Model predictive control strategies for parcel relay manoeuvres using drones

Francisco Matos[1] and Bruno Guerreiro[1,2]
[1]DEEC and CTS/UNINOVA, NOVA School of Science and Technology, 2829-516 Caparica, Portugal.
[2]Institute For Systems and Robotics, LARSyS, 1049-001, Lisbon, Portugal.
E-mails: fr.matos@campus.fct.unl.pt and bj.guerreiro@fct.unl.pt

***Abstract*—The increasing supply and demand by customers and companies is an important factor present in today's economy. With new business tactics like online shopping, the number of products being requested by clients is rapidly increasing, and to have a profitable business companies must be able to reach the client in a quick, effective way. Current delivery services distribute large quantities of items in urban environments, however, for individual deliveries, they are not the most efficient, since they result in high costs. Autonomous electric vehicles like quadcopters offer a new approach to the delivery paradigm, being more efficient and accessible, less polluting, among other advantages, however their low battery life presents a big disadvantage, limiting the range these aircrafts can reach to deliver packages. The present paper addresses this problem by studying advanced control methods, particularly Model Predictive Control (MPC) based techniques due to their effectiveness handling agile dynamics and performing aggressive manoeuvres, accounting to physical and mechanical constraints present in the process. The main goal is having two aircrafts perform a relay manoeuvre, exchanging a payload between them and effectively extending the delivery range. Firstly, the vehicle's model is established and used to formulate the optimal control problem. The resulting controller will be subjected to validation using complex trajectories and compared to an implementation of another similar controller. After validation, the controller will be adapted to support two vehicles simultaneously and subjected to a parcel exchange scenario. Conclusions about the feasibility of the controller will be drawn.**

***Index Terms*—Aggressive Manoeuvre, Model Predictive Control, Parcel Delivery**

I. Introduction

Parcel delivery is an important part of the global economy. It is essential for a business that a package is delivered to the client who ordered it in a fast and efficient way. A good delivery service can reduce costs and increase the company's efficiency and profit. Currently, the delivery services, like post mails services, are a good way to deliver products in an urban environment, however they have some problems, mainly high costs, the usage of less efficient paths due to traffic or poor planning, and pollution issues. The search for new, more efficient approaches to delivery services can be benefic for both the clients and the companies.

Quadcopters, or drones, are a technology whose study has been increasing in popularity, due to their versatile features, like great manoeuvrability, stability and hovering abilities. The idea of using quadcopters for parcel delivery offers a lot of advantages, such as quick and safe delivery in virtually any place due to their compact format, efficient option due to studies of optimal routing using GPS technology, allow for better time management, it is an environmentally friendly technology, and many others.

However, one of the challenges of using quadcopters is their short autonomy, which is dependent on their size and battery capacity. In order for a payload to travel large distances using a UAV, it is important to consider these limitations when planning the delivery process. One way of extending the travel range is by having multiple drones exchange the package between them as the previous one is about to run out of power.

In order to perform the payload exchange, control strategies that allow agile manoeuvres must be used to assure it is performed safely and successfully. Techniques using Nonlinear Model Predictive Control (NMPC) are studied as they effectively handle agile dynamics affected by physical and mechanical constraints and will allow for the development of a strategy capable of exchanging a parcel between two quadcopters. In [1], an NMPC with a Particle Swarm Optimization algorithm to solve the optimization problem is used to control the velocity in the three axes of an UAV and achieve good trajectory tracking under disturbance influence. In [2], it was proposed the development of a framework for attitude control of Micro Aerial Vehicles (MAVs), using fast NMPC, capable of exploiting the full flight envelope, and presenting superior disturbance rejection characteristics and actuator redundancy. In [3], an MAV is controlled through a set of precise aggressive manoeuvres using three controllers (attitude, hover and 3D path following) and switching between them in an intelligent way, allowing more complex tasks to be executed, such as flying through gaps in different angles and orientations, and perching on a target at different angles.

REPLACE [4] is a project, being currently developed, whose main goal is to integrate quadrotors in delivery systems, which aims to develop fast package delivery in urban environment, considering the autonomy limitations, that affect travel distance, and using drone relays to effectively deliver a package to a destination beyond a single drone's flight range. The main idea is to have two drones execute synchronized, strategical manoeuvres to safely exchange a parcel between

[*] This work was partially funded by FCT project REPLACE (PTDC/EEI-AUT/32107/2017) which includes Lisboa 2020 and PIDDAC funds, project CAPTURE (PTDC/EEI-AUT/1732/2020), and also projects CTS (UIDB/00066/2020) and LARSYS (UIDB/50009/2020).

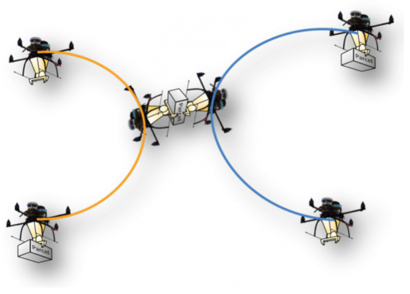

Fig. 1. Project REPLACE [4]

them, effectively extending its travel distance, as illustrated in Figure 1.

The remainder of this paper is organized as follows: in Section II the model of the considered multicopter is presented followed by the description of the NMPC strategies to control both an independent drone and two vehicles simultaneously in Sections III and IV, respectively. In Section V the simulation of the independent and joint controllers is presented, with some result analysis. Finally, conclusions are summarized in Section VI.

II. Quadcopter Model

The multirotor considered is a quadcopter, modeled as a 6 degrees-of-freedom (DoFs) rigid body. For the formulation of the NMPC controller, the chosen quadcopter model is

$$\begin{aligned} \dot{\mathbf{p}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= -g\mathbf{z_W} + \frac{u_1}{m}{}^{\mathbf{W}}\mathbf{R_B}\mathbf{z_W} \\ {}^{\mathbf{W}}\dot{\mathbf{R}}_{\mathbf{B}} &= {}^{\mathbf{W}}\mathbf{R_B}\hat{\boldsymbol{\omega}}_{\boldsymbol{B}} \end{aligned} \tag{1}$$

with the inputs being the total thrust u_1 and the angular velocity vector of the vehicle $\boldsymbol{\omega_B}$, and where $\mathbf{p}$ and $\mathbf{v}$ are the position and velocity of the center of mass, respectively, ${}^{\mathbf{W}}\mathbf{R_B}$ is the rotation matrix of the body with respect to the inertial frame, m is the total mass, g is the acceleration of gravity, $\mathbf{z_W}$ is the world z axis defined as $\mathbf{z_W} = [\,0\;0\;1\,]^T$ and $\hat{\boldsymbol{\omega}}_{\boldsymbol{B}}$ is the skew symmetric matrix calculated from $\boldsymbol{\omega_B}$.

The state vector for a given drone i is defined as

$$\mathbf{x_i} = \left[\mathbf{p_i^T}, \mathbf{v_i^T}, \mathbf{x_{iB}^T}, \mathbf{y_{iB}^T}, \mathbf{z_{iB}^T}\right]^{\mathbf{T}} \tag{2}$$

while its control input is defined as

$$\mathbf{u_i} = \left[u_{1_i}, \omega_{i_x}, \omega_{i_y}, \omega_{i_z}\right]^T. \tag{3}$$

This method assumes that the dynamics of control of the velocity of the propellers based on desired angular velocities are handled by an inner loop of the autopilot.

For a given drone i, the system dynamics are represented as

$$\dot{\mathbf{x}}_{\mathbf{i}} = f\left(\mathbf{x_i}, \mathbf{u_i}\right). \tag{4}$$

III. Independent Drone NMPC Formulation

In this section, it is presented the formulation of the optimal control problem, as well as the constraints subjected and the respective cost functional, for a single vehicle implementation.

A. Problem Formulation

For the independent controller, the chosen state and control input vectors are defined as equations (2) and (3), respectively, with i = 1. To have the penalties that the NMPC will attempt to minimize, it is necessary to first define the respective errors that will be weighted. The reference tracking errors for position and velocity are defined as

$$\mathbf{e_p} = \mathbf{p} - \mathbf{p_{ref}} \tag{5}$$

$$\mathbf{e_v} = \mathbf{v} - \mathbf{v_{ref}} \tag{6}$$

while the error between the desired and actual rotation is given by

$$\mathbf{e_R} = \frac{1}{2}\left({}^{\mathbf{W}}\mathbf{R}_{\mathbf{B_{des}}}^{\mathbf{T}}\,{}^{\mathbf{W}}\mathbf{R_B} - {}^{\mathbf{W}}\mathbf{R_B^T}\,{}^{\mathbf{W}}\mathbf{R}_{\mathbf{B_{des}}}\right)^{\vee} \tag{7}$$

with $\vee$ being the inverse of the skew matrix.

To achieve a stable steady state, it is desired that the angular velocities tend to 0 and the thrust generated by the propellers is equal to the force of gravity to achieve hover. The angular velocity error is given by

$$\mathbf{e}_{\boldsymbol{\omega}} = \boldsymbol{\omega_B} \tag{8}$$

while the thrust error is given by

$$e_U = u_{1_1} - u_{ss} \tag{9}$$

with $u_{ss} = mg$.

B. Constraints

One property of NMPC is the ability to include system's limitations in the optimal control problem when calculating the control input, therefore, the controller uses equation (4) as a constraint.

The first obvious constraint is that the drone's z position must be always non-negative, since the height is always measured referencing the ground, expressed as

$$p_z \geq 0. \tag{10}$$

The propellers of the quadcopter can provide a limited amount of thrust, and so the controller needs to take into account the actuator's saturation, given by

$$0 \leq u_{1_1} \leq u_{sat} \tag{11}$$

while the angular velocity, being also a function of the moments created by the propellers, is also limited, which can defined as

$$-\boldsymbol{\omega}_{max} \leq \boldsymbol{\omega_B} \leq \boldsymbol{\omega}_{max} \tag{12}$$

To assure some stability, it is imposed that the $\mathbf{z_B}$ vector must stay inside a cone around the world z-axis to prevent large attitude angles, expressed by

$$\mathbf{z_B} \cdot \mathbf{w_3} \geq \cos(\theta) \tag{13}$$

where θ is the half angle of the cone.

C. Optimal Control Problem

The optimal control problem, P_1, can now be formulated as

$$\begin{aligned} \min_{X,U} \quad & \int_{t_0}^{t_0+T_h} \left(h_p + h_v + h_\omega + h_R + h_u\right) dt \\ \text{s.t} \quad & \text{Equation (4)} \\ & \text{Equations (10), (11), (12) and (13)} \end{aligned} \tag{14}$$

where the error penalties in position, velocity, rotation, angular velocity and thrust are respectively given by

$$\begin{aligned} h_p &= \mathbf{e_p^T Q_p e_p} \\ h_v &= \mathbf{e_v^T Q_v e_v} \\ h_R &= \mathbf{e_R^T Q_R e_R} \\ h_\omega &= \boldsymbol{\omega_B}^{\mathbf{T}} \mathbf{Q}_{\boldsymbol{\omega}} \boldsymbol{\omega_B} \\ h_u &= Q_u e_u^2 \end{aligned}$$

and $\mathbf{Q_p}$, $\mathbf{Q_v}$, $\boldsymbol{Q_\omega}$ and $\mathbf{Q_R}$ are positive definite matrices, while Q_u is a positive constant.

IV. Joint Drones NMPC Formulation

In this section, the optimal control problem, as well as the constraints and respective cost functional, are adapted to support two vehicles simultaneously.

A. Problem Formulation

For the joint drones controller, it is necessary to consider the state variables and control inputs of both vehicles, thus it is defined a new state vector $\mathbf{x_{12}} = [\mathbf{x_1}, \mathbf{x_2}]$ and a new control input vector $\mathbf{u_{12}} = [\mathbf{u_{1_1}}, \mathbf{u_{1_2}}]$ In the same manner, all the penalties defined must be duplicated to accommodate both drones. Similar to the independent drone controller, it is necessary to first define the set of errors that will be weighted.

The reference tracking errors for position and velocity, for both drones, are defined as

$$\mathbf{e_{p_1}} = \mathbf{p_1} - \mathbf{p_{1ref}} \qquad \mathbf{e_{v_1}} = \mathbf{v_1} - \mathbf{v_{1ref}} \tag{15}$$

$$\mathbf{e_{p_2}} = \mathbf{p_2} - \mathbf{p_{2ref}} \qquad \mathbf{e_{v_2}} = \mathbf{v_2} - \mathbf{v_{2ref}} \tag{16}$$

while the errors between the desired and actual rotation are given by

$$\mathbf{e_{R_1}} = \frac{1}{2}\left({}^{\mathbf{W}}\mathbf{R}_{\mathbf{1B_{des}}}^{\mathbf{T}}\,{}^{\mathbf{W}}\mathbf{R_{1B}} - {}^{\mathbf{W}}\mathbf{R}_{\mathbf{1B}}^{\mathbf{T}}\,{}^{\mathbf{W}}\mathbf{R}_{\mathbf{1B_{des}}}\right)^\vee \tag{17}$$

$$\mathbf{e_{R_2}} = \frac{1}{2}\left({}^{\mathbf{W}}\mathbf{R}_{\mathbf{2B_{des}}}^{\mathbf{T}}\,{}^{\mathbf{W}}\mathbf{R_{2B}} - {}^{\mathbf{W}}\mathbf{R}_{\mathbf{2B}}^{\mathbf{T}}\,{}^{\mathbf{W}}\mathbf{R}_{\mathbf{2B_{des}}}\right)^\vee \tag{18}$$

with $\vee$ being the inverse of the skew matrix.

To achieve a stable steady state, it is desired that the angular velocities tend to 0 and the thrust generated by the propellers be equal to the force of gravity to achieve hover, therefore the angular velocity errors are given by

$$\mathbf{e}_{\boldsymbol{\omega_1}} = \boldsymbol{\omega_1} \qquad \mathbf{e}_{\boldsymbol{\omega_2}} = \boldsymbol{\omega_2} \tag{19}$$

while the thrust errors are defined as

$$e_{u_1} = u_{1_1} - u_{ss} \qquad e_{u_2} = u_{1_2} - u_{ss} \tag{20}$$

with $u_{ss} = mg$.

In order to introduce some cooperative capabilities between the quadcopters, two new variables were implemented, being the distance vector between both vehicles, defined as

$$\boldsymbol{e_D} = \boldsymbol{p_1} - \boldsymbol{p_2} \tag{21}$$

and the relative attitude angle between the drones, by defining the rotation between drone 1 and drone 2 as

$$\mathbf{R_L} = {}^{\mathbf{W}}\mathbf{R}_{\mathbf{B_1}}^{\mathbf{T}}\,{}^{\mathbf{W}}\mathbf{R_{B_2}} \tag{22}$$

and then calculating the error related to a desired angle $\mathbf{R_{Ld}}$ as

$$e_{RDs} = Tr\left(\mathbf{I_3} - \mathbf{R_{Ld}^T R_L}\right) \tag{23}$$

with Tr representing the trace of the matrix.

B. Constraints

Like the independent drone version of the controller, it is possible to include system's limitations in the optimal control problem when formulating the control input. The equations from the model (1) are used as constraints, using equation (4) for i = 1, 2.

To avoid crashing to the floor, it is imposed that both drone's position in the z axis must always be non-negative, since the height is always measured referencing the ground, expressed as

$$p_{1_z} \geq 0 \qquad p_{2_z} \geq 0. \tag{24}$$

The propellers of both quadcopters can provide a limited amount of thrust, and so the controller needs to take into account the actuator's saturation, given by

$$0 \leq u_{1_1} \leq u_{sat} \qquad 0 \leq u_{1_2} \leq u_{sat} \tag{25}$$

while the angular velocity, being also a function of the moments created by the propellers, is also limited, which can defined as

$$-\boldsymbol{\omega_1}_{max} \leq \boldsymbol{\omega_1} \leq \boldsymbol{\omega_1}_{max} \quad -\boldsymbol{\omega_2}_{max} \leq \boldsymbol{\omega_2} \leq \boldsymbol{\omega_2}_{max}. \tag{26}$$

The same stability constraint from the previous controller, being the $\mathbf{z_B}$ vector must stay inside a cone around the world z axis to prevent large attitude angles, is considered to the present controller, expressed by

$$\mathbf{z_{1B}} \cdot \mathbf{w_3} \geq \cos(\theta) \qquad \mathbf{z_{2B}} \cdot \mathbf{w_3} \geq \cos(\theta) \tag{27}$$

where θ is the half angle of the cone.

Due to the addition of cooperative capabilities between the two quadcopters, a new constraint imposing a minimal distance between them was implemented to prevent collisions, and is expressed as

$$\|\mathbf{e_D}\| \geq d_{min}. \tag{28}$$

C. Optimal Control Problem

The new variables and equations added to the NMPC problem require the addition of two new penalties on the cost functional, and thus the control problem P_{12} can be now defined as

$$\begin{aligned}\min_{X,U} \quad & \int_{t_0}^{t_0+T_h} (h_{e_D} + h_{e_{RDs}} + \\ & \qquad + h_{p_1} + h_{v_1} + h_{\omega_1} + h_{R_1} + h_{u_1} + \\ & \qquad + h_{p_2} + h_{v_2} + h_{\omega_2} + h_{R_2} + h_{u_2})\, dt \\ \text{s.t} \quad & \text{Equation (4)} \\ & \text{Equations (24), (25), (26), (27) and (28)}\end{aligned} \tag{29}$$

where the penalties on the distance between both vehicles and on the relative rotation between both drones are respectively given by

$$\begin{aligned} h_{e_D} &= \mathbf{e_D^T Q_D e_D} \\ h_{e_{RDs}} &= \mathbf{e_{RDs}^T Q_{RDs} e_{RDs}} \end{aligned}$$

and h_{p_i}, h_{v_i}, h_{ω_i}, h_{R_i}, h_{u_i} are the position, velocity, rotation, angular velocity and control input error penalties, respectively, for the drone i, as defined in III-C. $\mathbf{Q_D}$, $\mathbf{Q_{RDs}}$, $\mathbf{Q_p}$, $\mathbf{Q_v}$, $\mathbf{Q}_\omega$ and $\mathbf{Q_R}$ are positive definite matrices, while Q_u is a positive constant.

V. Simulation Results

After the formulation of the optimal control problems, both controllers were simulated. For validation, the performance of the independent controller was compared with a replicated controller [2]. The capabilities of the joint drones controller were tested by designing a cooperative parcel exchange scenario.

A. Independent Drone Controller

The controller was implemented resorting to the ACADO toolbox [5] with MATLAB interface, which generated a C/C++ block that can be imported to Simulink.

For the optimal control problem specifications, it was selected a sampling time $T_s = 0.05$ s, a time horizon $T_h = 1$ s, and $N = 20$ time steps. It was also chosen a maximum thrust $u_{sat} = 30$ N, a maximum angular rate $\boldsymbol{\omega}_{\mathbf{B}max} = \frac{\pi}{2}$ rad/s and a half angle $\theta = 45°$.

The penalty matrices selected for the position, velocity and rotation error, respectively, are

$$\mathbf{Q_p} = \begin{bmatrix} 12 & 0 & 0 \\ 0 & 12 & 0 \\ 0 & 0 & 16 \end{bmatrix}, \quad \mathbf{Q_v} = \begin{bmatrix} 9 & 0 & 0 \\ 0 & 9 & 0 \\ 0 & 0 & 8 \end{bmatrix},$$

$$\mathbf{Q_R} = 3 \cdot \mathbf{I_3}$$

while the matrix and weight to penalize the control inputs, angular velocity and thrust respectively, are

$$\mathbf{Q}_\omega = 1.5 \cdot \mathbf{I_3}, \quad Q_u = 1.$$

To test the performance of the controller, a complex trajectory, known as lemniscate, was applied as a reference signal, being defined as

$$\begin{cases} x = & 5 \cdot \cos(0.1 \times 2\pi t) \\ y = & 2.5 \cdot \sin(0.2 \times 2\pi t) \\ z = & 10 \end{cases} \tag{30}$$

where $t \in [0, 30]$. The vehicle starts at the first point of the curve, $\mathbf{p_0} = [5, 0, 10]^T$.

We will also compare the controller's performance, referred as *FullMPC*, with a replication of the [2] NMPC controller, referred as *FastMPC*. During implementation, we encountered some problems while trying to achieve a stable control and therefore it was necessary to impose some additional restrictions, which resulted in a slower response from this controller.

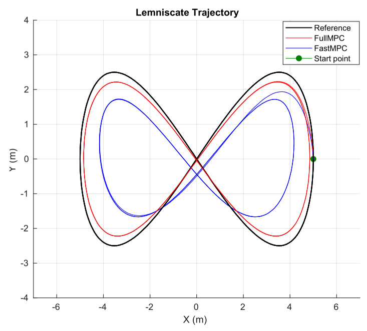

Fig. 2. Comparison between both controllers on a lemniscate trajectory

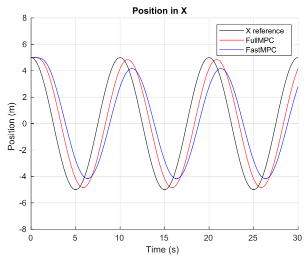

Fig. 3. Position in X in lemniscate trajectory

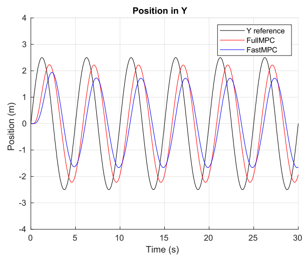

Fig. 4. Position in Y in lemniscate trajectory

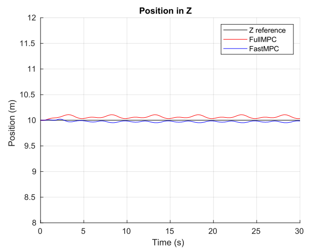

Fig. 5. Position in Z in lemniscate trajectory

As observed in Figures 2, 3, 4 and 5, the FullMPC controller has a better trajectory tracking performance than the FastMPC controller. This result is expected, as the former has a faster response. Note however that, due to a lack of integral effect, there's a deviation between the trajectory of the quadrotor and the reference signal.

Figure 6 presents the execution (or computation) times necessary for both controllers to compute the control actions. It is evident that the times required for computation of the control actions by the *FastMPC* are larger than the new controller. This is due to the fact that the controller FullMPC calculates less control inputs (four variables, as opposed to the six from the former controller), as well as the equations of the cost functional of the FastMPC require more complex computations. Also, the computation time from the FastMPC controller demonstrates spikes from around 2 ms to over 10 ms, which are result from the irregular shape of the angular velocity curves, that increase the computational load of the controller. However, it is noted that, for both controllers, the execution time is always smaller than the sampling time $T_s = 50$ ms.

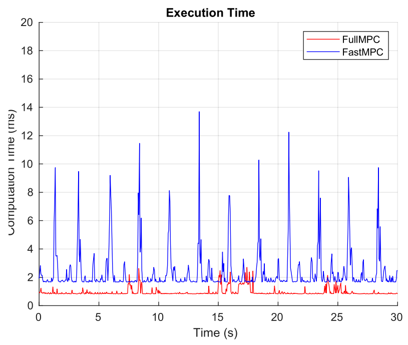

Fig. 6. Execution Time for in lemniscate trajectory

B. *Joint Drones Controller*

The joint controller was implemented resorting to the ACADO toolbox [5] with MATLAB interface, which generated a C/C++ block that can be imported to Simulink.

For the optimal control problem specifications, it was selected a sampling time $T_s = \ 0.05$ s, a time horizon $T_h = \ 1$ s, and $N = 20$ time steps. It was also chosen a maximum thrust $u_{sat} = 20$ N, a maximum angular rate $\boldsymbol{\omega}_{\boldsymbol{B_{max}}} = \pi$ rad/s, a half angle $\theta = 40^\circ$ and $d_{min} = 0.5$ m.

The penalty matrices selected for the position, velocity and rotation error, respectively, for both drones are

$$\mathbf{Q_{p_1}} = 10 \cdot \mathbf{I_3}, \quad \mathbf{Q_{v_1}} = 3 \cdot \mathbf{I_3}, \quad \mathbf{Q_{R_1}} = 5 \cdot \mathbf{I_3}$$
$$\mathbf{Q_{p_2}} = 10 \cdot \mathbf{I_3}, \quad \mathbf{Q_{v_2}} = 3 \cdot \mathbf{I_3}, \quad \mathbf{Q_{R_2}} = 5 \cdot \mathbf{I_3}$$

while the matrices and weights to penalize the control inputs, angular velocity and thrust respectively, for both vehicles are

$$\mathbf{Q_{w_1}} = \mathbf{I_3}, \quad Q_{u_1} = 1$$
$$\mathbf{Q_{w_2}} = \mathbf{I_3}, \quad Q_{u_2} = 1.$$

and to penalize the distance between drones and their relative angle, the matrix and weight used are

$$\mathbf{Q_{e_D}} = 5 \cdot \mathbf{I_3}, \quad Q_{e_{RDs}} = 1.$$

A cooperative manoeuvre was designed to test the capabilities of the controller. For this specific scenario, it is assumed that drone d_2 is carrying a parcel on the top, while the drone d_1 will collect it from below.

The weights used throughout the manoeuvre will change depending if we want to move the drones to a specific point or to a specific distance between them. Initially, the penalty of the distance between the two vehicles is set to a zero matrix, while the position and velocity penalties are set to their original values. The relative angle penalty is not used during the manoeuvre, so it is always set to 0.

The two drones start in positions $\mathbf{p_{01}} = (-2, -2, 0)$ and $\mathbf{p_{02}} = (2, 2, 0)$, respectively, and rise to points $\mathbf{p_1} = (0, 0, 5)$ and $\mathbf{p_2} = (5, 0, 5)$. After that, both the position error and velocity error penalties and set to a zero matrix, while the penalty of the distance between drones is set to its initial value $\mathbf{Q_{e_D}}$, and the drones are instructed to position themselves 2 meters apart, in the z axis, followed by positioning 0.5 meters apart, moment where the relay can occur, return then to position 2 meters apart. After that, the position and velocity penalties are set again to their initial values, while the distance penalty is set to a zero matrix, and the drones are instructed to land at positions $\mathbf{p_1} = (2, -2, 0)$ and $\mathbf{p_2} = (-2, 2, 0)$. A 3-D representation of the manoeuvre is presented in Figure 7.

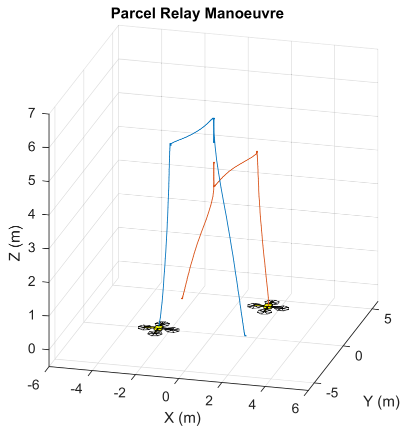

Fig. 7. 3D Plot of the cooperative manoeuvre

Figures 8 to 15 present the position, rotation, thrust and angular velocity of both vehicles, respectively, during the manoeuvre. Analyzing each figure, it is evident that the constraints imposed on the problem are respected, as confirmed by noting that, for both drones, the angular velocity values are smaller than the imposed π rad/s, while the roll and pitch rotation angles are always within the $\theta = 40^\circ$ limit, and the thrust values are within the 20 N saturation.

Figure 16 presents the necessary times to compute the control actions required to perform the manoeuvre. Analyzing the figure, it is evident that the times required are superior when compared with the independent drone controller, as expected due to the duplication of variables and equations being used, implying extra computation. However, the execution times in

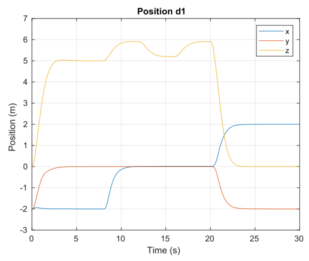
Fig. 8. Position of drone d1

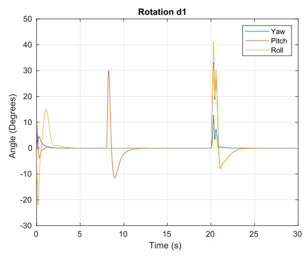
Fig. 9. Rotation of drone d1

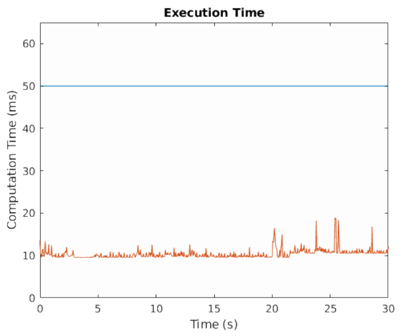
Fig. 16. Execution Time for the cooperative manoeuvre

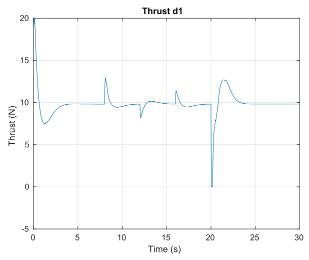
Fig. 10. Thrust of drone d1

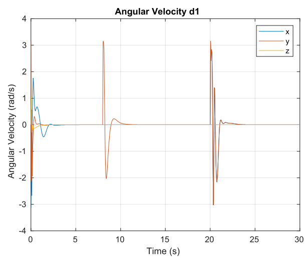
Fig. 11. Angular velocity of drone d1

the presented test are still always lower than the set sampling time $T_s = 50$ ms, an important factor for the feasibility of the manoeuvre. Near the end of the simulation, some spikes appear as a result of moving the vehicles near constrained values, in this case landing having the (24) restriction, as it was noted during testing that the computational load increases when operating near the imposed constraints.

VI. Conclusion

The goal of this paper is to address the problem of designing a nonlinear model predictive controller capable of handling agile dynamics and operate over fast trajectories, in order to have two drones perform a relay manoeuvre, exchanging of a payload between them, in an efficient and safe way. Firstly, the vehicle's model was established allowing for the formulation of optimal control problem. After validation, a complex trajectory was applied to test its tracking performance, which was then compared with the performance of the implementation of another controller, concluding that the studied controller allows for a better tracking of faster trajectories, evident by the faster velocities and steeper angles achieved. However, it is evident the lack of integral effect, as there is a deviation between the trajectory of the quadrotor and the reference signal.

After the validation of the NMPC controller, a strategy consisting on adapting the optimal problem to consider two vehicles at the same time, and imposing a set of constraints that are favorable to perform the relay manoeuvre was implemented. The controller was implemented in Simulink and a parcel exchange scenario was designed. By analysis of the results, we conclude that the computational load of the controller increases due to the addition of more state variables and error calculations, however, the vehicles respect the imposed constraints and the execution time is inferior to the set sampling time, which shows the feasibility of the controller.

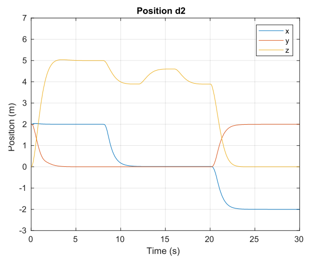
Fig. 12. Position of drone d2

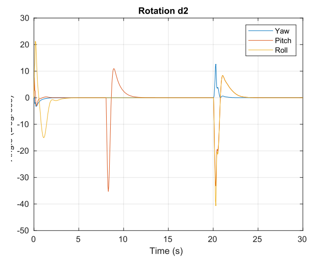
Fig. 13. Rotation of drone d2

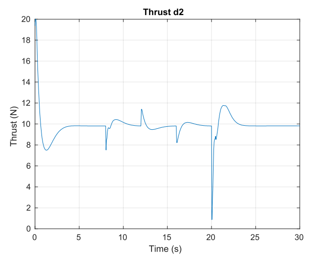
Fig. 14. Thrust of drone d2

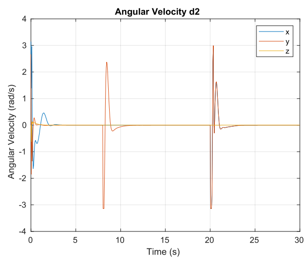
Fig. 15. Angular velocity of drone d2

References

[1] H. Merabti, I. Bouchachi, and K. Belarbi. "Nonlinear model predictive control of quadcopter." In: 16th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering, STA 2015 (2016), pp. 208–211. doi: 10.1109/STA.2015.7505151.

[2] M. Kamel, K. Alexis, M. Achtelik, and R. Siegwart. "Fast nonlinear model predictive control for multicopter attitude tracking on SO(3)." In: 2015 IEEE Conference on Control and Applications, CCA 2015 - Proceedings 3 (2015), pp. 1160–1166. doi: 10.1109/CCA.2015.7320769.

[3] D. Mellinger, N. Michael, and V. Kumar. "Trajectory generation and control for precise aggressive maneuvers with quadrotors." In: International Journal of Robotics Research 31.5 (2012), pp. 664–674. issn: 02783649. doi: 10.1177/0278364911434236

[4] Project REPLACE. http://replace.isr.tecnico.ulisboa.pt/. Accessed: 2020-01-17.

[5] B. Houska, H. Ferreau, M. Vukov, and R. Quirynen. ACADO Toolkit User's Manual. http://www.acadotoolkit.org. 2009–2013.