tag{23}$$

where the vector $\mathbf{b}$ of order 1×3 and the constant $\mathbf{c_a}$ are chosen depending on the constraint.

4.3. Unconstrained GPC position control

The unconstrained GPC utilizes a translational linear model obtained from cascade interconnection between the translational and rotational quadrotor dynamics. This linear model results from the analysis presented in ref. [9]. For T_T, define

$$\begin{aligned} T_T &= \mathbf{u}^\top \mathbf{b_3} \\ \mathbf{b_{3_d}} &= \frac{\mathbf{u}}{\|\mathbf{u}\|} \end{aligned} \tag{24}$$

with $\mathbf{u} = [\, u_x \quad u_y \quad u_z \,]^\top$ the new control input, $\mathbf{b_3}$ the third column of the rotation matrix $\mathbf{R}$ and $\mathbf{b_{3_d}}$ the desired reference for the third column of the rotation matrix. From (24), the translational dynamics (13) can be expressed as follows:

$$m_a \ddot{\mathbf{x}} = m_a g \mathbf{e_3} - \mathbf{u} - \Phi(\mathbf{u}, \mathbf{b_3}) \tag{25}$$

with

$$\Phi(\mathbf{u}, \mathbf{b_3}) = \frac{\mathbf{u}^\top \mathbf{b_3}}{\mathbf{b_{3_d}}^\top \mathbf{b_3}} \left[\left(\mathbf{b_{3_d}}^\top \mathbf{b_3} \right) \mathbf{b_3} - \mathbf{b_{3_d}} \right] \tag{26}$$

If the attitude controller commands $\mathbf{b_3}$ to converge to $\mathbf{b_{3_d}}$, then $\Phi(\mathbf{u}, \mathbf{b_3})$ converges to zero. Hence, assuming a small enough attitude controller time constant as in ref. [14] or using a high gain attitude controller similar to the high gain controller in ref. [4], it is possible to consider a linear translational quadrotor dynamics. This paper utilizes the geometric attitude controller proposed in ref. [10].

The following differential equation describes the translational quadrotor dynamics for synthesis of the unconstrained GPC controller

$$m_a \ddot{\mathbf{x}} = m_a g \mathbf{e_3} - \mathbf{u} \tag{27}$$

The linear model (27) can be expanded as

$$\begin{aligned} &\dot{x} = v_x \qquad \dot{y} = v_y \qquad \dot{z} = v_z \\ &\dot{v}_x = -\tfrac{1}{m_a} u_x \qquad \dot{v}_y = -\tfrac{1}{m_a} u_y \qquad \dot{v}_z = g - \tfrac{1}{m_a} u_z \end{aligned} \tag{28}$$

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

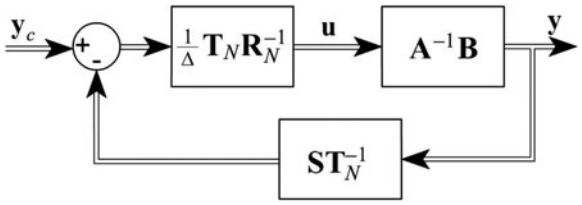


Fig. 5. Closed-loop diagram for stability analysis of the unconstrained Multiple-input multiple-output (MIMO) GPC.

From where the following matrix of transfer functions is obtained:

$$\mathbf{y}(s) = \begin{bmatrix} -\frac{1}{m_a}\frac{1}{s^2} & 0 & 0 \\ 0 & -\frac{1}{m_a}\frac{1}{s^2} & 0 \\ 0 & 0 & -\frac{1}{m_a}\frac{1}{s^2} \end{bmatrix} \mathbf{y}(s) \tag{29}$$

where $\mathbf{y} = [\,x \quad y \quad z\,]^\top$ and $\mathbf{y}(s) = [\,u_x \quad u_y \quad u_z\,]^\top$ are the output and input vectors, respectively. The gravity acceleration constant g takes the role of an unknown perturbation. Then, one obtains

$$\mathbf{y}(z^{-1}) = \begin{bmatrix} \beta\left(z^{-1}\right) & 0 & 0 \\ 0 & \beta\left(z^{-1}\right) & 0 \\ 0 & 0 & \beta\left(z^{-1}\right) \end{bmatrix} \mathbf{u}\left(z^{-1}\right) \tag{30}$$

with $\beta\left(z^{-1}\right) = \frac{-T_s^2}{2m_a}\frac{z^{-1}+z^{-2}}{\left(1-z^{-1}\right)^2}$ and T_s the sampling time.

Now, it is possible to define the polynomial matrices of the CARIMA model (1) as follows:

$$\mathbf{A}\left(z^{-1}\right) = \text{diag}\left[\,1 - 2z^{-1} + z^{-2}, \quad 1 - 2z^{-1} + z^{-2}, \quad 1 - 2z^{-1} + z^{-2}\,\right] \tag{31}$$

$$\mathbf{B}\left(z^{-1}\right) = \text{diag}\left[\,-\frac{T_s^2}{2m_a} - \frac{T_s^2}{2m_a}z^{-1}, \quad -\frac{T_s^2}{2m_a} - \frac{T_s^2}{2m_a}z^{-1}, \quad -\frac{T_s^2}{2m_a} - \frac{T_s^2}{2m_a}z^{-1}\,\right] \tag{32}$$

The control action $\mathbf{u}$ is obtained by minimizing the cost function

$$J_{c2}\left(N_{p_2}, N_{u_2}\right) = \sum_{j=1}^{N_{p_2}} \left\|\hat{\mathbf{y}}\,(t+j|t) - \mathbf{y_c}\,(t+j)\right\|^2_{\mathbf{R_{y_2}}} + \sum_{j=1}^{N_{u_2}} \left\|\Delta\mathbf{u}\,(t+j-1)\right\|^2_{\mathbf{Q_{u_2}}} \tag{33}$$

where $\mathbf{y_c} = [\,x_c \quad y_c \quad z_c\,]^\top$ is the reference vector generated by the constrained GPC controller.

4.3.1. Unconstrained GPC stability. In ref. [7], is shown that in the absence of constrains, the closed-loop dynamics with the Single-input single-output (SISO) GPC controller can be represented in the classical control form in terms of the polynomials **A**, **B**, **T**, $\mathbf{R_N}$ and **S**. Similarly, the reference-output representation for the MIMO case is shown in Fig. 5. The closed-loop relationship from the reference to the output can be written as

$$\mathbf{y}\,(t) = \mathbf{P}_c^{-1}\mathbf{B}\Delta\mathbf{T}_N z^{-1}\mathbf{y}_c\,(t) + \mathbf{P}_c^{-1}\mathbf{T}\mathbf{R}_N\mathbf{e}\,(t) \tag{34}$$

and the characteristic equations for analyzing stability are obtained from the diagonal of the matrix

$$\mathbf{P}_c\left(z^{-1}\right) = \mathbf{R}_N\left(z^{-1}\right)\boldsymbol{\Delta}\mathbf{A}\left(z^{-1}\right) + \mathbf{S}\left(z^{-1}\right)\mathbf{B}\left(z^{-1}\right)z^{-1} \tag{35}$$

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

where the polynomial matrices $\mathbf{R}_N\left(z^{-1}\right)$ and $\mathbf{S}\left(z^{-1}\right)$ are calculated as

$$\mathbf{R}_N\left(z^{-1}\right) = \sum_{i=1}^{N_{p_2}} \mathbf{K_i}^{-1}\left(\mathbf{T}_N\left(z^{-1}\right) + z^{-1}\sum_{i=1}^{N_{p_2}} \mathbf{K_i}\mathbf{I}_i\left(z^{-1}\right)\right) \tag{36}$$

$$\mathbf{S}\left(z^{-1}\right) = \sum_{i=1}^{N_{p_2}} \mathbf{K_i}^{-1}\sum_{i=1}^{N_{p_2}} \mathbf{K_i}\mathbf{F}_i\left(z^{-1}\right) \tag{37}$$

where $\mathbf{T}\left(z^{-1}\right)$ is a 3×3 identity matrix since to $\mathbf{C}\left(z^{-1}\right)$ is the identity matrix. Matrix $\mathbf{I}_i$ is obtained from the matrix $\mathbf{G}_{jp}\left(z^{-1}\right)$ from Eq. (5), $\mathbf{F}_i\left(z^{-1}\right)$ is obtained by solving (4) and $\mathbf{K_i} = \text{diag}(k_{1i}, k_{2i}, k_{3i})$ is the matrix obtained from the first three rows of the gain matrix in Eq. (8).

4.4. Attitude control

This work considers the geometric attitude controller proposed in ref. [9]. As defined in ref. [9], the attitude e_R and angular velocity e_Ω errors are defined as

$$\mathbf{e_R} = \frac{1}{2}\left(\mathbf{R_d}^\top\mathbf{R} - \mathbf{R}^\top\mathbf{R_d}\right)^\vee \tag{38}$$

$$\mathbf{e_\Omega} = \mathbf{\Omega} - \mathbf{R}^\top\mathbf{R_d}\mathbf{\Omega_d} \tag{39}$$

where the map $\overset{\vee}{\cdot} : \mathbf{so}(3) \to \mathbb{R}^3$ is the inverse of the map $\overset{\wedge}{\cdot}$, and $\mathbf{R_d}$ and $\mathbf{\Omega_d}$ are the desired references for the rotation matrix and the angular velocity vector, respectively. Then, the control law is proposed as follows:

$$\mathbf{M_e} = -\mathbf{K_R}\mathbf{e_R} - \mathbf{K_\Omega}\mathbf{e_\Omega} + \mathbf{\Omega}\times\mathbf{J}\mathbf{\Omega} - \mathbf{J}\left(\hat{\mathbf{\Omega}}\mathbf{R}^\top\mathbf{R_d}\mathbf{\Omega_d} - \mathbf{R}^\top\mathbf{R_d}\dot{\mathbf{\Omega}}_\mathbf{d}\right) \tag{40}$$

with $\mathbf{K_R}$, $\mathbf{K_\Omega}$ positive define matrices. The definition of the following vector

$$\mathbf{b_{2_d}} = \frac{\mathbf{b_{3_d}}\times\mathbf{b_{1_d}}}{\left\|\mathbf{b_{3_d}}\times\mathbf{b_{1_d}}\right\|} \tag{41}$$

with $\mathbf{b_{1_d}}$ defining the rotation around the z_e-axis completes the desired rotation matrix as follows:

$$\mathbf{R_d} = \left[\mathbf{b_{2_d}}\times\mathbf{b_{3_d}}, \mathbf{b_{2_d}}, \mathbf{b_{3_d}}\right]$$

The proposed control algorithm can be summarized in the following steps:

1. Compute the prediction signal $\hat{\mathbf{y}}_\mathbf{c}(t+j|t)$ with horizons N_{p_1} and N_{u_1} using the identified model shown in Table III.
2. Minimize the cost function (16) subject to (17) to calculate the control action $\mathbf{u_a}(t)$.
3. Apply control signal $\mathbf{u_a}(t)$ to the first control level. The response $\mathbf{y_c}(t)$ is generated and serves as a reference to the second control level.
4. The second level of control calculates $\hat{\mathbf{y}}(t+j|t)$ with horizons N_{p_2} and N_{u_2} using the model (30).
5. The unconstrained cost function (33) is minimized to obtain the control sequence $\mathbf{u}(t)$.
6. The first component of signal $\mathbf{u}(t)$ is sent to the real quadcopter.
7. From $\mathbf{u}(t)$, (24) is calculated and attitude control is computed from (40).
8. Return to step 1.

5. Experimental Results

Figure 6 shows the infrastructure employed to perform a real-time implementation of the proposed control strategy. The experiments were performed indoors. The Optitrack motion capture system composed of 12 cameras model Flex 13 works at a sampling frequency of 120 Hz. The Optitrack photogrammetry system measures the quadrotor position using markers on the quadcopter. The

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

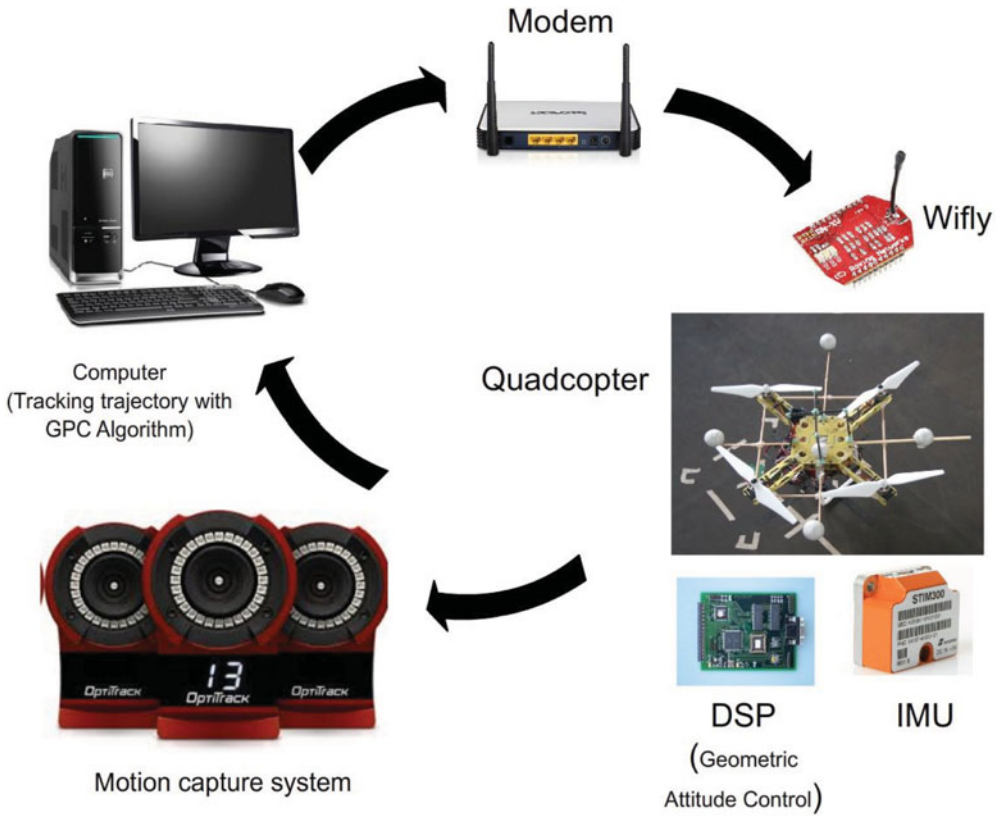


Fig. 6. Experimental device.

quadrotor measured position reaches a personal computer through a USB cable where the two MPC control loops run. MPC algorithms were programmed in C++ language. The personal computer sends, using a modem via the Wi-Fi protocol, the control actions u_x, u_y and u_z to the quadcopter. An AHRS placed onboard the quadcopter measures the attitude and rotational velocity. The translational control actions, the quadrotor attitude, and rotational velocity are fed to the onboard DSP that computes the geometric attitude controller. Finally, the DSP sends the motor control signals to each electronic speed controller (ESC). It was decided to implement the full control scheme in two different processors because the onboard DSP does not have sufficient memory to support the MPC algorithms. However, there are options to compute the full controller in a single device. For instance, using microcomputers that are currently booming as Raspberry Pi, Odroid or Jetson TK1.

The proposed control strategy was first tested in simulation. The simulation served to evaluate different values of N_{p_2}, N_{u_2}, $\mathbf{R}_{\mathbf{y_2}}$ and $\mathbf{Q}_{\mathbf{u_2}}$ to obtain a smooth response with zero steady-state error. One parameter was adjusted at a time, while the others remained constant, first N_{p_2}, then N_{u_2} and at the end $\mathbf{R}_{\mathbf{y_2}}$. The quadrotor translational position had a smooth and stable response by adjusting N_{p_2} values between 60 and 85 and N_{u_2} between 7 and 15. Values of $\mathbf{R}_{\mathbf{y_2}} > \mathbf{Q}_{\mathbf{u_2}}$ showed a response with tracking error equal to zero. The parameters found in the simulation were used as initial adjustment values in the experiments. The numerical values of N_{p_2} and N_{u_2} obtained in simulation showed similar performance during the experiment. Experiments showed that a ratio between $\mathbf{R}_{\mathbf{y_2}}$ and $\mathbf{Q}_{\mathbf{u_2}}$ of 10 to 1 avoids aggressive reactions of the quadcopter and maintains a reduced tracking error.

With $N_{p_2} = 70$, $N_{u_2} = 8$, $\mathbf{R}_{\mathbf{y_2}} = \text{diag}(1, 1, 1)$ and $\mathbf{Q}_{\mathbf{u_2}} = \text{diag}\,(10, 10, 10)$, the polynomial diagonal matrix (35) is obtained:

$$\mathbf{P}_c\,(z) = \text{diag}(-3.5329\text{z}^4 + 10.4629\text{z}^3 - 10.2866\text{z}^2 + 3.3129\text{z} + 0.0437) \tag{42}$$

which provides the characteristic values of the closed-loop translational dynamics with the controller (8).

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

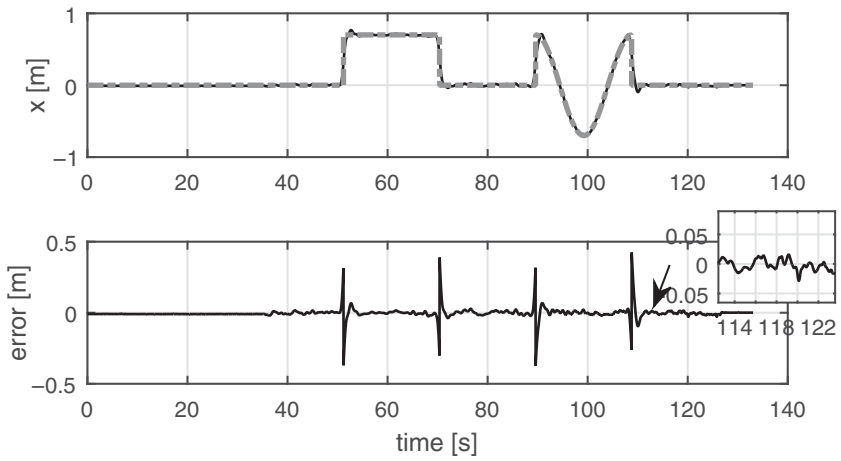


Fig. 7. First experiment. Position tracking along the x-axis (top) and position tracking error at x (bottom).

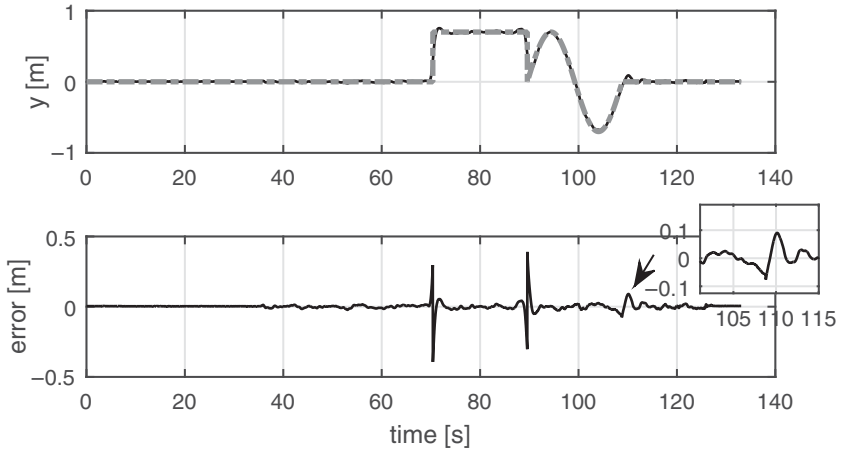


Fig. 8. First experiment. Position tracking along the y-axis (top) and position tracking error at y (bottom).

The magnitudes of the characteristic values obtained when Eq. (42) is solved are

$$\begin{aligned} |z_1| &= |z_2| = 0.9978 \\ |z_3| &= 0.9794 \\ |z_4| &= 0.0127 \end{aligned} \tag{43}$$

then the condition of asymptotic stability is obtained.

Figures 7–9 show the trajectory of the quadcopter along the x, y and z axes with the unconstrained GPC and the geometric attitude controller. The reference signals are displayed as discontinuous lines, while the measured quadrotor positions are shown as continuous lines. The desired references for this experiment are as follows: $x_{ref} = 0$ [m], $y_{ref} = 0$ [m] and $z_{ref} = -0.5$ [m]. This reference takes off the quadrotor. Then, two waypoints are assigned: the first one at $x_{ref} = 0.7$ [m], $y_{ref} = 0$ [m] and $z_{ref} = -0.5$ [m], and the second at $x_{ref} = 0$ [m], $y_{ref} = 0.7$ [m] and $z_{ref} = -0.5$ [m]. Once the quadrotor visits both waypoints a circular reference is assigned. The circular reference is defined as $x_{ref} = 0.7\cos(\omega t)$ [m], $y_{ref} = 0.7\sin(\omega t)$ [m] with $\omega = 3.14$ [rad/s] and constant altitude $z = -0.5$ [m]. Finally, the quadrotor goes back to the initial position $x_{ref} = 0$ [m], $y_{ref} = 0$ [m] and $z_{ref} = -0.05$ [m], where it lands. The position $z(t)$ shows that the quadrotor takes off and lands at 0.2 [m] due to the height of the markers on the quadcopter. It is observed that due to the speed of rotation of the quadcopter around the reference circle, perturbations are induced in the z coordinate; however, the GPC controller oscillates around the reference. Additionally, it is observed that although the force

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

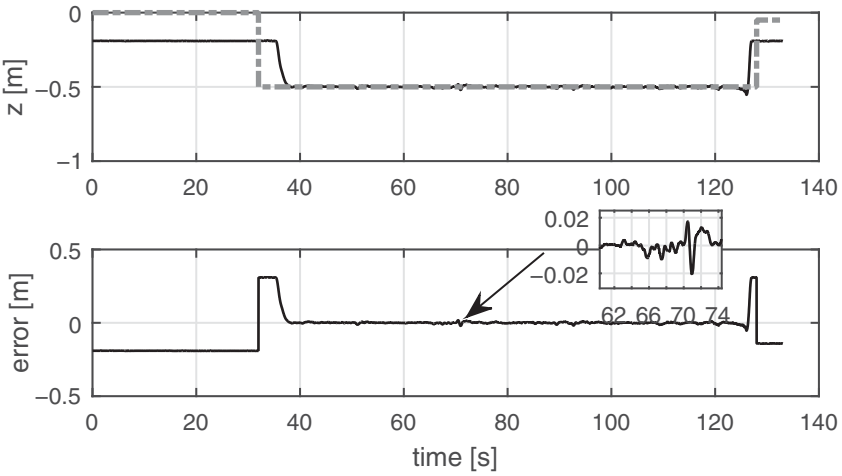


Fig. 9. First experiment. Position tracking along the z-axis (top) and position tracking error at z (bottom).

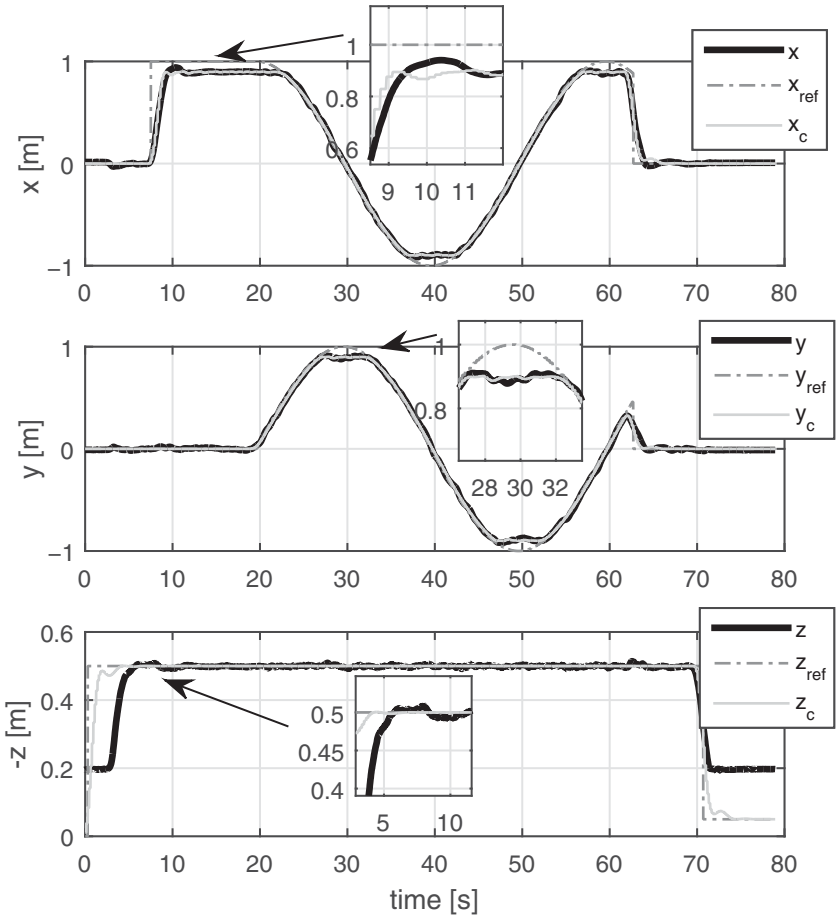


Fig. 10. First experiment. Constrained quadcopter trajectories along the x-axis (top), the y-axis (middle) and the z-axis (bottom).

of gravity is not taken into account during control design, the controller is able to compensate it as a disturbance.

We use the Quadratic programming (QP) algorithm for collision avoidance in a second experiment. This strategy allows the generation of trajectories in interior spaces without collision with the walls. At present, there are very efficient algorithms to deal with the QP optimization problem.

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

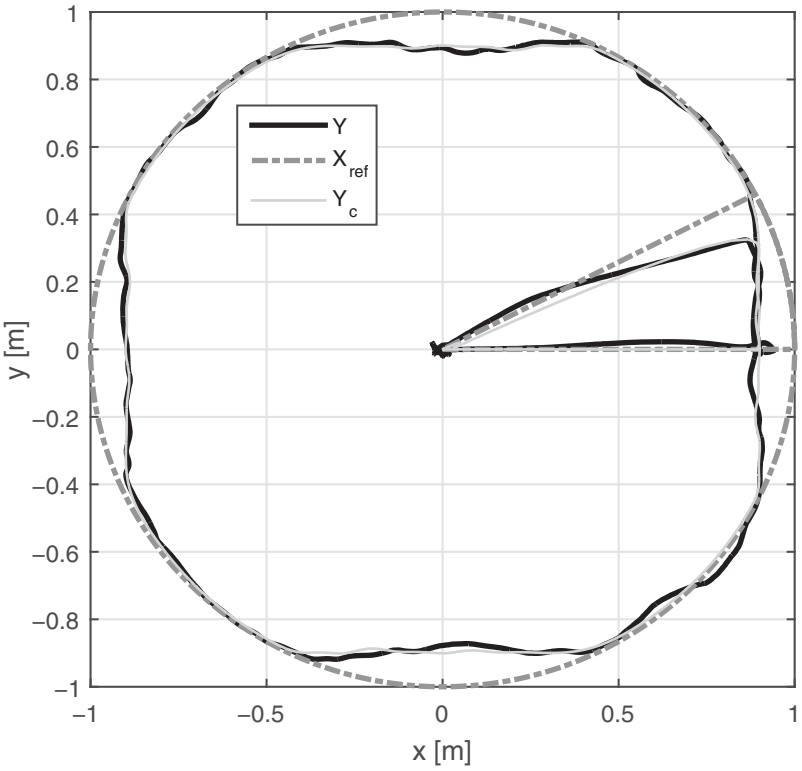


Fig. 11. First experiment. Constrained quadcopter trajectories. A two-dimensional view.

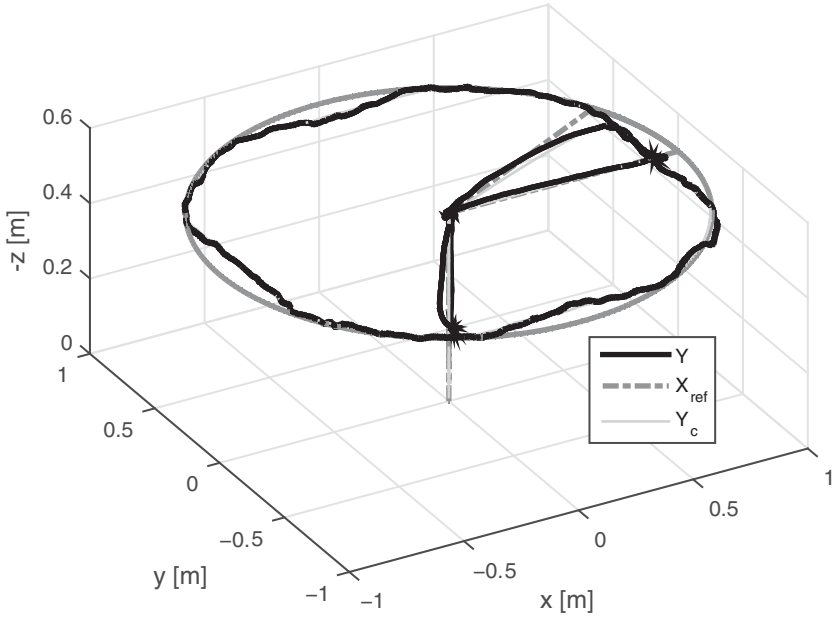


Fig. 12. First experiment. Constrained quadcopter trajectories. A three-dimensional view.

In the second experiment, the controller commands the quadrotor to track a constrained circle. It is not necessary to use the slack parameter d to solve the optimization problem subject to the resulting constraints. The quadratic programming method implemented in C++ code, obtained from ref. [4], is employed, under a GNU (GNU is Not Unix) General Public License. Figure 10 presents the quadrotor trajectory under the action of the two MPC control loops and the geometric attitude controller. The desired trajectory has constraints on the cartesian plane. In this experiment, the constraints were

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

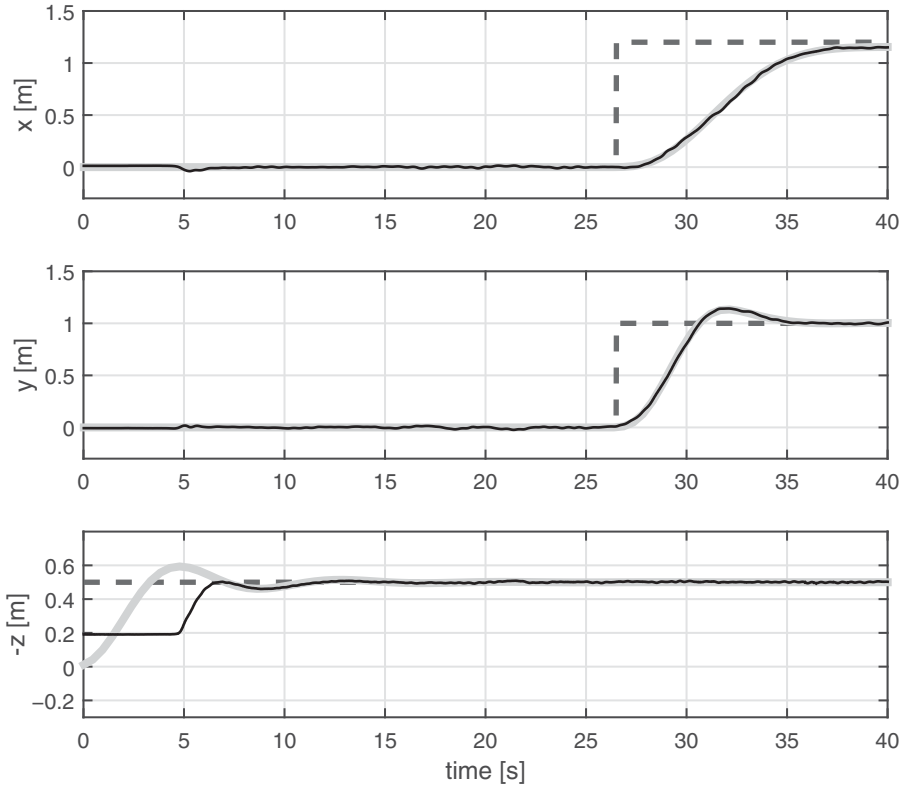


Fig. 13. Second experiment. Constrained quadcopter trajectories along the x-axis (top), the y-axis (middle) and the z-axis (bottom).

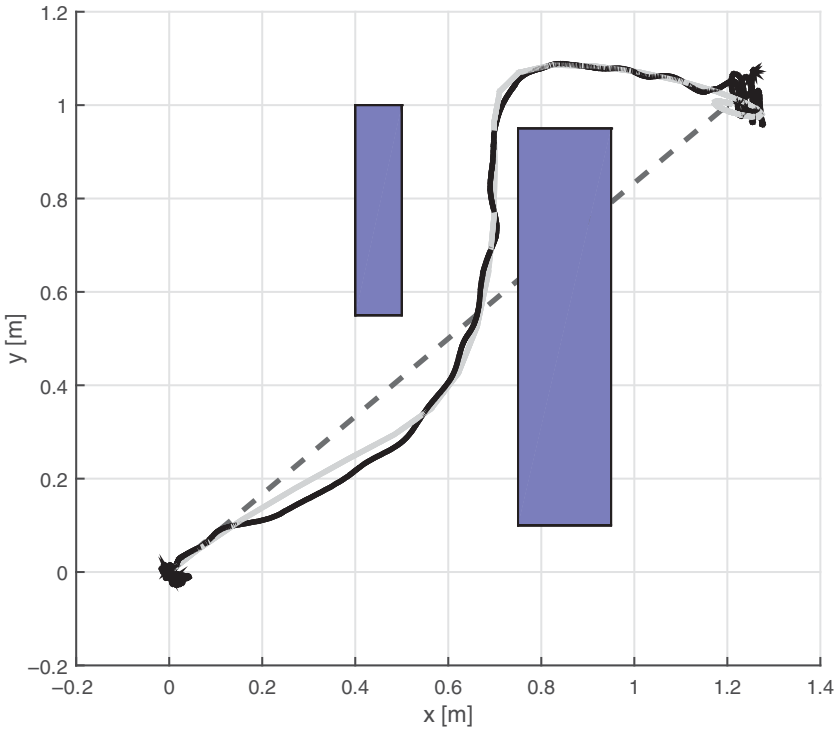


Fig. 14. Second experiment. Constrained quadcopter trajectories. A two-dimensional view.

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

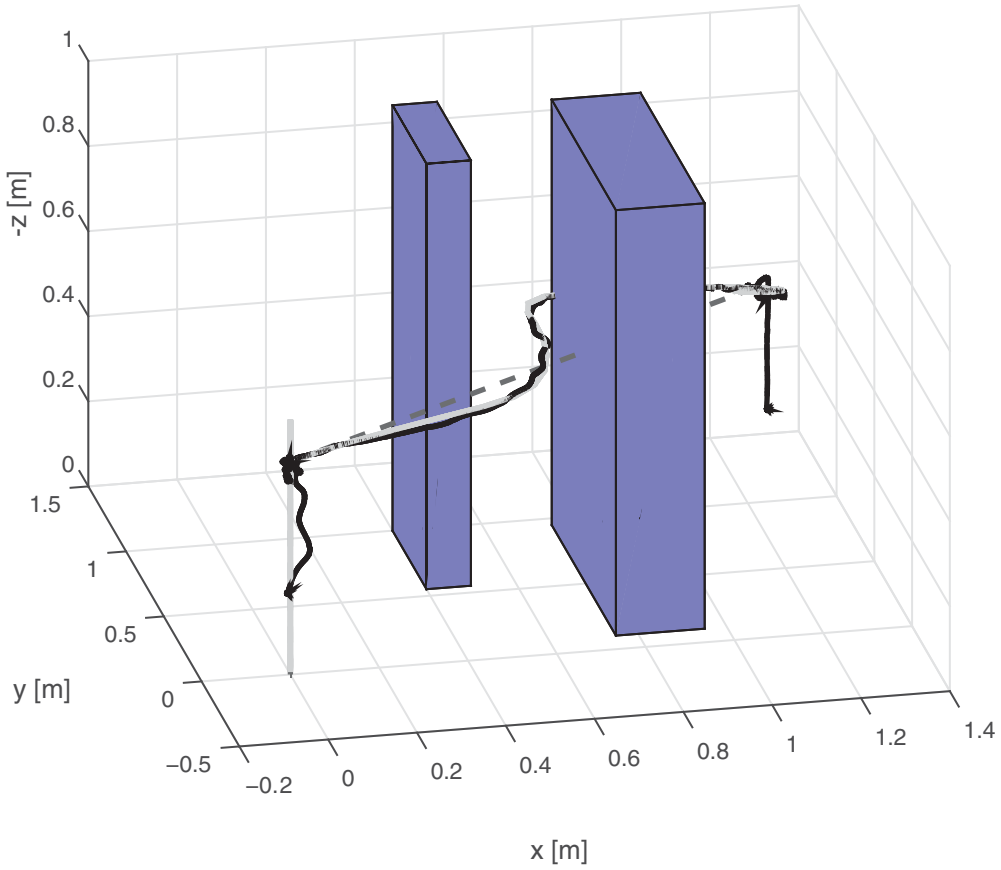


Fig. 15. Second experiment. Constrained quadcopter trajectories. A three-dimensional view.

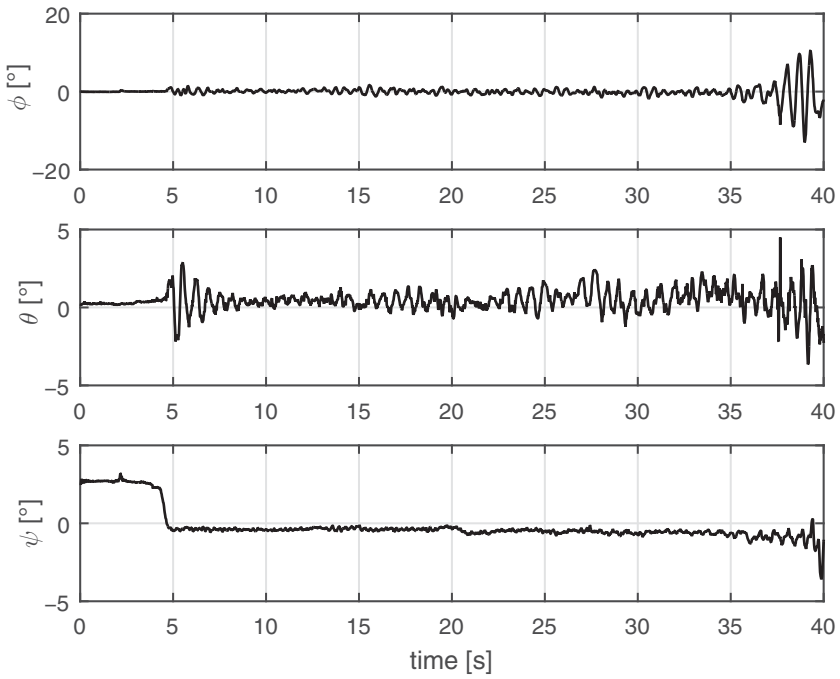


Fig. 16. Second experiment. Quadrotor attitude parameterized by Euler angles.

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

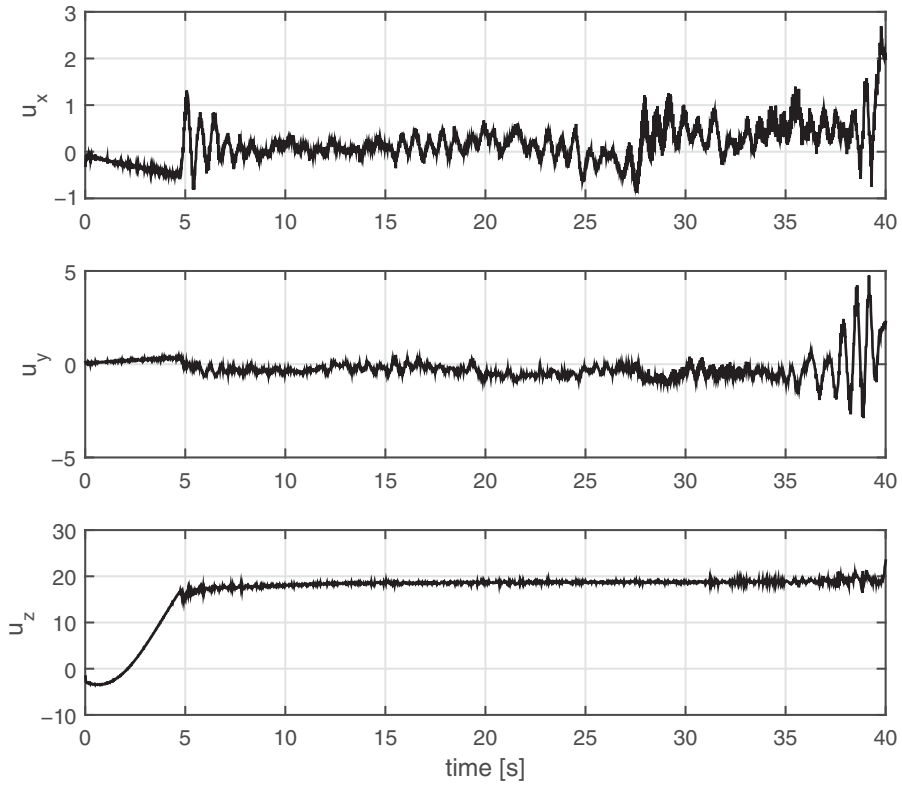


Fig. 17. Second experiment. Quadcopter control signals.

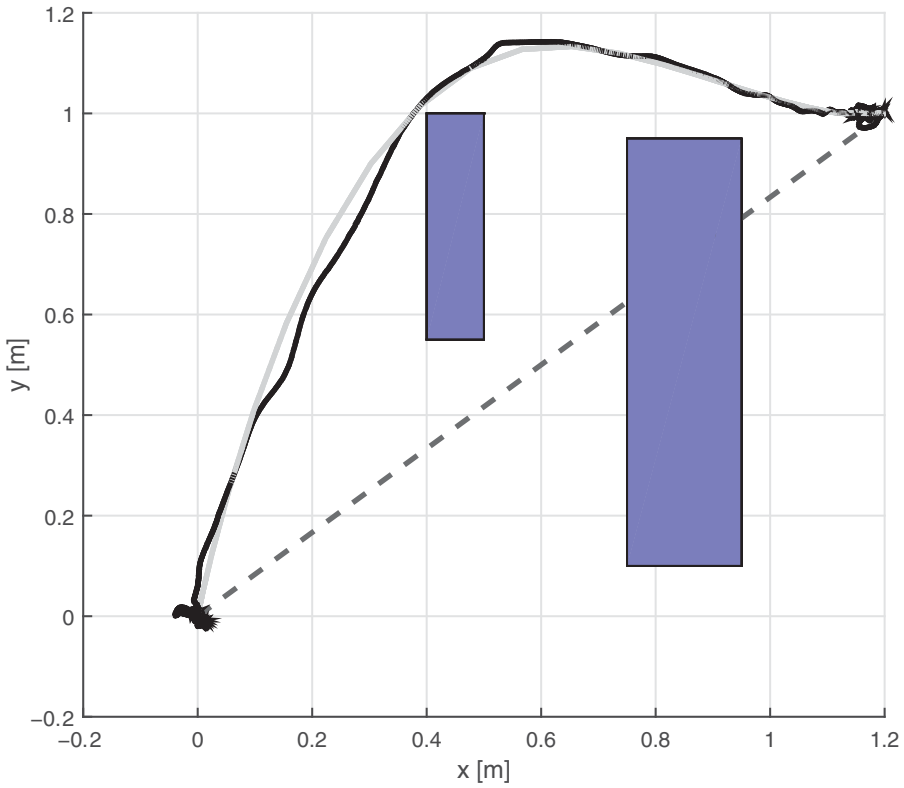


Fig. 18. Third experiment. Quadcopter constrained trajectory with $\mathbf{R_{y1}} = \text{diag}(100, 1000, 0)$. A two-dimensional view.

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

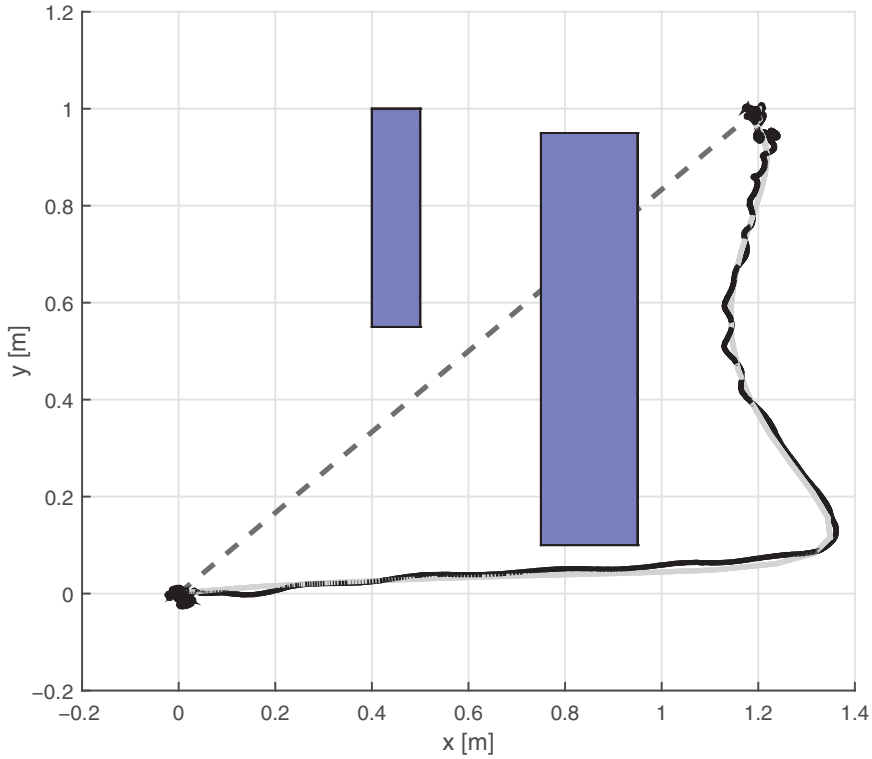


Fig. 19. Third experiment. Quadcopter constrained trajectory with $\mathbf{R_{y1}} = \text{diag}(1000, 100, 0)$. A two-dimensional view.

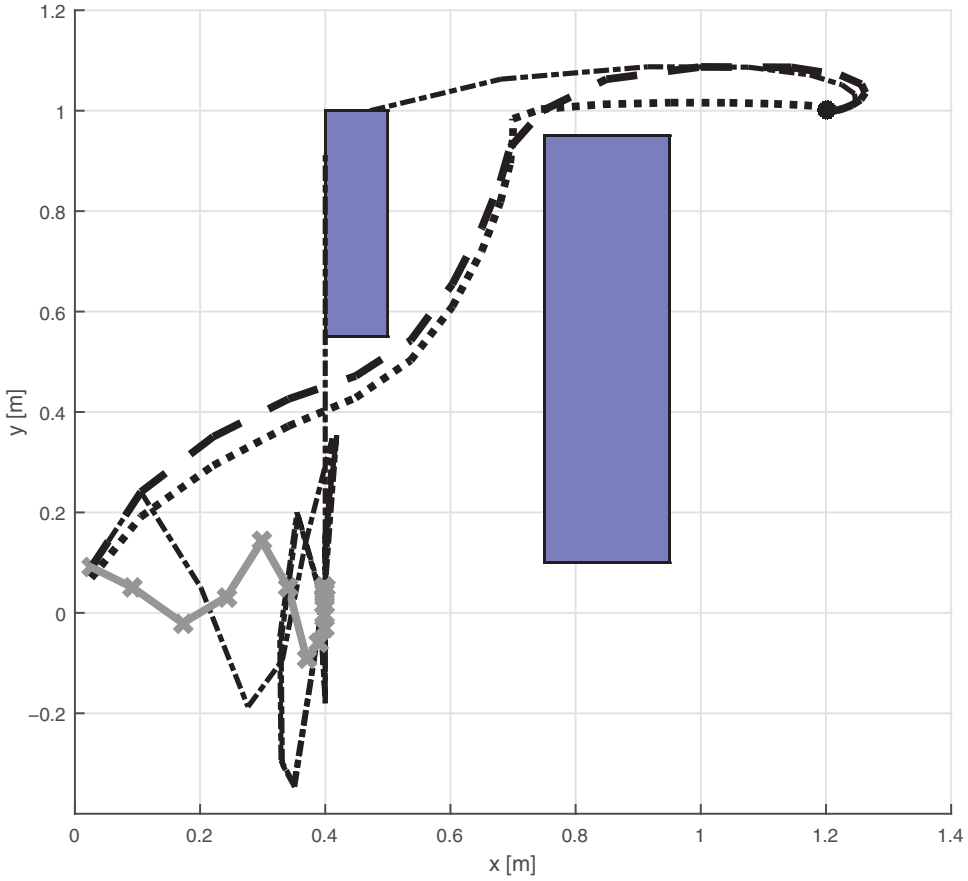


Fig. 20. Quadcopter constrained trajectory in the *xy* plane.

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

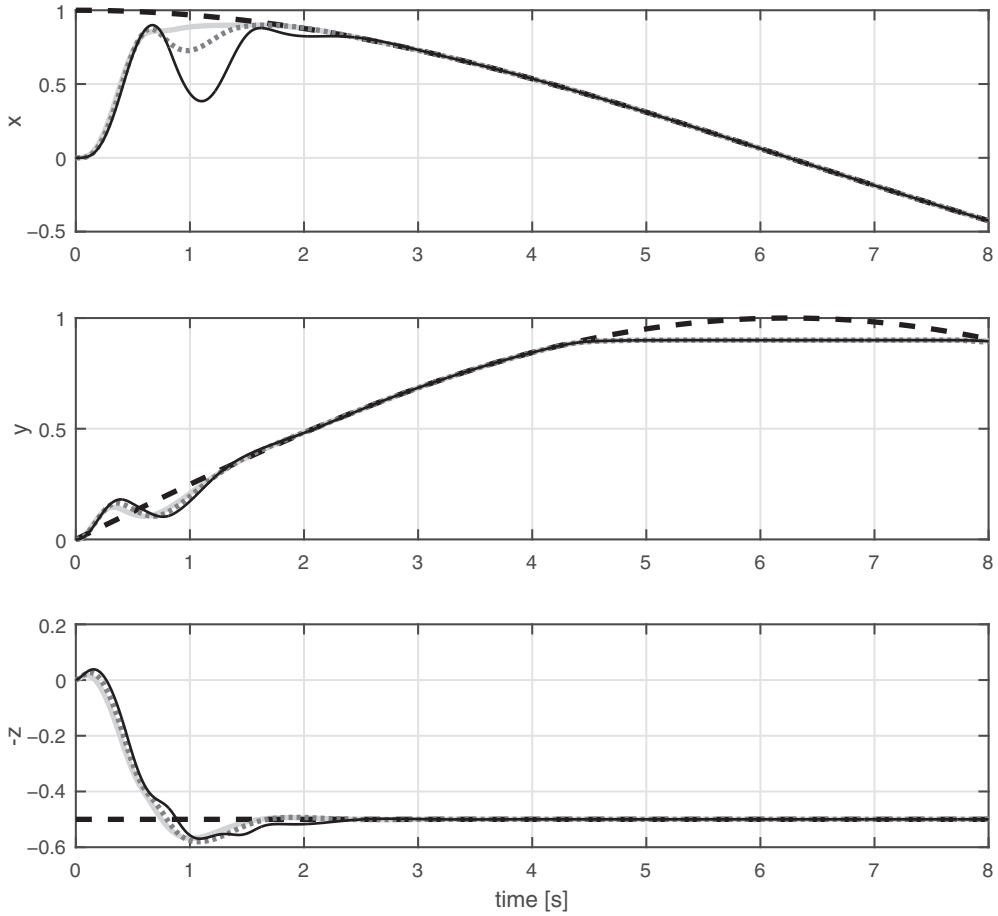


Fig. 21. Trajectory along the x-axis (top), y-axis (middle) and z-axis (bottom) with variations in mass.

defined as $-0.9 < x < 0.9$ and $-0.9 < y < 0.9$. It can be observed that the quadrotor tracks the desired reference respecting the imposed constraints and returns to the reference trajectory. The graphs in Figs. 11 and 12 show the displacement of the quadcopter on the Cartesian plane and the three-dimensional trajectory, respectively.

In the third experiment, the MIQP algorithm solves the optimization problem. The MIQP is based on the branch and bound approach. The IBM Cplex libraries[1] were used in Matlab and C++ language to perform simulations and experiments, respectively. In this experiment two obstacles were considered. The obstacles are modeled with vertices in the xy plane defined as $(0.4, 0.5)$, $(0.4, 1)$, $(0.5, 1)$, $(0.5, 0.5)$ for the first obstacle, and $(0.7, 0.1)$, $(0.7, 1)$, $(1, 1)$, $(1, 0.1)$ for the second one. Both obstacles have a height of 1 m; thus, the $\mathbf{y_{min}}$ and $\mathbf{y_{max}}$ vectors are constructed. In addition, $\mathbf{b} = [\,1 \quad 1 \quad 0\,]$ and $\mathbf{c_a} = 3$. These constraints define two rectangular columns as is shown in Fig. 15.

In Figs. 13–15, the displacements of the quadcopter on the x, y and z axes are shown with $\mathbf{R_{y1}} = \text{diag}(1000, 1000, 0)$, $\mathbf{Q_{u1}} = \text{diag}(10, 10, 0)$ and $d = 10$. The reference for position trajectories in the absence of obstacles is shown in a dashed line. The gray signals show the trajectory generated by the GPC-MIQP algorithm and black lines show the measurement of the real-time trajectory of the quadcopter. Figures 16 and 17 show the evolution of the Euler angles and the control signals, respectively, obtained in this experiment.

The modeling of obstacles as polygons causes the optimization algorithm QP not to be able to generate a trajectory that surrounds an obstacle to reach a position beyond the modeled polygon. To overcome this disadvantage, we use a strategy based on the MIQP algorithm, which relaxes constraints and allows the generation of a trajectory that avoids collisions with obstacles.

The third experiment considers a change in the error weight matrix $\mathbf{R_{y1}}$. Figure 18 shows the quadcopter trajectory considering $\mathbf{R_{y1}} = \text{diag}(100, 1000, 0)$. Figure 19 shows the quadcopter trajectory with $\mathbf{R_{y1}} = \text{diag}(1000, 100, 0)$. Note that when the error weight on the y coordinate is

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

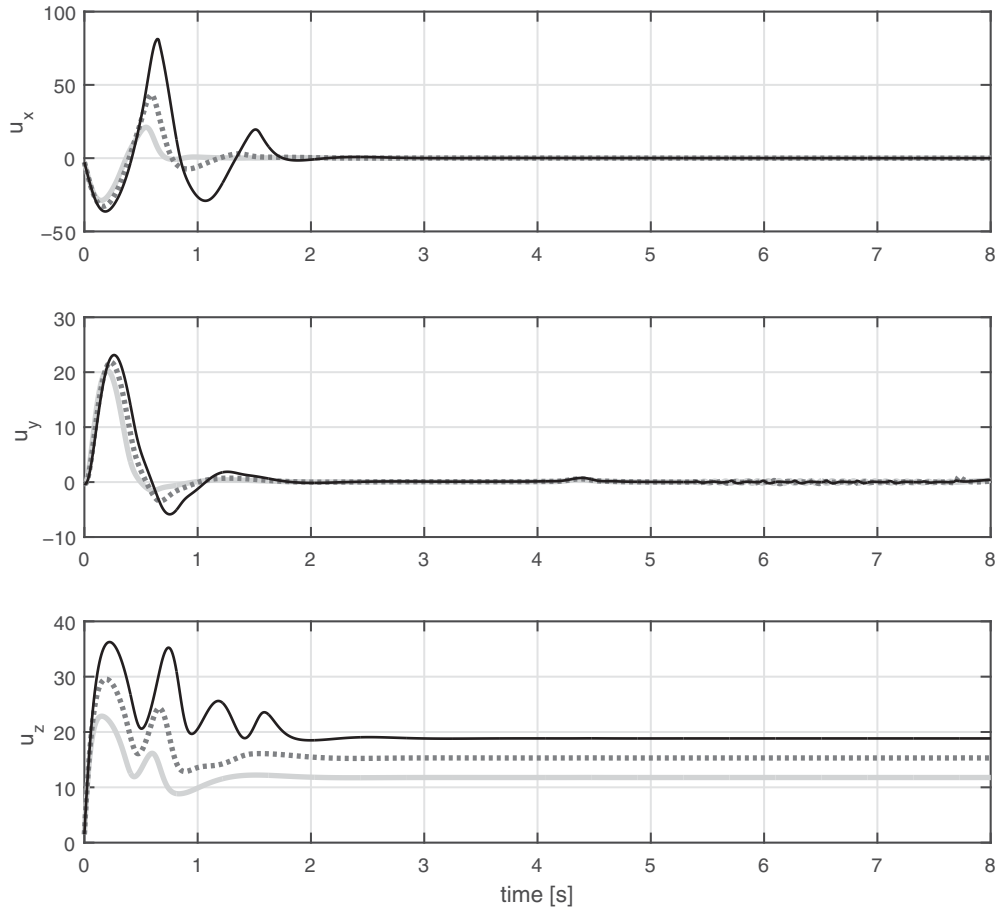


Fig. 22. Behavior of control signals with variations in mass.

greater than the error weight on the x coordinate the trajectory generator first reduces the error in the y coordinate. The opposite occurs when the error weight in the x coordinate is higher than the error weight in the y coordinate. The effect of the error weight matrix can be observed in Fig. 18.

To verify the behavior of the trajectory generator when d is modified in the constraints a simulation was carried out. The trajectories generated for different values of d are shown in Fig. 20, where values of $d = 1$ (dashed line), $d = 1000$ (dotted line), $d = 2000$ (dash-dot line) and $d = 2500$ (solid line in gray) were used. For very large values of d, the MIQP algorithm showed faults in the expected behavior. For d values close to zero, the algorithm has no feasible solutions because all the constraints cannot be met.

Figures 21 and 22 show a simulation of the second level of control, which stabilizes the quadcopter. These simulations show the behavior of the position tracking (Fig. 21) and the control signals (Fig. 22) when the parameter of mass m_a change in the quadcopter model (11)–(13) with respect to the mass in the nominal model (28) used to obtain the position prediction for the controller.

In Figs. 21 and 22, the reference signal is shown as a black dashed line and the position tracking with $m_a = 1.2$ kg, which is considered the nominal value of the mass is shown in gray solid line. The simulations for $m_a = 1.56$ kg and $m_a = 1.92$ kg are shown in dotted line and black continuous line, respectively. The simulated behavior obtained is acceptable up to an increase of 60% in the mass parameter. When m_a increases beyond 60%, the simulations show unstable behavior.

6. Conclusion

In this work, a cascade control approach for a multirotor vehicle that solves the constrained trajectory tracking problem has been proposed. The proposed strategy generates trajectories using the GPC

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press

algorithm together with the MIQP method to manage constraints on the trajectory. The experimental results show that the control objective is reached in real time.

Acknowledgment
This work was carried out under the postdoctoral fellowship to Dr. José Luis Mendoza-Soto from CONACyT and partially funded by CONACyT-AEM under grant 262794.

References

1. IBM Corp. 1987, 2015. *IBM ILOG CPLEX Optimization Studio Getting Started with CPLEX, Version 12 Release 6*. IBM, USA (2015).
2. M. Abdolhosseini, Y. M. Zhang and C. A. Rabbath, "An efficient model predictive control scheme for an unmanned quadrotor helicopter," *J. Intell. Robot. Syst.* **70**(1–4), 27–38 (2013).
3. K. Alexis, C. Papachristos, R. Siegwart and A. Tzes, "Robust model predictive flight control of unmanned rotorcrafts," *J. Intell. Robot. Syst.* **81**(3–4), 443–469 (2016).
4. M. Bangura and R. Mahony, "Real-time model predictive control for quadrotors," *IFAC Proc. Vol.,* **47**(3), 11773–11780 (2014) 19th IFAC World Congress.
5. A. Bemporad, C. A. Pascucci and C. Rocchi, "Hierarchical and hybrid model predictive control of quadcopter air vehicles," *IFAC Proc. Vol.* **42**(17), 14–19 (2009) 3rd IFAC Conference on Analysis and Design of Hybrid Systems.
6. A. Bemporad and C. Rocchi, "Decentralized Linear Time-Varying Model Predictive Control of a Formation of Unmanned Aerial Vehicles," *Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference* (Dec. 2011) pp. 7488–7493.
7. E. F. Camacho and C. Bordons, *Model Predictive Control* (Springer-Verlag, Great Britain, 2004).
8. T. Engelhardt, T. Konrad, B. Schafer and D. Abel, "Flatness-Based Control for a Quadrotor Camera Helicopter Using Model Predictive Control Trajectory Generation," *Proceedings of the 24th Mediterranean Conference on Control and Automation (MED)*, Athens, Greece (Jun. 21–24, 2016) pp. 852–859.
9. T. Lee, M. Leok and N. H. McClamroch, "Geometric Tracking Control of a Quadrotor UAV on SE(3)," *Proceedings of the 49th IEEE Conference on Decision and Control (CDC)* (Dec. 2010) pp. 5420–5425.
10. T. Lee, M. Leok and N. H. McClamroch, "Nonlinear robust tracking control of a quadrotor UAV on SE(3)," *Asian J. Control* **15**(2), 391–408 (2013).
11. C. Liu, S. Tang, S. Yang and Y. Li, "Generalized Predictive Control with Dynamic Compensation for Quadrotor Attitude Stabilization," *Proceedings of the 33rd Chinese Control Conference* (Jul. 2014) pp. 7709–7714.
12. J. M. Maciejowski, *Predictive Control with Constraints* (Pearson Education Limited, Prentice Hall, London, 2002).
13. D. Mellinger, A. Kushleyev and V. Kumar, "Mixed-Integer Quadratic Program Trajectory Generation for Heterogeneous Quadrotor Teams," *Proceedings of the IEEE International Conference on Robotics and Automation* (May 2012) pp. 477–483.
14. M. W. Mueller and R. D'Andrea, "A Model Predictive Controller for Quadrocopter State Interception," *Proceedings of the European Control Conference (ECC)* (Jul. 2013) pp. 1383–1389.
15. M. Neunert, C. de Crousaz, F. Furrer, M. Kamel, F. Farshidian, R. Siegwart and J. Buchli, "Fast Nonlinear Model Predictive Control for Unified Trajectory Optimization and Tracking," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (May 2016) pp. 1398–1404.
16. A. Richards, T. Schouwenaars, J. P. How and E. Feron, "Spacecraft trajectory planning with avoidance constraints using mixed-integer linear programming," *J. Guid., Control, Dyn.* **25**, 755–764 (2002).
17. Y. Rochefort, S. Bertrand, H. Piet-Lahanier, D. Beauvois and D. Dumur, "Cooperative nonlinear model predictive control for flocks of vehicles," *IFAC Proc. Vol.* **45**(1), 169–174 (2012) 1st IFAC Workshop on Embedded Guidance, Navigation and Control in Aerospace.
18. T. Schouwenaars, E. Feron and J. How, "Multi-Vehicle Path Planning for Non-Line of Sight Communication," *Proceedings of the American Control Conference* (Jun. 2006) p. 6.
19. T. Schouwenaars, B. De Moor, E. Feron and J. How, "Mixed Integer Programming for Multi-Vehicle Path Planning," *Proceedings of the European Control Conference (ECC)* (Sep. 2001) pp. 2603–2608.
20. Y. E. Tlatelpa-Osorio, J. J. Corona-Sánchez and H. Rodríguez-Cortés, "Quadrotor Control Based on an Estimator of External Forces and Moments," *Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS)*, Arlington, VA, USA (Jun. 7–10, 2016) pp. 957–963.
21. P. Vlantis, P. Marantos, C. P. Bechlioulis and K. J. Kyriakopoulos, "Quadrotor Landing on an Inclined Platform of a Moving Ground Vehicle," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (May 2015) pp. 2202–2207.
22. A. Zulu and S. John, "A review of control algorithms for autonomous quadrotors," *Open J. Appl. Sci.* **4**, 547–556 (2014).

https://doi.org/10.1017/S026357471800036X Published online by Cambridge University Press