Nonlinear Model Predictive Control for Aerial Manipulation

Dario Lunni, Angel Santamaria-Navarro, Roberto Rossi, Paolo Rocco, Luca Bascetta and Juan Andrade-Cetto

***Abstract*— This paper presents a nonlinear model predictive controller to follow desired 3D trajectories with the end effector of an unmanned aerial manipulator (*i.e.*, a multirotor with a serial arm attached). To the knowledge of the authors, this is the first time that such controller runs online and on board a limited computational unit to drive a kinematically augmented aerial vehicle. Besides the trajectory following target, we explore the possibility of accomplishing other tasks during flight by taking advantage of the system redundancy. We define several tasks designed for aerial manipulators and show in simulation case studies how they can be achieved by either a weighting strategy, within a main optimization process, or a hierarchical approach consisting on nested optimizations. Moreover, experiments are presented to demonstrate the performance of such controller in a real robot.**

I. Introduction

Over the last few years, the application field of unmanned aerial vehicles (UAVs) has been considerably expanded from "passive" applications, like monitoring or surveillance, to "active" operations which require interactions with the environment (*e.g.*, manipulation). With the improvements of brush-less motors and batteries, new sophisticated UAV prototypes have been developed that can physically interact with the environment. These high-performance vehicles, called unmanned aerial manipulators (UAMs), usually consist on a multirotor platform with a robot arm attached underneath (*e.g.*, the UAM shown in Fig. 1).

First studies on UAMs analyzed how load transportation affects the vehicle dynamics [1], [2]. In most of those works, controllers were designed to achieve trajectory tracking with the aerial vehicle during the navigation phases. However, when using a serial arm able to move with respect to the platform, the models involved in the design of these controllers are complicated due to dynamic coupling between the parts of the system [3]. A common practice is to simplify the controller design and consider the dynamics of each part decoupled. In [4] a scheme of nested proportional-integral-derivative controllers (PIDs) was designed for attitude stabilization, vision-based navigation and a gripping maneuver. Similar PID schemes are compared with an integral backstepping controller in [5]. Among the undesired dynamic effects, there is the change of the center of mass during flight, that can be solved designing a low-level attitude controller such as a Cartesian impedance controller [6], or an adaptive controller [7]. Moreover, a desired end effector pose might require a non-horizontal robot configuration that the low level controller would try to compensate, changing in turn the arm end effector position. In this way, [8] designs a controller exploiting the manipulator and multirotor models, however flight stability is preserved by restricting the arm movements to those not jeopardizing UAM integrity.

Fig. 1. Aerial manipulator, used in the experiments, consisting on a 3DR-X8 coaxial multirotor platform with a custom-built 3 degrees-of-freedom serial arm attached below.

Recently, there has been growing interest in using numerical optimal control for aerial vehicles. In [9] a method is presented for 3D trajectory generation of a UAV limiting the 3 translational jerks and accelerations. The computed trajectory is then fed to a closed-loop control, achieving a similar performance as in the case of using model predictive control. Similarly, [10] describes a system for real-time generation of optimal trajectories with minimum snap, allowing large excursions from the hover position during operations like take-off and navigation. In both [9] and [10] the generated trajectories are for nonholonomic multirotors, which at a high level of control have 4 degrees of freedom (DoF). In [11], this trajectory generation is optimized for a vehicle with a cable-suspended load computing nominal trajectories with various constraints regarding the load swing. All these trajectory generation methods are actually considered part of the planning module.

The application of optimal control to UAMs can be seen in [12] where a nonlinear model predictive control scheme (NMPC) is described using a direct multiple shooting method, but results for a multirotor equipped with a serial arm are only shown in a simulated environment. Similarly, [13] presents an NMPC to achieve pick-and-place operations with a real robot. However, both in [12] and [13] the

Dario Lunni, Roberto Rossi, Paolo Rocco and Luca Bascetta are with Politecnico di Milano, Dipartimento di Elettronica, Informazione e Bioingegneria, Piazza Leonardo da Vinci, 32, Italy, E-mail: dario.lunni, roberto.rossi, paolo.rocco, luca.bascetta @polimi.it

Angel Santamaria-Navarro and Juan Andrade-Cetto are with the Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Llorens i Artigas 4-6, Barcelona 08028, Spain, E-mail: asantamaria,cetto@iri.upc.edu. Their work was partially funded by the EU project AEROARMS H2020-ICT-2014-1-644271, and by the Spanish Ministry of Economy and Competitiveness project ROBINSTRUCT TIN2014-58178-R.

NMPCs are used for offline trajectory generation and online performance is left for future work. Moreover, none of these methods aims to accomplish several tasks simultaneously by exploiting the redundancy of UAMs in terms of degrees-of-freedom (DoFs).

The redundancy of UAM systems (*i.e.*, 4 DoFs from the multirotor and, at least, 3 DoFs from the arm system), can be exploited not only to achieve a primary task (*e.g.*, trajectory tracking) but also a lower priority stabilizing task by optimizing some given quality indices (*e.g.*, improving manipulability or avoiding joint limits) as in [14], [15]. In these works, as in most of recent approaches (*e.g.*, [16], [17], [18]) the problem is solved using hierarchical task composition control but in almost all cases without using optimal control. A close approach is [19], which presents a trajectory generation method for UAMs through quadratic programming, taking advantage of redundancy. However, the hierarchy of tasks is softly imposed using a weighted sum and conflicts with antagonistic tasks would arise.

In this paper, we combine both ideas of using optimal control and accomplishing several tasks with kinematically redundant UAMs. We present an NMPC controller for UAMs able to work online and on board a limited computational unit. We consider a task consisting on a 3D trajectory tracking with the arm end effector, and define other tasks to improve the arm manipulability and align the arm center of gravity (CoG) with the platform gravitational vector. Moreover, system bounds and constraints are set in the optimization process to limit the UAM motion and avoid self-collisions with the arm end effector. The use of these tasks is shown in simulation case studies by using a weighting strategy and a nested optimization scheme which allows us to impose a hierarchy. Real robot experiments are presented to show the performance of the NMPC.

The remainder of this article is structured as follows. The robot kinematics is presented in the following Section. Section III provides the NMPC details. Tasks and constraints are defined in Sections IV and V, respectively. Section VI presents the validation of the method throughout simulations and real experiments. Finally, conclusions are given in Section VII.

II. Robot model

Multirotors are equipped with more than two aligned coplanar propellers. Due to their symmetric design, motion control is achieved by altering the rotation rate of one or more of these propellers, thereby changing its torque load and thrust lift characteristics. At a high level of control, a multirotor is an underactuated vehicle with only 4 DoFs, *i.e.*, the platform tilting (roll and pitch variables) is used by the low level attitude controller to produce desired translational velocities of the vehicle. Instead, attaching a robotic arm with more than two degrees of freedom to a multirotor the whole system becomes redundant.

In this paper, we consider a UAM composed of a free-flying platform (*i.e.*, a multirotor) with a serial arm attached, allowing not only to accomplish tasks using the arm end effector (*e.g.*, end effector positioning) but also to exploit the whole system redundancy to achieve internal motions that ease the flight behaviour. The multirotor-arm system considered has n DoFs, with $n \geq 7$, defined as follows.

Let ${}^i\boldsymbol{p}_b$ and ${}^i\boldsymbol{\phi}_b$ denote the absolute position and orientation (*i.e.*, the triple of ZYX yaw-pitch-roll angles) of the vehicle body (b) expressed in a global inertial reference frame (i). Let $\boldsymbol{q}$ be the arm joint positions. The complete state of the UAM can be described by

$$\boldsymbol{\xi} = \begin{bmatrix} {}^i\boldsymbol{p}_b{}^\top & {}^i\boldsymbol{\phi}_b{}^\top & \boldsymbol{q}^\top \end{bmatrix}^\top . \tag{1}$$

Considering the multirotor underactuation, it is worth to separate the non controllable DoFs in the state, leading to

$$\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\mu}^\top & \boldsymbol{\sigma}^\top \end{bmatrix}^\top , \tag{2}$$

where $\boldsymbol{\mu}$ is the vector of controlled variables (*i.e.*, platform position plus yaw angle and arm joint variables) and $\boldsymbol{\sigma}$ the non controlled variables (*i.e.*, platform roll and pitch).

III. Nonlinear model predictive control

Model predictive control (MPC) is a control technique that consists in numerically solving an optimization problem at each time step. A model of the process is used to predict the future evolution of the system in order to optimize the control signal. This future time horizon, in which the algorithm is computed, is called prediction horizon. NMPC refers to particular MPC problems where the process model is nonlinear, the cost functional is nonquadratic or general nonlinear constraints are used.

A. Weighting strategy

Considering a generic dynamic system, which has a state $\boldsymbol{x}$ and is controlled by the variables in $\boldsymbol{u}$, the solution of the optimal control problem at time $t_k = kT_s$ ($\forall\ 1 < k < N$), where T_s is the time step, over a finite prediction horizon of N steps, is defined by

$$\begin{aligned} \min_{\mathbf{u}} \quad & h(\boldsymbol{x}_k, \mathbf{u}, t_k) \\ s.t. \quad & \boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k) \\ & \boldsymbol{y}_{min} \leq g(\boldsymbol{x}_k, \boldsymbol{u}_k) \leq \boldsymbol{y}_{max} \\ & \boldsymbol{x}_{min} \leq \boldsymbol{x}_k \leq \boldsymbol{x}_{max} \\ & \boldsymbol{u}_{min} \leq \boldsymbol{u}_k \leq \boldsymbol{u}_{max} \end{aligned} \tag{3}$$

where $\mathbf{u} = \begin{bmatrix} \boldsymbol{u}_k^\top, \boldsymbol{u}_{k+1}^\top, \ldots, \boldsymbol{u}_{k+N-1}^\top \end{bmatrix}^\top$ is a vector of control variables, f is the model of the system, g represents a generic constraint, and h is a generic cost function defined by

$$\begin{aligned} h(\boldsymbol{x}_k, \mathbf{u}, t_k) = \sum_{i=0}^{N-1} \Big(& \sum_{j=0}^{N_h-1} h_j + ||\boldsymbol{u}_{k+i} - \boldsymbol{u}_{k+i-1}||^2_{W_u} \Big) \\ & + ||\boldsymbol{u}_{k+N}||^2_{W_s} \Big) , \end{aligned} \tag{4}$$

where $h_j = h_j(\boldsymbol{x}_{k+i}, \boldsymbol{u}_{k+i}, t_k)$ are N_h generic positive cost functions to be minimized, and W_u and W_s weights on control actions and terminal cost respectively. Notice how in (3) the optimization process is constrained by the dynamic model of the system, $\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k)$, subject to

lower and upper bounds on state, control actions, and output variables defined by $\boldsymbol{x}_{min}$ and $\boldsymbol{x}_{max}$, $\boldsymbol{u}_{min}$ and $\boldsymbol{u}_{max}$ and $\boldsymbol{y}_{min}$ and $\boldsymbol{y}_{max}$, respectively.

B. Hierarchical approach

Instead of following a weighting strategy, we can impose a hierarchy between the tasks (cost functions), similarly to [17] and [18], but in this case using optimal control. The required scheme is based on nested optimizations as explained in the following.

Drawing inspiration from [20], we propose to compute a cascade of optimizations for each time step in order to minimize different cost functionals, each one related to a different task. However and differently from [20], we do not require to linearize the system. Initially, we solve the optimal control problem considering the cost functional of a main task (*e.g.*, the end effector trajectory tracking described later on in Section (IV-A)). From this minimization we are able to compute the predicted values of the cost functional for the N future steps. Then, the hierarchy is imposed by adding a constraint in the secondary task resolutions. In particular, the value of the cost function at higher priority is set as a constraint to the next optimization iteration.

For a secondary task, the minimization of the cost functional follows the same procedure as for the primary task, but this time incorporating the constraint resulting from solving the primary task (see (3)), defined by

$$g_1(\boldsymbol{x}_k, \mathbf{u}, t_k) \;=\; 0\,, \;\; \forall k. \tag{5}$$

This method can be repeated as many times as wanted for several subtasks imposing the hierarchy, depending on the redundancy of the robot with respect to the task definitions, *e.g.*, the optimization solver will find a solution in the alloted time only if the tasks higher in the hierarchy do not require that DoF to be accomplished (*i.e.*, if the optimization process does not have a constraint for that DoF).

C. System model in the optimization problem

In this paper we consider that UAMs are not meant for acrobatic maneuvers. Hence, we will not include the platform tilt in the trajectory generation algorithm (platform roll and pitch angles will be assumed negligible in our analysis). Then, for the optimization problem we can consider the state

$$\boldsymbol{x} = \boldsymbol{\mu} = \begin{bmatrix} {}^i\boldsymbol{p}_b^\top & {}^i\psi_b & \boldsymbol{q}^\top \end{bmatrix}^\top , \tag{6}$$

while the control action is directly the state derivative

$$\boldsymbol{u} = \dot{\boldsymbol{\mu}} = \begin{bmatrix} {}^i\dot{\boldsymbol{p}}_b^\top & {}^i\dot{\psi}_b & \dot{\boldsymbol{q}}^\top \end{bmatrix}^\top . \tag{7}$$

Formulating the control action in differential form is a well-known method to obtain zero steady state error. The function of the system model f is just an integrator of input $\boldsymbol{u}$. The non-linearity of the approach appears in the definition of the cost function and in the constraints. The choice of this reduced system model was made to guarantee online computation, on board a platform with limited resources.

Hereafter, we assume that the inner control loop of the system can perfectly track the computed references. There are some works [21], [22] which show that adding a manipulator introduces unwanted external disturbances to the multirotor which, if not compensated for, will make it unstable. Although, with slow arm movements (as in our case) the inner-loop controller can already compensate most of these effects, one has to specially consider from the following tasks, the ones designed to improve stability of the platform.

IV. UAM TASKS

We can consider several cost functions for UAMs (3), in order to accomplish a given task or to preserve stability. In this Section we present a few of such task functions.

A. End effector tracking

The main interaction task will be executed by the arm end effector, it is thus important to be able to track a desired end effector trajectory. The following cost function can be defined

$$h_1 = \sum_{i=0}^{N-1} ||\boldsymbol{e}_e\left(\boldsymbol{x}_{k+i}\right)||^2_{W_1} + ||\boldsymbol{e}_e\left(\boldsymbol{x}_{k+N}\right)||^2_{W_{s1}}\,, \tag{8}$$

where

$$\boldsymbol{e}_e\left(\boldsymbol{x}_{k+i}\right) = {}^i\boldsymbol{p}_e(\boldsymbol{x}_{k+i}) - {}^i\boldsymbol{p}_e^d\left(t_{k+i}\right) \tag{9}$$

is the end effector tracking error, ${}^i\boldsymbol{p}_e^d$ is the desired end effector position over the predicted horizon, and W_1 and W_{s1} are the weight matrices for the task and the terminal cost. Although h_1 encloses the end effector position error, notice how this cost function can also consider its orientation by augmenting with ${}^i\boldsymbol{\phi}_e$ and ${}^i\boldsymbol{\phi}_e^d$ the current and desired end effector poses, respectively.

Notice that each cost function presented in the following includes a terminal cost. In a number of works, collected in [23], it is pointed out that terminal cost weights are a key ingredient to achieve stability with NMPCs. However, stability proof of the proposed NMPC is outside the scope of the present work.

B. Robot center of gravity alignment

When the arm CoG is not aligned with the geometrical center of the platform, undesired torques appear in the vehicle's base, which must be compensated with the action of the propellers. In order to minimize this actuation effort and avoid instability, it is beneficial to design a task to favor this alignment, such as

$$h_2 = \sum_{i=0}^{N-1} ||{}^b\boldsymbol{p}_{Gxy}(\boldsymbol{x}_{k+i})||^2_{W_2} + ||{}^b\boldsymbol{p}_{Gxy}\left(\boldsymbol{x}_{k+N}\right)||^2_{W_{s2}}\,, \tag{10}$$

where ${}^b\boldsymbol{p}_{Gxy}(\boldsymbol{x}_{k+i})$, is the vector describing the position of the arm CoG projected onto the xy plane of the body reference frame during the prediction horizon. This expression can be obtained as in [17], [18]. Notice that we are assuming that the quadrotor is internally balanced. Otherwise, a different equilibrium point should be assigned for the arm CoG.

The formulation of (10) can effectively reduce the static torques produced on the quadrotor. However, an additional cost function can be considered in order to reduce the disturbances produced by inertial forces on the quadrotor. In fact, the following cost function accounts for the velocity of the CoG:

$$h_3 = \sum_{i=0}^{N-1} ||^b \boldsymbol{v}_{Gxy}\,(k+i)\,||^2_{W_3} + ||^b \boldsymbol{v}_{Gxy}\,(k+N)\,||^2_{W_{s3}}\,, \tag{11}$$

with

$$^b \boldsymbol{v}_{Gxy}\,(k+i) = {}^b J_{Gxy}(\boldsymbol{x}_{k+i})\,\boldsymbol{u}_{k+i}\,, \tag{12}$$

and ${}^b J_{Gxy}(\boldsymbol{x}_{k+i})$ the arm CoG Jacobian. Penalizing the motion of the CoG has the goal of reducing the inertial forces resulting on the quadrotor plane, thus on axes orthogonal to the thrust direction.

C. Arm manipulability

During a manipulation task, a useful objective function is represented by the arm manipulability index. A first direct way of expressing this cost function is

$$h_4 = \sum_{i=0}^{N-1} W_4 \frac{1}{||D\,(\boldsymbol{x}_{k+i})\,||^2} + W_{s4} \frac{1}{||D\,(\boldsymbol{x}_{k+N})\,||^2}\,, \tag{13}$$

where

$$D\,(\boldsymbol{x}_{k+i}) = \det\left(J_{e,q}\,(\boldsymbol{x}_{k+i})\,J_{e,q}\,(\boldsymbol{x}_{k+i})^\top\right)\,. \tag{14}$$

$J_{e,q}\,(\boldsymbol{x}_{k+i})$ is the submatrix of the end effector Jacobian $J_e\,(\boldsymbol{x}_{k+i})$ composed by the columns corresponding to the arm joints.

A second cost function useful to maximize the manipulability of the arm is obtained by minimizing the conditioning number of matrix $J_{e,q}J_{e,q}^\top$:

$$h_5 = \sum_{i=0}^{N-1} W_5||1-\rho_{J_{eq}}(k+i)||^2 + W_{s5}||1-\rho_{J_{eq}}(k+N)||^2\,, \tag{15}$$

with

$$\rho_{J_{eq}}(k+i) = \rho\Big(J_{e,q}\,(\boldsymbol{x}_{k+i})\,J_{e,q}\,(\boldsymbol{x}_{k+i})^\top\Big)\,. \tag{16}$$

$\rho(A) = \lambda_{A,M}/\lambda_{A,m}$ is the conditioning number of a matrix A, with $\lambda_{A,M}$ and $\lambda_{A,m}$ the maximum and minimum eigenvalues, respectively. Notice that, far from singularity points, the matrix $J_{e,q}J_{e,q}^\top$ is symmetric positive definite, then its eigenvalues are real and positive. Notice also that the closer the cost function h_5 in (15) is to zero, the more the manipulability ellipsoid is similar to a generalized sphere.

The weights W_4, W_5, W_{s4} and W_{s5} are scalar values.

A third alternative formulation of the cost function related to manipulability can be obtained by applying the gradient based method in [24]. It would be expressed with a cost function on the control action $\boldsymbol{u}$.

D. Cost function on the control action

The cost function $h(\boldsymbol{x}_k, \mathbf{u}, t_k)$ includes the term $||\boldsymbol{u}_{k+i} - \boldsymbol{u}_{k+i-1}||^2_{W_u}$ penalizing the control action. Notice that the control action for our system consists in the derivatives of UAM joint positions. Then, with the weight matrix W_u properly set, the motion can be arbitrarily distributed on the UAM joints. For instance, it can be desirable to penalize quadrotor motion in favor of arm DoFs during manipulation tasks, and vice versa during navigation phases.

V. Specific NMPC Constraints for UAMs

We introduce a first constraint consisting in avoiding self-collisions. To do so, we impose a safety distance between the joints and the aerial base with

$$\begin{aligned} {}^b z_i\,(\boldsymbol{x}_k) &\geq 0.1\,[m] \\ {}^b z_e\,(\boldsymbol{x}_k) &\geq 0.1\,[m]\,, \end{aligned} \tag{17}$$

where ${}^b z_e$ is the position of the end effector in the z direction of the body reference frame and, similarly, ${}^b z_i$ is the distance of the i-th joint. Notice that just a limited number of points of the arm can potentially have a collision with the aerial base, reducing the number of required constraints. These constraints will be integrated in the optimization through the term g (see (3)). Thanks to the flexibility of the method, a generic constraint can be added to the problem.

A part from constraints, we also impose bounds to both joint positions and velocities. Joint position bounds are expressed as state bounds, by imposing specific values to $\boldsymbol{x}_{min}$ and $\boldsymbol{x}_{max}$. On the other hand, velocity bounds correspond to control action constraints, thus they can be obtained by setting values of $\boldsymbol{u}_{min}$ and $\boldsymbol{u}_{max}$.

VI. Validation and experimental results

A. Preliminary simulations using a reduced model with a weighting task strategy

In order to validate the described controller, we first present simulation case studies programmed in MATLAB and considering a simplified model. In particular, the system consists of a planar multirotor, thus described by three DoFs, roll angle, y displacement and z altitude, and a 2 DoFs robotic arm. As described in Section III, roll angle is not considered in the state of the NMPC model. Therefore, the model of the system has 4 DoFs. The aim of this preliminary test is to show how several cost functions can be integrated using different weights, and solved exploiting the redundancy. The main cost function is the end effector trajectory tracking h_1. The end effector desired trajectory is described by the two translational positions y_e and z_e. In addition, the CoG cost functions h_2 and h_3 are integrated in the model.

The low cardinality of the model state allows to include the manipulability cost functions h_4 and h_5 too, which is quite difficult for systems of higher order, due to the required computational burden. This is only possible for those optimization solvers able to obtain a solution even

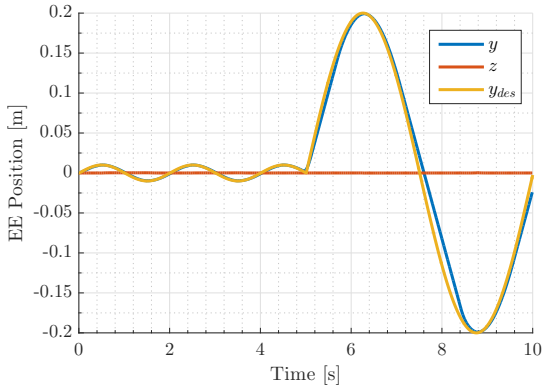


Fig. 2. End effector trajectory compared with the reference signal. Notice how the z reference is not reported because it is always null.

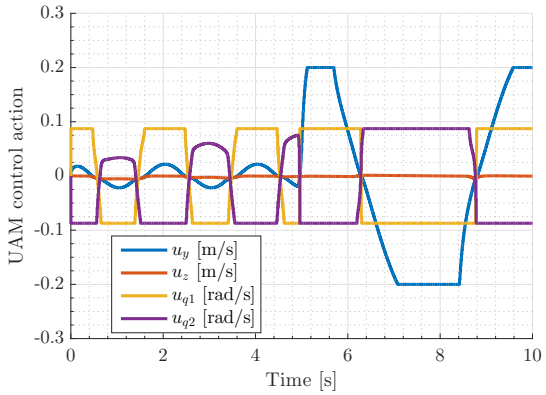


Fig. 3. Control actions on the four degrees of freedom computed by the NMPC. Blue and red lines represent multirotor translational DoFs, while yellow and violet lines represent arm joint velocities.

without the analitical Jacobians of the cost functions (*i.e.*, using numerical computations).

The desired end effector trajectory is in y direction and it is divided in two phases. In the first phase a small sinusoidal trajectory is required, while the second phase is characterized by a bigger sinusoid.

The chosen optimization solver for the non linear problem is the ACADO Toolkit [25]. The system can be set with different weight ratios among the cost functions, depending on the desired interactions between them. Hereafter, we present the results of a test in which all the five cost functions have weights different from zero.

In Fig. 2, the end effector trajectory is reported. The reference trajectory is correctly tracked in both directions. The z reference is not reported because it is always null. In Fig. 3, the control actions, output of the NMPC algorithm, are represented. Notice that bounds on maximum velocity are respected. It is interesting to show how the motion is distributed on different joints. In fact, analyzing the blue line, corresponding to the control action on y, it is evident that the small sinusoidal motion is mainly performed with the arm joints, while the large sinusoid is performed using multirotor DoFs.

Fig. 4 reports the behavior of the conditioning number $\rho_{J_{e,q}}$, defined in (16), by showing results of three experiments, performed with different weights. The blue line

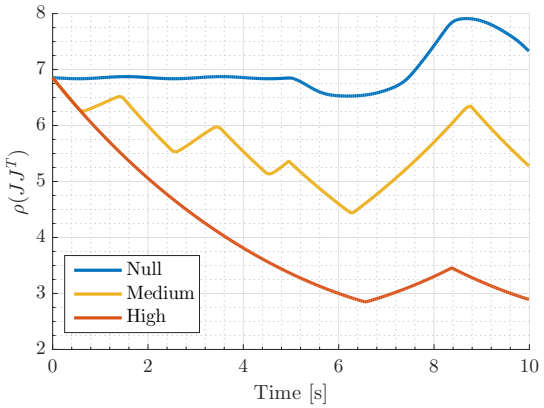


Fig. 4. Behavior of the conditioning number $\rho_{J_{e,q}}$ in three different experiments, differing in the weight assigned to the conditioning number cost function. The red line represents the experiment with larger assigned weight, achieving a smaller $\rho_{J_{e,q}}$, whereas the blue line is the result of the experiment with zero weight assigned to the cost function, thus not reducing $\rho_{J_{e,q}}$.

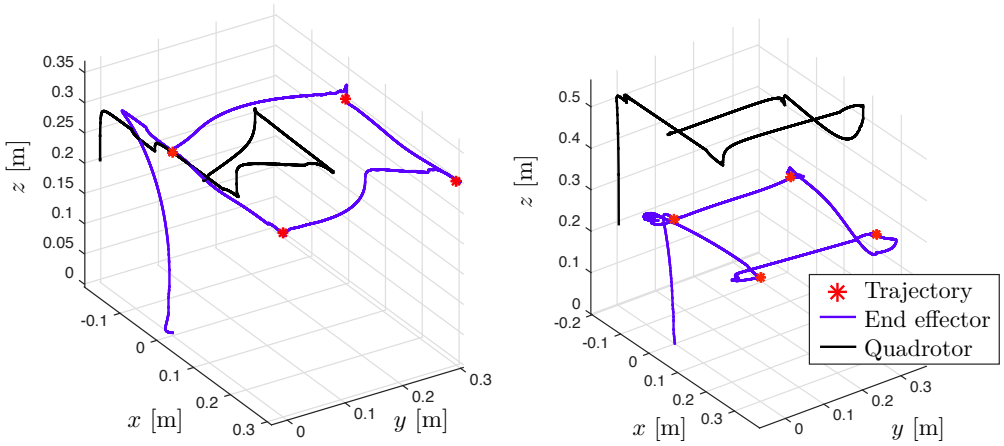
represents an experiment when the cost function h_5 is not included in the optimization process. The red line is a result of an experiment in which a large weight was assigned to this cost function, while the yellow line is the case of the experiment presented in the previous Figures. The approach can effectively reduce the conditioning number $\rho_{J_{e,q}}$ with a proper choice of the associated weight.

A clear disadvantage of a weighting strategy is the lack of good solutions when tasks are antagonistic (*i.e.*, when the accomplishment of a task requires the oposite solution of an other task). Instead, when using a hierarchical solution, we cannot guarantee the fulfillment of a secondary task, but at least we can find reasonable solutions for tasks higher in the hierarchy. In the following sections simulations and experiments using a hierarchical NMPC are presented.

B. Simulations using a complete model and a hierarchical task strategy

In this Section a simulation of a fully actuated UAM consisting on a multirotor with 4 DoFs plus a 3-DoF serial arm (UAM of 7 DoFs) is used to show how the secondary tasks can also be executed with a different approach.

The effect of adding a secondary task hierarchically is shown in Fig. 5. In this simulation the controller is set with a prediction horizon of 8s. In both cases the optimal control drives the end effector through the required waypoints. This comparison is then extended to Fig. 6 with a different trajectory. Here, it is reasonable to expect the improvement of the flight behaviour when the whole CoG is vertically aligned with the platform center of rotation. Moreover, considering that most of the workspace of the arm exists below the aerial platform (*i.e.*, without colliding with the multirotor), it is clearly shown how the multirotor (black line trajectory) performs quite better when a secondary task to align the arm CoG is present (Fig. 6(b)). In these particular simulations we removed all motion bounds to show how, in the case where only the tracking task is present (Fig. 6(a)), the platform and end effector are moving in similar vertical positions, thus with the arm fully extended in a forward position.



(a) Non-hierarchical NMPC (b) Hierarchical NMPC

Fig. 6. Simulation results for 3D trajectories, done by the arm end effector using only the positioning task (non-hierarchical) or adding a secondary task (hierarchical) to vertically align the arm CoG and improve the platform flight behavior. In both cases the main task consists in following 4 desired waypoints with the arm end effector.

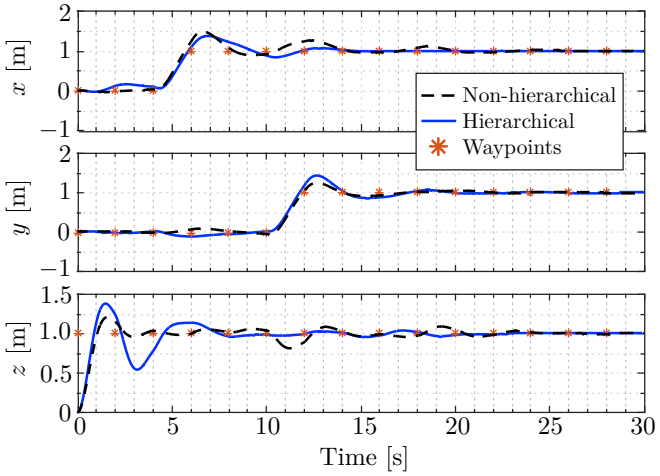


Fig. 5. Simulation results for 3D end effector positioning. NMPC performance comparison between applying only the positioning task (non-hierarchical, dashed black line) or together with a secondary task (hierarchical, blue line)

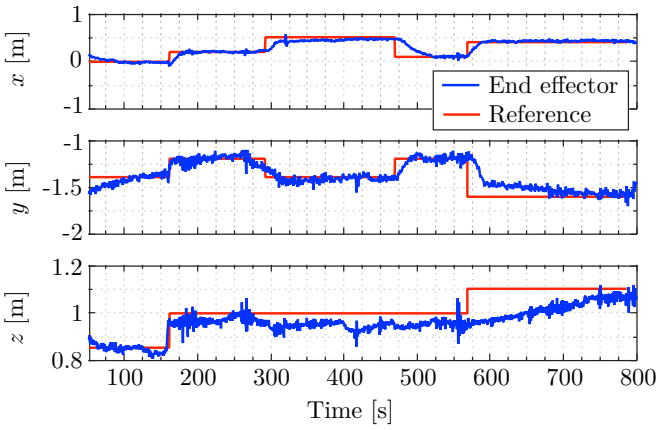


Fig. 7. Real arm end effector positions during a trajectory following task and using the hierarchical NMPC, with the arm CoG alignment as secondary task. The arm end effector position (blue line) has been obtained using a motion capture system.

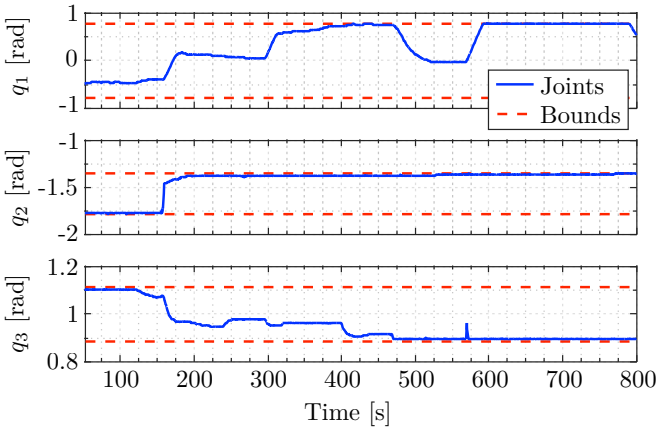


Fig. 8. Arm joint positions (q_i) during the same trajectory following task and hierarchical NMPC as in Fig. 7. The horizontal dashed lines correspond to lower and upper bounds of a constraint, set to avoid self-collisions.

C. Experiments

Here we present the results of the implementation of the NMPC strategy on the real UAM shown in Fig. 1. The platform is equipped with a Pixhawk board to allocate lower level controllers for the multirotor and the arm (PID controllers), which are tuned to track the references provided by the NMPC. As in the previous Section, the optimization algorithm has been implemented using the ACADO toolkit [25], already used to generate complex 3D trajectories for multirotors [26], and runs on board an ODROID XU3 at 2Hz with a prediction horizon of 5s (*i.e.*, $N = 10$). In the following experiments, we use the hierarchical NMPC as in the previous section, with a main task consisting on positioning the end effector and a secondary task to vertically align the arm CoG.

Fig. 7 shows a trajectory following experiment with the arm end effector during a 13 minutes flight. The control on both x and y directions work properly, however the vertical axis shows slower dynamic response, requiring finer thrust tuning.

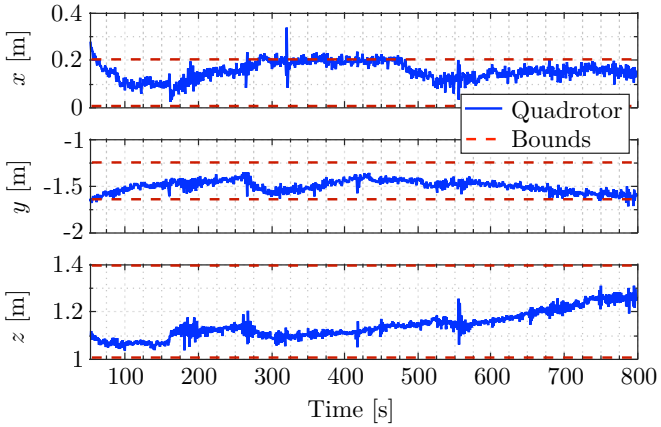


Fig. 9. Real platform positions during the same trajectory following task and hierarchical NMPC as in Fig. 7. The multirotor position (blue line) has been obtained using a motion capture system. The horizontal dashed lines correspond to lower and upper position bounds.

In these experiments, we also imposed a main constraint consisting on avoiding self-collisions between the arm end effector and the platform base. Fig 8 shows the positions of the joints, where it is clear how some joints reach their bounds but without overpassing them. Moreover, we added bounds in the solver for the platform positioning. The resulting multirotor movements are shown in Fig. 9.

VII. Conclusions

This paper presents the implementation of an NMPC controller for UAMs able to work online and on board a hardware with limited computational power. Several cost functions for UAMs are presented in order to accomplish a given task or to preserve stability. Among them, tasks to track trajectories with the arm end effector, improve the arm manipulability or align the arm CoG with the platform gravitational vector have been discussed. Two methods are presented to accomplish these tasks within simulation case studies. Firstly, a weighting strategy, then a hierarchical solution consisting on nested optimizations. The NMPC architecture has been validated through experiments with a real UAM.

References

[1] D. Zameroski, G. Starr, J. Wood, and R. Lumia, "Rapid swing-free transport of nonlinear payloads using dynamic programming," *J. of Dyn. Syst. Meas. and Cont.*, vol. 130, no. 4, p. 041001, 2008.

[2] I. Palunko, R. Fierro, and P. Cruz, "Trajectory generation for swing-free maneuvers of a quadrotor with suspended payload: A dynamic programming approach," in *Proc. IEEE Int. Conf. Robotics Autom.*, Saint Paul, May. 2012, pp. 2691–2697.

[3] K. Kondak, K. Krieger, A. Albu-Schaeffer, M. Schwarzbach, M. Laiacker, I. Maza, A. Rodriguez-Castano, and A. Ollero, "Closed-loop behavior of an autonomous helicopter equipped with a robotic arm for aerial manipulation tasks," *I. J. of Adv. Robotic Syst.*, vol. 10, 2013.

[4] V. Ghadiok, J. Goldin, and W. Ren, "Autonomous indoor aerial gripping using a quadrotor," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, San Francisco, Sep. 2011, pp. 4645–4651.

[5] A. Jimenez-Cano, J. Martin, G. Heredia, A. Ollero, and R. Cano, "Control of an aerial robot with multi-link arm for assembly tasks," in *Proc. IEEE Int. Conf. Robotics Autom.*, Karlsruhe, May. 2013, pp. 4916–4921.

[6] V. Lippiello and F. Ruggiero, "Exploiting redundancy in Cartesian impedance control of UAVs equipped with a robotic arm," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Vilamoura, Oct. 2012, pp. 3768–3773.

[7] G. Antonelli, E. Cataldi, P. Giordano, S. Chiaverini, and A. Franchi, "Experimental validation of a new adaptive control scheme for quadrotors MAVs," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Tokyo, Nov. 2013, pp. 2439–2444.

[8] M. Orsag, C. Korpela, M. Pekala, and P. Oh, "Stability control in aerial manipulation," in *Proc. Amer. Cont. Conf.*, Washington, Jun. 2013, pp. 5581–5586.

[9] M. Hehn and R. D'Andrea, "Quadrocopter trajectory generation and control," in *In Proc. 18th IFAC World Con.*, vol. 44, no. 1, 2011, pp. 1485–1491.

[10] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. IEEE Int. Conf. Robotics Autom.*, Shanghai, May. 2011, pp. 2520–2525.

[11] K. Sreenath, N. Michael, and V. Kumar, "Trajectory generation and control of a quadrotor with a cable-suspended load-a differentially-flat hybrid system," in *Proc. IEEE Int. Conf. Robotics Autom.*, Karlsruhe, May. 2013, pp. 4888–4895.

[12] M. Geisert and N. Mansard, "Trajectory generation for quadrotor based systems using numerical optimal control," in *Proc. IEEE Int. Conf. Robotics Autom.*, Stockholm, May. 2016, pp. 2958–2964.

[13] G. Garimella and M. Kobilarov, "Towards model-predictive control for aerial pick-and-place," in *Proc. IEEE Int. Conf. Robotics Autom.*, Seattle, May. 2015, pp. 4692–4697.

[14] S. Chiaverini, "Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators," *IEEE Trans. Robotics Autom.*, vol. 13, no. 3, pp. 398–410, Jun 1997.

[15] P. Baerlocher and R. Boulic, "Task-priority formulations for the kinematic control of highly redundant articulated structures," in *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, vol. 1, Oct 1998, pp. 323–329.

[16] A. Santamaria-Navarro, V. Lipiello, and J. Andrade-Cetto, "Task priority control for aerial manipulation," in *Proc. IEEE Int. Symp. Safe. Sec. Resc. Robotics.*, Toyako-cho, Oct. 2014, pp. 1–6.

[17] A. Santamaria-Navarro, P. Grosch, V. Lippiello, J. Sola, and J. Andrade-Cetto, "Uncalibrated visual servo for unmanned aerial manipulation," *IEEE/ASME Transactions on Mechatronics*, vol. PP, no. 99, pp. 1–1, 2017.

[18] V. Lippiello, J. Cacace, A. Santamaria-Navarro, J. Andrade-Cetto, M. . Trujillo, Y. R. Esteves, and A. Viguria, "Hybrid visual servoing with hierarchical task composition for aerial manipulation," *IEEE Rob. Autom. Letters*, vol. 1, no. 1, pp. 259–266, Jan. 2016.

[19] R. Rossi, A. Santamaria-Navarro, J. Andrade-Cetto, and P. Rocco, "Trajectory generation for unmanned aerial manipulators through quadratic programming," *IEEE Rob. Autom. Letters*, vol. 2, no. 2, pp. 389–396, Apr. 2017.

[20] A. Del Prete, F. Romano, L. Natale, G. Metta, G. Sandini, and F. Nori, "Prioritized optimal control," in *Proc. IEEE Int. Conf. Robotics Autom.*, Hong Kong, Jun. 2014, pp. 2540–2545.

[21] G. Antonelli, F. Arrichiello, S. Chiaverini, and P. R. Giordano, "Adaptive trajectory tracking for quadrotor mavs in presence of parameter uncertainties and external disturbances," in *IEEE/ASME Int. Conf. on Adv. Int. Mech.*, Wollongong, Jul. 2013, pp. 1337–1342.

[22] Y. E. Tlatelpa-Osorio, J. J. Corona-Sánchez, and H. Rodrguez-Cortés, "Quadrotor control based on an estimator of external forces and moments," in *Int. Conf. on Unman. Airc. Syst.*, Arlington, June 2016, pp. 957–963.

[23] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[24] Y. Zhang, D. Guo, K. Li, and J. Li, "Manipulability-maximizing self-motion planning and control of redundant manipulators with experimental validation," in *In Proc. IEEE Int. Conf. on Mech. and Autom.*, Chengdu, Aug 2012, pp. 1829–1834.

[25] B. Houska, H. J. Ferreau, and M. Diehl, "Acado toolkitan open-source framework for automatic control and dynamic optimization," *Opt. Cont. App. and Methods*, vol. 32, no. 3, pp. 298–312, 2011.

[26] D. Brescianini, M. Hehn, and R. D'Andrea, "Quadrocopter pole acrobatics," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Tokyo, Nov. 2013, pp. 3472–3479.