Enforcing Vision-Based Localization using Perception Constrained N-MPC for Multi-Rotor Aerial Vehicles

Martin Jacquet, Antonio Franchi

Enforcing Vision-Based Localization using Perception Constrained N-MPC for Multi-Rotor Aerial Vehicles

Martin Jacquet[1], Antonio Franchi[2,1]

***Abstract*— This work introduces a Nonlinear Model Predictive Control (N-MPC) for camera-equipped Unmanned Aerial Vehicles (UAVs), which controls at the motor level the UAV motion to ensure the quality of vision-based state estimation while performing other tasks. The controller ensures visibility over a sufficient amount of features, while optimizing their coverage, based on an assessment of the estimation quality. The controller works for the very broad class of generic multi-rotor UAVs, including platforms with any number of propellers, which can be both collinear, as in the quadrotor, and fixedly-tilted. The low-level inputs are computed in real-time and realistically constrained, in terms of maximum motor torque. This allows the platform to exploit its full actuation capabilities to maintain the visibility over the set of points of interest. Our implementation is tested in Gazebo simulations and in mocap-free real experiments, and features a visual-inertial state estimation based on Kalman filter. The software is provided open-source.**

I. Introduction

Unmanned Aerial Vehicles (UAVs) are increasingly used in a large range of applications, from aerial monitoring and exploration of vast regions, to work in high-risk and human-denied areas. In particular, monitoring and observation activities are increasingly being deployed, for instance for autonomous cinematography [1], or indoor building inspection [2].

Furthermore, the increasing efficiency, and decreasing weight of the available cameras and computation units allowed the deployment of efficient computer vision algorithms on UAVs [3], [4]. Indeed, most commercial UAVs are designed for photography, and many applications, such as autonomous drone racing, are enforcing the necessity of powerful and efficient embedded vision-processing algorithms. Such cameras, along with the appropriate software, are also powerful tools to fulfill the ego-localization of the UAV, which is of paramount importance for autonomous navigation, even more so in unstructured environments.

Onboard vision-based localization have been an active field of study for decades, resulting nowadays in very robust, efficient Visual-Inertial Odometry (VIO) or Simultaneous Localization and Mapping (SLAM) software [5], [6]. In addition, these softwares are proposed open-source and are actively maintained by their developers and communities.

[1]LAAS-CNRS, Université de Toulouse, CNRS, Toulouse, France, martin.jacquet@laas.fr, antonio.franchi@laas.fr
[2]Robotics and Mechatronics lab, Faculty of Electrical Engineering, Mathematics & Computer Science, University of Twente, Enschede, The Netherlands, a.franchi@utwente.nl
This work was partially funded by the ANR, under the project ANR-17-CE33-0007 'MuRoPhen', and by the European Commission project H2020 AERIAL-CORE (EC 871479).

However, exploiting a camera for ego-localization might conflict with the task imposed on the UAV. For instance, an exploration task or the transient phase toward a given destination might drive the UAV through a feature-poor area. Such event could compromise the quality of the pose estimation, leading to the inability to fulfill the task, or jeopardize the stability of the system. Such issues are prominent for collinear quadrotors, the most widely spread type of UAVs, due to their strong position/orientation coupling, which implies the necessity to tilt in order to achieve lateral motion (and, conversely, to move in order to tilt). Thus, the limited sensing domain of the UAVs can be altered when moving.

The study of perception-awareness in the intrinsic design of the UAV controller recently arose in the literature [7]–[10]. In these works, Nonlinear Model Predictive Controllers (N-MPC) are used to intertwine the control of the platform with the perception of a given object or phenomenon. This is achieved by exploiting the non-linear geometrical relation between the sensor measurement and the robot pose, thus allowing to predict, over a receding horizon, how the perception of the object of interest will evolve as the UAV moves.

However, these works are tackling the problem of active sensing or mobile object monitoring. Such approaches cannot be simply transposed to the monitoring of static landmarks for vision-based ego-localization, since all the tracked features are treated individually at all time during the trajectory. As a result, the multiple tracking would quickly become conflicting, as the UAV moves and new features are discovered. Indeed, the experiments presented in the aforementioned works rely on motion capture systems or external VIO software, rather than exploiting the perception-awareness of the controller to enforce the localization.

In [7], the N-MPC is given incentive to maintain the detected features from a VIO software close to the center of the Field of View (FoV) of the camera. Rather than considering the individual features, the controller tries to optimize the visibility over their barycenter. However, this barycenter is pre-computed outside of the N-MPC algorithm, which has no knowledge of the individual features' positions and thus cannot ensure their visibility. Therefore, it is implicitly assumed that enough feature points can be detected simply by optimizing visibility over their barycenter, i.e., that the features are dense and far from the camera. Such assumption might be proven wrong in, e.g., some SLAM scenarios or in exploratory tasks. Finally, the quality of the vision-based state estimation is not considered in the framework.

In [11], an N-MPC is employed to interlace control and

vision-based localization. The authors define a data-driven perception model and implement a chance constraint to produce valid observations. This approach is however path-specific since it requires a training to build the probabilistic model of the feature detection. Moreover, they make use of a gimbal-mounted stereo camera, which is not commonly found on all standard platforms. Finally, their controller relies on differential flatness to linearize the system dynamics, hence is neither generalized to UAVs with larger actuation span, nor can ensure that the planned motion is feasible by the platform real actuation which comes from the single motor torques.

In this work, we build upon our previous works [9], [12] to propose a new method that is able to control the real low-level input of the plaform to ensure the feasibility of vision-based localization and optimize its quality while performing additional tasks. Our main contribution is a N-MPC formulation that extends the various perception-related constraints and objectives used in the aforementioned works to this newly formulated problem. This method is designed for generic UAVs and computes in real-time the low-level inputs, namely the brushless motor torques. We also follow the approach in [13] in order to ensure that the computed control actions are feasible for the real platform and that the complete motion capability of the UAV is fully exploited. It can then be interfaced with any VIO or SLAM software that could provide the 3D pose of detected features.

In Sec. II we introduce the models of the UAV and camera. Then, we summarily present, in Sec. III, a basic SLAM algorithm for UAV state estimation, whose non-trivial aspects are recalled and detailed, before introducing in Sec. IV the new N-MPC formulation. Finally, the proposed framework is tested in experiments and simulations in Sec. V before concluding in Sec. VI.

II. Modeling

A. Generically Tilted Multirotor Dynamics

Similar to previous works, [9], [12], we consider the UAVs to be Generically-Tilted Multi-Rotors (GTMR). This model includes standard quadrotors, but goes beyond, e.g., hexarotors or tilted-propeller platforms.

We define the world inertial frame $\mathcal{F}_W$, with its origin O_W and its axes $\mathbf{x}_W, \mathbf{y}_W, \mathbf{z}_W$. The other frames are denoted using the same convention, e.g., $\mathcal{F}_B$ is the body frame of a robot, with its origin O_B and its axes $\mathbf{x}_B, \mathbf{y}_B, \mathbf{z}_B$. A GTMR is defined as a rigid body of mass m and inertia $\mathbf{J} \in \mathbb{R}^{3\times3}$, actuated by $n_p \geq 4$ propellers arbitrarily placed and oriented in $\mathcal{F}_B$. The position of O_B w.r.t. the $\mathcal{F}_W$ is denoted by ${}^W\mathbf{p}_B$ and the unit quaternion representing the orientation of $\mathcal{F}_B$ with respect to (w.r.t.) $\mathcal{F}_W$ is denoted by ${}^W\mathbf{q}_B$; and similarly for all the other frame pairs.

The robot state $\mathbf{x}$ is expressed as the concatenation of the body state $\mathbf{x}_b$ and actuator state $\mathbf{x}_a$, which are defined as

$$\mathbf{x}_b = [\mathbf{p}^\top \mathbf{q}^\top \mathbf{v}^\top \boldsymbol{\omega}^\top]^\top \in \mathbb{R}^{13}, \tag{1a}$$

$$\mathbf{x}_a = \boldsymbol{\gamma} \in \mathbb{R}^{n_p}, \tag{1b}$$

where, for conciseness, $\mathbf{p} = {}^W\mathbf{p}_B$, $\mathbf{q} = {}^W\mathbf{q}_B$, $\mathbf{v}$ is the velocity of O_B expressed in $\mathcal{F}_W$, $\boldsymbol{\omega}$ is the angular velocity of $\mathcal{F}_B$ w.r.t. $\mathcal{F}_W$, expressed in $\mathcal{F}_B$, and $\boldsymbol{\gamma}$ are the propeller thrusts.

The dynamic equations of a GTMR are then given by

$$\dot{\mathbf{p}} = \mathbf{v}, \tag{2a}$$

$$\dot{\mathbf{q}} = \frac{1}{2}\,\mathbf{q} \otimes \begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix}, \tag{2b}$$

$$\begin{bmatrix} m\ddot{\mathbf{p}} \\ \mathbf{J}\dot{\boldsymbol{\omega}} \end{bmatrix} = \begin{bmatrix} -mg\mathbf{z}_W \\ -\boldsymbol{\omega} \times \mathbf{J}\boldsymbol{\omega} \end{bmatrix} + \begin{bmatrix} \mathbf{q} \otimes \mathbf{G}_f\boldsymbol{\gamma} \otimes \mathbf{q}^* \\ \mathbf{G}_\tau\boldsymbol{\gamma} \end{bmatrix}, \tag{2c}$$

$$\dot{\boldsymbol{\gamma}} = \mathbf{u}, \tag{2d}$$

where $\otimes$ denotes the Hamilton product of two quaternions, $\mathbf{q}^*$ is the conjugate quaternion of $\mathbf{q}$, g is the intensity of the gravity acceleration. $\mathbf{G}_f$ and $\mathbf{G}_\tau \in \mathbb{R}^{3\times n_p}$ are respectively the *force* and *moment allocation matrices*, mapping the forces produced by each propeller to the total force and moment acting on the body. $\mathbf{u} \in \mathbb{R}^{n_p}$ are the system control inputs, and $\dot{\boldsymbol{\gamma}}$ are the time derivatives of the propeller thrusts, directly related to the torques applied to the brushless motors, see [13].

B. Camera Model

The camera is modeled as a punctual device rigidly attached to the GTMR body such that the pose transformation between the camera frame $\mathcal{F}_C$ and $\mathcal{F}_B$ is constant and known. The camera field of view (FoV) has a pyramidal shape centered around the principal axis $\mathbf{z}_C$, defined by two angles α_v and α_h.

The camera complies with the pinhole camera model, and its intrinsic calibration matrix $\mathbf{K}$ is known. Moreover, the camera images are assumed undistorted, avoiding reprojection errors in the peripheral FoV. The camera thus provides measurements $\mathbf{c} \in \mathbb{R}^2$ of 3D points ${}^C\mathbf{p} = [p_x\ p_y\ p_z]^\top$ that fall into its FoV, following

$$p_z\mathbf{c} = \mathbf{K}\ {}^C\mathbf{p} \tag{3}$$

These measurements are subject to an isotropic Gaussian noise of standard deviation σ_c.

III. Visual-Inertial Localization

In this section, we introduce a simple yet effective vision-based state estimation algorithm, which covers the minimum requirements needed by the proposed N-MPC control method. Notice that any VIO or SLAM algorithm could be used in replacement, provided the ability to retrieve the required data, i.e., the GTMR state and the position of the feature points detected by the camera.

For the sake of readability, in the following we denote the GTMR state variable as in Sec.II-A, even though they refer to estimated variables, hence are semantically different.

A. Error-state Extended Kalman Filter for SLAM

We use an Extended Kalman Filter (EKF) based SLAM approach, presented, e.g., in [14]. Both the robot state and the features are aggregated in the filter, and are jointly estimated in the inertial frame (be it $\mathcal{F}_W$ or $\mathcal{F}_B$ at time $t = 0$).

In particular, we make use of the so-called error-state approach of Kalman filtering [15]. This approach decouples the nominal state (large signal) and the error-state (small

signal, thus linearizable and integrable). It allows in particular to consider a minimal vectorial representation for the orientation error (e.g., angle-axis), while operating far from possible singularities.

The angular rates and linear accelerations are included in the filter, rather than used as inputs with estimated bias correction, as it is typically done in EKF-based SLAM [14]. We motivate this choice by the requirement of providing a filtered estimate of the angular velocities to the N-MPC, given the model presented in Sec.II-A. Hence, we make the assumption of pre-calibrated IMU [16].

For conciseness, the error-state EKF (ESEKF) prediction and update equations are not recalled hereafter. We refer to, e.g., [15] for details.

B. State Parametrization and Simplified Motion Model

We chose the inertial frame $\mathcal{F}_W$ to be coincident with the initial body frame. The KF state is defined as the concatenation of the GTMR state and feature states:

$$\mathbf{x}_{\text{KF}} = [{}^W\mathbf{p}_B^\top \ {}^W\mathbf{q}_B^\top \ {}^B\mathbf{v}_B^\top \ {}^B\boldsymbol{\omega}_B^\top \ {}^B\mathbf{a}_B^\top \ {}^W\mathbf{t}_1^\top \dots {}^W\mathbf{t}_n^\top]^\top, \quad (4)$$

where ${}^B\mathbf{a}_B$ is the acceleration of O_B, expressed in $\mathcal{F}_B$, n is the number of tracked features, and ${}^W\mathbf{t}_i$ is the state of the *i-th* feature. The parametrization of this pose will be discussed in Sec. III-C.

This state is propagated using a simplified motion model, i.e., assuming constant linear accelerations and angular velocities. This allows to reduce the non-linearity of the filtered process. Moreover, while it is important to consider the model non-linearities from the controller point of view, in order to to predict an accurate behavior of the system, the resulting motion can be instead observed as linearized, given a sufficiently small sampling time.

Then, the error state, whose variables are pre-fixed using δ, and for whose reference frames have been dropped for simplicity, is defined as

$$\delta\mathbf{x} = [\delta\mathbf{p}^\top \ \delta\boldsymbol{\theta}^\top \ \delta\mathbf{v}^\top \ \delta\boldsymbol{\omega}^\top \ \delta\mathbf{a}^\top \ \delta\mathbf{t}_1^\top \dots \delta\mathbf{t}_n^\top]^\top, \quad (5)$$

where $\delta\boldsymbol{\theta} = \text{Log}(\delta\mathbf{q})$ is the angular error associated to the orientation error $\delta\mathbf{q}$ [17]. We note that $\boldsymbol{\omega}$ and $\delta\boldsymbol{\theta}$ are expressed in the local frame $\mathcal{F}_B$, which has an impact on the filter equations, see [15]. Finally, the filter typically uses measurements from the camera to observe $\delta\mathbf{p}$, $\delta\boldsymbol{\theta}$ and $\delta\mathbf{t}_i$, and an IMU to observe $\delta\boldsymbol{\omega}$ and $\delta\mathbf{a}$. Additionally, other onboard sensors can be used to improve the estimation, e.g. a magnetometer or an altimeter.

C. Feature Representation and Measurements

The *i-th* observed feature state ${}^W\mathbf{t}_i$ can be defined in several ways [14]. The objective is to provide the N-MPC with a filtered estimate of the bearing vectors as well as the distances. The immediate parametrization is to consider the 6D pose of each feature independently. This allows the camera to observe jointly the feature and body poses. Since the features are assumed static in the inertial frame, they are less subject to process noise than the body state, hence they would not drift as the GTMR moves. Thus, we define

$${}^W\mathbf{t}_i = \left[{}^W\mathbf{p}_i^\top \ {}^W\mathbf{q}_i^\top\right]^\top \in \mathbb{R}^7. \quad (6)$$

Nevertheless, this approach has two main drawbacks. First, the state grows linearly with n, each new tracked feature adding 7 new state variables. This is counterbalanced by the small number of features required to retrieve with accuracy the body state. Second, the pinhole camera is not able to provide the measurements for depth and orientation. It implies that a geometrical a priori on the feature is needed as, e.g., in [18]. In practice, this is typically done using fiducial markers, such as Apriltags or AruCo [19], thanks to their reliability and practicality. Then, a pose estimation algorithm (e.g., [20]) allows to reconstruct the relative 6D transform between the marker and the camera, thus the body. Finally, the ESEKF measurement is ${}^B\mathbf{t}_i$.

In the following, we will refer to the fiducial markers indifferently as "features" and "markers".

In order to exploit such marker poses as measurements for the EKF, the measurement noise must be estimated. A constant Gaussian noise is not able to capture reliably the uncertainty, in particular for the orientation part, which is maximum when the marker is centered and orthogonal to the camera principal axis. Since the actual Gaussian measurement of the camera are the pixel positions of the corners of the marker, we chose to apply a first order propagation scheme to retrieve the 6D pose uncertainty $\Sigma_{\mathbf{t}_i}$ from σ_c, as in [9], [21]. This is achieved by computing in closed form, using (3), the Jacobian $\mathbf{J}_f$ of the inverse map

$$f\colon \ \mathbb{R}^3 \times SO(3) \ \to \mathbb{R}^8 \quad (7)$$

$$\left[{}^C\mathbf{p}^\top \, {}^C\boldsymbol{\theta}^\top\right]^\top \ \mapsto \left[\mathbf{c}_1^\top \ \mathbf{c}_2^\top \ \mathbf{c}_3^\top \ \mathbf{c}_4^\top\right]^\top, \quad (8)$$

where ${}^C\mathbf{p}$ and ${}^C\boldsymbol{\theta}$ are the position and orientation of the marker in camera frame, and $\mathbf{c}_i$ are the 4 corner pixel coordinates. Then, the 6D pose covariance $\Sigma_{\mathbf{p},\boldsymbol{\theta}}$ can be obtained through

$$\Sigma_{\mathbf{p},\boldsymbol{\theta}} = \sigma_c(\mathbf{J}_f^\top\mathbf{J}_f)^{-1}. \quad (9)$$

Finally, the measurement covariance $\Sigma_{\mathbf{t}_i}$ is obtained by further propagating $\Sigma_{\mathbf{p},\boldsymbol{\theta}}$ using the Jacobians of the roto-translation from $\mathcal{F}_C$ to $\mathcal{F}_B$. We note that $\Sigma_{\mathbf{t}_i} \in \mathbb{R}^{3\times3}$ contains the uncertainty of the orientation ${}^B\mathbf{q}_i$ expressed as an "orientation element" $\in SO(3)$ and not as quaternion $\in \mathbb{Q}$. In order to retrieve the covariances of the 4 individual quaternion elements, $\Sigma_{\mathbf{t}_i}$ needs to be transformed using the Jacobian of the Exponential map [17].

D. Non-additive Noise

Finally, another non-trivial aspect of this approach, which is not tackled in [15], is the use of non-vectorial measurements subject to non-additive noise, namely the quaternion ${}^B\mathbf{q}_i$. Indeed, the unit sphere of $\mathbb{Q}$ is not a vector space, hence not a closed set w.r.t. summation. Yet, being a group, it is stable by multiplication. The orientation measurement model, using a multiplicative noise, is given by

$$\mathbf{z}_q = \mathbf{h}(\mathbf{x}_{\text{KF}}) \otimes \mathbf{q}_v \in \mathbb{Q}, \quad (10)$$

where $\mathbf{h}(\mathbf{x}_{\text{KF}})$ is the observation function for the orientation, and $\mathbf{q}_v = \text{Exp}(\mathbf{v})$ is the measurement noise, where $\mathbf{v} \sim \mathcal{N}(0, \mathbf{R})$ is a Gaussian noise in its angle-axis representation, of covariance $\mathbf{R} \in \mathbb{R}^{3\times3}$.

Hence, the residual cannot be written as $\mathbf{res} = \mathbf{z}_q - \mathbf{h}(\mathbf{x}_{\text{KF}})$. Rather, to fit the EKF formalism, the residual on the manifold can be linearized, giving the observation model for $\delta\mathbf{x}$:

$$\mathbf{res} = \text{Log}(\mathbf{h}(\mathbf{x}_{\text{KF}})^{-1} \otimes \mathbf{z}_q) \tag{11}$$
$$\approx \mathbf{H}\delta\mathbf{x} + \mathbf{D}\mathbf{v}, \tag{12}$$

where $\mathbf{H} = \frac{\partial \mathbf{h}(\mathbf{x}_{\text{KF}})}{\partial \delta\mathbf{x}}\Big|_{\delta\mathbf{x}=0}$ and $\mathbf{D} = \frac{\partial \mathbf{z}_q}{\partial \mathbf{v}}\Big|_{\mathbf{v}=0}$ are the Jacobians used to propagate the state covariance.

A general formalism for ESEKF on manifolds, using multiplicative noise measurements and detailing the equations, can be found, e.g., in [22].

IV. Perception-Awareness in N-MPC

Assuming that an external VIO software provides the estimates of the GTMR state and the detected markers positions, the N-MPC can be designed to ensure that the visibility over these features is maintained while performing a given task. Additionally, some vision-related objectives can be included in the general Optimal Control Problem (OCP) to modulate the GTMR behavior according to vision-based metrics. This section first presents the perception constraints applied on the N-MPC, then details the various objectives integrated in the cost function. Finally, the overall OCP is formalized.

In the following, n will refer to the maximum number of tracked markers (fixed by the size of the filter state), while n_t will refer to the number of markers detected during the task. Markers detected after the n-*th* are ignored. Since the framework does not rely on any prior knowledge of the marker poses, and that the initial configuration of the system must allow to retrieve the robot pose, we have that $0 < n_t \leq n$.

A. Perception Constraints

In order to perform a reliable state estimation, the GTMR must always maintain visibility on a certain amount of markers while performing its task. If not, the pose estimate would rapidly drift due to the integration of noisy IMU measurements, and consequently jeopardize the system stability.

Previous works [8], [9] included visibility coverage constraints for each individual tag. This is expressed, for each feature i, as a pair of constraints on the position of the feature position in the image plane (vertical and horizontal):

$$|{}^c x_i / {}^c z_i| \leq \tan\frac{\alpha_h}{2} = H, \quad |{}^c y_i / {}^c z_i| \leq \tan\frac{\alpha_v}{2} = V. \tag{13}$$

However, it is important to let the robot move freely without being constrained to keep the visibility at each instant over a specific set of markers. Hence, we propose to integrate the constraint that "at least $\underline{n} > 1$ marker(s) should be visible at each instant", which translates as the inequality constraint

$$\underline{n} < \sum_i \xi_i = \xi \leq n, \tag{14a}$$

$$\xi_i = \begin{cases} 1 \text{ if the marker } i \text{ is inside the FoV} \\ 0 \text{ otherwise} \end{cases}, \tag{14b}$$

where ξ_i expresses the Boolean value associated with the pair of inequalities (13). This value is a function of the state, hence it can be propagated over the receding horizon.

Nonetheless, these Boolean values cannot be integrated in the N-MPC as they are, since they would include a discontinuity preventing any gradient-based optimization. This is circumvented by approximating (13) by a product of two sigmoid functions:

$$\xi_i \approx \frac{1}{1+e^{-\lambda(H-|\frac{c\,x_i}{c\,z_i}|)}} \times \frac{1}{1+e^{-\lambda(V-|\frac{c\,y_i}{c\,z_i}|)}}, \tag{15}$$

where $\lambda \gg 1$ defines the steepness of the transition. This approximation needs to be tuned such that the transition is not too steep to avoid numerical instability, but steep enough such that the constraint (14) does not get fictitiously unsatisfied. In practice, $\lambda = 100$ provides a good compromise.

We note that the state estimator from Sec. III provides the N-MPC with an estimate of the positions of all the n_t known markers. These are considered in the computation of ξ and exploited by the algorithm, as the loss of visibility over a feature could be compensated by the recovery of others.

B. Perception Objectives

1) Barycenter of Bearing Vectors: Additionally, a classical approach in perception-aware N-MPC is to include the perception-related tasks directly in the cost function [8], [9]. Such objective is typically to maintain the features close to center of the image, to increase the reaction margin to external disturbances, while driving the GTMR away from configurations where the constraints (13) are active.

In the formalism introduced in [9], maintaining a feature close to the image center is efficiently expressed as the maximization of the cosine angle between the bearing vector and the principal axis. This quantity, denoted with $\text{c}\beta_i$, can be efficiently computed in closed form as a function of the state variables.

However, considering the markers separately is meaningful only if they should be all observed individually. For the sake of localization, there is no reason to optimize visibility over some specific markers if this opposes the requested motion task, and if some others are also available. Therefore, similarly to [7], we decide to consider the barycenter of all detected features, whose angle w.r.t. $\mathbf{z}_C$ has to be minimized. This can be equivalently rephrased, using the triangle inequality, as the minimization of

$$\frac{1}{n_t}\sum_i (1 - \text{c}\beta_i). \tag{16}$$

2) Weighting the Barycenter: Considering all the n_t tags for the computation of the barycenter is not desirable, since it would give an incentive to move the camera FoV away from the markers that are currently detected, and are thus actively used for the state estimation. Furthermore, in the cost function, it is desirable to increase the importance of the markers which are providing the best measurements, since in turn they contribute to a better precision in the GTMR state estimation.

These two aspects can be addressed by assigning a weight λ_i for each marker in (16). This weight should be designed in

such a way that it tends to 1 as the marker estimation quality improves, and rapidly decreases to 0 when the marker leaves the FoV, thus providing no benefit to the estimation process.

A good metric to assess the quality of the estimation retrieved from a Kalman filter is the volume of the ellipsoid associated with the covariance matrix, denoted with $\mathbf{P}$. This volume can be computed using either the determinant or approximated, at a smaller computation cost, using the trace of $\mathbf{P}$ [12]. We denote the trace of the covariance matrix concerning the *i-th* tag by $\mathrm{tr}(\mathbf{P})_i$.

We remark that the diagonal of a covariance matrix is the sum of squared terms, hence $\mathrm{tr}(\mathbf{P})_i > 0$. Thus, a suitable expression for λ_i is

$$\lambda_i = e^{-\mu.\mathrm{tr}(\mathbf{P})_i} \in]0, 1[\,, \tag{17}$$

where $\mu > 0$ arbitrarily defines the decreasing rate.

We note that, contrary to [12], since the N-MPC does not include any internal representation of the EKF uncertainty, $\mathrm{tr}(\mathbf{P})_i$ cannot be written as function of the state, hence cannot be propagated. This quantity is rather to be computed at the initialization of each control cycle, then kept constant over the receding horizon. It still provides an accurate assessment of the estimation quality of each marker at a given instant, but cannot be used to predict the effect of the GTMR motion on this quality. To adopt a paradigm similar to [12], the uncertainty (9) of the measurements that the UAV would produce while moving through the horizon could be used to weight the features, at the cost of a higher computational load.

The λ_i being constant over the horizon, the normalization of the barycenter (16) can be omitted in the minimization problem. Hence, the perception objective is defined as

$$C_{\text{vision}} = \sum_i \lambda_i (1 - \mathrm{c}\beta_i). \tag{18}$$

C. Optimal Control Problem

As per [9], we consider realistic physical limitations of the motors in the N-MPC. Such limitations are equivalently recast as constraints on $\boldsymbol{\gamma}$ and $\dot{\boldsymbol{\gamma}}$, whose lower and upper bounds can be obtained through an identification campaign, as shown in [13].

The GTMR is assigned with a motion task, defined by an output map $\mathbf{y} = \mathbf{h}(\mathbf{x})$ and its reference $\mathbf{y}_r$. The latter is provided by, e.g., an external motion planner, as a trajectory for the full receding horizon, in position, attitude, and their first- and second-order derivatives.

The motion objective $C^{\mathbf{W}}_{\text{motion}}$ is defined as the squared distance[1] between $\mathbf{y}$ and $\mathbf{y}_r$, weighted by a diagonal matrix $\mathbf{W}$. The perception objective is weighted by a scalar w.

Finally, the discrete-time OCP, over the receding horizon T, sampled in N shooting points, at a given instant t, is expressed as

$$\min_{\substack{\mathbf{x}_0 \dots \mathbf{x}_N \\ \mathbf{u}_0 \dots \mathbf{u}_{N-1}}} \sum_{k=0}^{N} C^{\mathbf{W}}_{\text{motion},k} + w \sum_{k=0}^{N} C_{\text{vision},k} \tag{19a}$$

[1]We consider the geodesic distance on the unit quaternion manifold for the quaternion orientation [23].

Fig. 1: Snapshot of the platform used in the reported experiment.

$$s.t. \quad \mathbf{x}_0 = \mathbf{x}(t) \tag{19b}$$
$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{p}_k), \qquad k \in \{0, N-1\} \tag{19c}$$
$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k), \qquad k \in \{0, N\} \tag{19d}$$
$$\underline{\mathbf{x}_b} \leq \mathbf{m}(\mathbf{x}_{b,k}, \mathbf{p}_k) \leq \overline{\mathbf{x}_b}, \qquad k \in \{0, N\} \tag{19e}$$
$$\underline{\boldsymbol{\gamma}} \leq \boldsymbol{\gamma}_k \leq \overline{\boldsymbol{\gamma}}, \qquad k \in \{0, N\} \tag{19f}$$
$$\underline{\dot{\boldsymbol{\gamma}}_k} \leq \mathbf{u}_k \leq \overline{\dot{\boldsymbol{\gamma}}_k}, \qquad k \in \{0, N-1\} \tag{19g}$$
$$\underline{n} \leq \xi_k \leq n, \qquad k \in \{0, N\} \tag{19h}$$

where $\mathbf{x}(t)$ is the measurement of the current state and $\mathbf{f}$ synthetically denotes the system dynamics, expressed in (2). Possible motion constraints on the body state $\mathbf{x}_b$ are expressed using a selection function $\mathbf{m}$. The lower and upper bounds of the inequality constraints are denoted using respectively $\underline{\cdot}$ and $\overline{\cdot}$. Finally, $\mathbf{p}_k$ are the external parameters passed to the N-MPC, namely the feature estimated poses from the ESEKF.

V. Validation

A. Experimental and Simulation Setup

The framework is implemented in C++, using GenoM [24] which is a middleware-independent component generator, that can be compiled for a given middleware, e.g., ROS. The N-MPC implementation is the one introduced in [9], based on [25]. It uses a 4^{th}-order explicit Runge-Kutta integrator and implements a Real-Time Iteration solving scheme. The simulated hardware interface and path planning are done using the TeleKyb3 software, available on the OpenRobots platform[2]. The software framework is either connected to the flight controller, or to a Gazebo simulated system that emulates the actual platform interface. Details on how to use this software can be found in the provided git repository[3].

The GTMR is equipped with a grayscale monocular camera. In Gazebo, we use the embedded camera simulator, with an horizontal FoV $\alpha_h = 2\,\text{rad}$, and the aspect ratio $\alpha_h/\alpha_v = \frac{4}{3}$. On the actual platform, we use an Intel RealSense T265, rectified and undistorted to achieve a FoV of $\frac{\pi}{2}$ rad, with aspect ratio 1. We use this tracking device for its convenience, however it is only exploited as a standard monocular camera. The camera frequency is set to 60Hz,

[2]https://git.openrobots.org/projects/telekyb3
[3]https://redmine.laas.fr/projects/N-MPC-localization

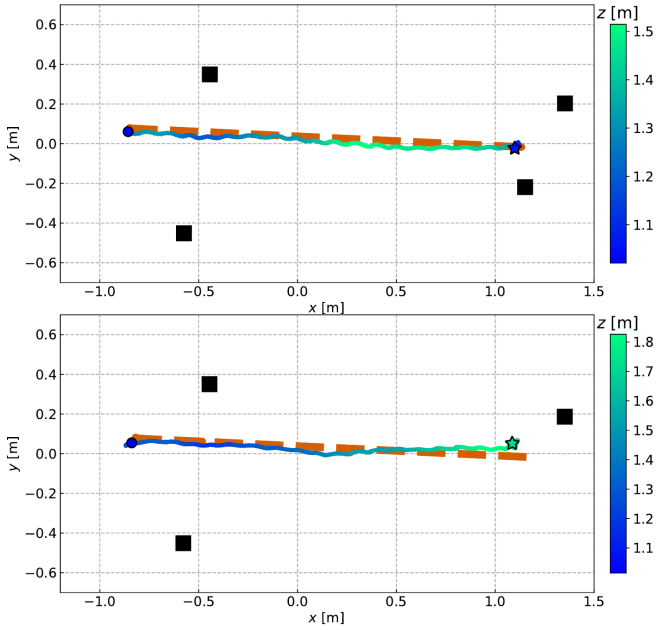

Fig. 2: Experiments with a quadrotor and a down-facing camera. The (x, y) trajectory, with z denoted by the color gradient, following the dashed orange reference. The circle and star are the initial and final positions, while the black squares are the feature (x, y) positions ($z = 0$).

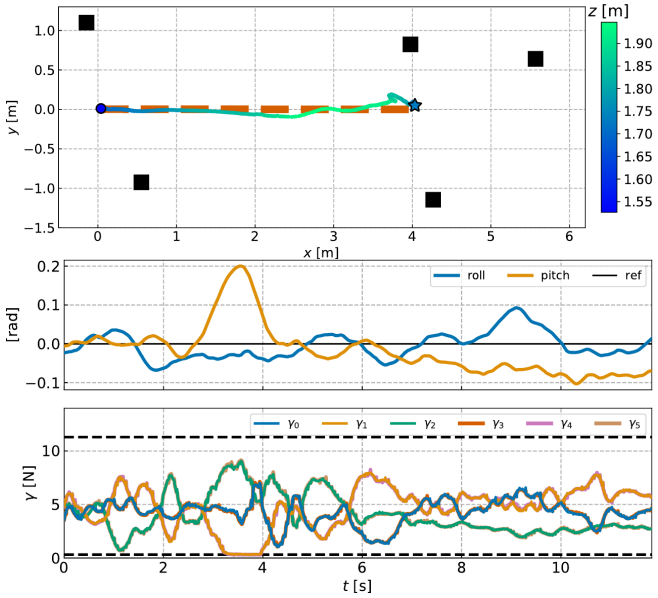

Fig. 3: Simulation with a tilted-propeller hexarotor and a down-facing camera. Above, the (x, y) trajectory using the same presentation as Fig. 2. Below, the roll/pitch angles of the tilted-propeller hexarotor, as well as the 6 propeller thrusts, over time.

and the image processing takes about 3ms on CPU, using OpenCV.

The state estimation of the GTMR is achieved using the algorithm presented in Sec. III fusing the visual marker measurements and the inertial data from an IMU whose frequency is 500Hz. No motion capture is used in any way. A Gaussian noise is applied on each of these inertial measurements, of respective standard deviations 0.02rad/s and 0.1m/s^2. The ESEKF frequency is set to 1kHz. In the reported experiments, the N-MPC average computation time is 5.2ms with a standard deviation of 1.9ms on the Intel NUC onboard computer, using an horizon of 2s, sampled in 25 shooting points.

Videos of all the reported experiments and simulations, can be found in the attached multimedia file.

B. Motion with Visibility Constraints

This section presents the overall behavior induced by the N-MPC formulation (19) when a motion task is requested. Although the observation of 1 marker is sufficient for the ego-localization, we impose $\underline{n} = 2$ to add redundancy and increase the resilience to, e.g., false detections.

1) Experiments with a quadrotor: In these experiments, the GTMR is a collinear quadrotor equipped with a down-facing camera, pictured in Fig. 1, in order to underline the capability of the N-MPC to exploit the non-linear position/attitude coupling for perception.

The (x, y) desired and actual trajectories are reported in Fig. 2. The GTMR is commanded to move forward of about 2 m at constant height $z = 1$ m. As the distance to the left side markers increases, the quadrotor flies upwards to maintain the visibility. In the upper graph, as the right-hand side tags enter the FoV, the quadrotor starts to descend, maintaining visibility over the newly detected features.

In the experiment reported in the lower graph, as there is only one marker visible from the final position, the quadrotor does not reach the final position in altitude, but hovers at a higher altitude enforcing (19h).

2) Simulation with a tilted-propeller hexarotor: A similar simulation performed with a tilted-propeller hexarotor tasked to move forward by 4 m with z =1.5 m is presented in Fig. 3. The hexarotor achieves a better trajectory tracking than the quadrotor, exploiting at best its larger actuation span both during the transient and final hovering phases. In the lower graph, we can see (around 3.5 s) that the N-MPC forcibly reaches the lower bound for 2 propellers while tilting independently from translating, in order to maintain the visibility. Finally, the hexarotor pitches in order to maintain visibility over the newly detected markers, while staying in the desired position.

3) Motion along a longer path: The previously reported results illustrate the capability of the N-MPC to modulate its trajectory to ensure the feasibility of the localization. In this subsection, we show how C_{vision} allows to mitigate a given reference to follow a feature-full path.

We propose a simulation where the GTMR follows a 16 meters long path, at constant height. As reported in Fig. 4, the markers are placed aside from the main path, in such a way that the quadrotor modulates its (x, y) trajectory w.r.t. its reference, in to maintain visibility. The hexarotor, however, exploits its larger actuation and the ability to tilt independently from the translation to achieve a better tracking of the reference trajectory.

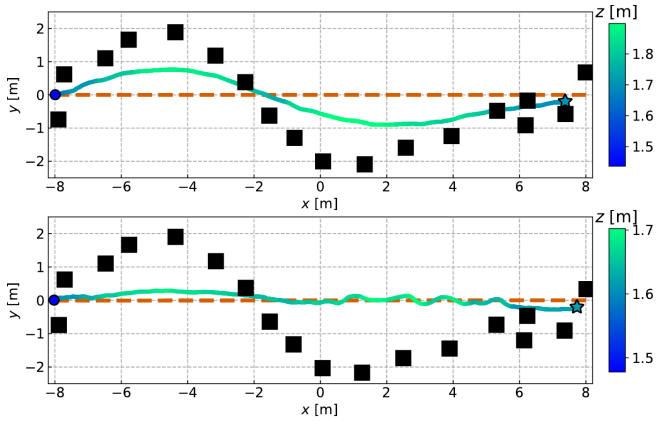

Fig. 4: Simulations using a quadrotor (top) and a tilted-propeller hexarotor (bottom). The same presentation as Fig. 2 is used. Both N-MPC are tuned with the same set of weights.

VI. Conclusion

In this work, we have introduced a perception-aware N-MPC able to enforce vision-based ego-localization of the GTMR while performing additional tasks. We build upon previous works to formalize a suited set of constraints and objectives for this application. We also included in our framework an assessment of the quality of the feature detection, in order to generate motion such that the overall state estimation is improved. The controller considers the full non-linear dynamics of the GTMR, and imposes realistic constraints of the low-level actuation. We presented the behavior of the controller with two GTMR designs in Gazebo simulations, performing ESEKF-SLAM with monocular camera measurements and inertial data, on which we imposed a Gaussian noise. Finally the framework was tested on a real quadrotor, to show its capability to control a real platform in a motion tracking task, while satisfying the visibility requirements.

This work could be extended further with the integration of a more advanced existing VIO software, allowing the removal of fiducial markers and a larger scalability w.r.t. the amount of features considered. Moreover, an exploratory behavior could be induced in the cost function through the discovery of new markers, whilst enforcing constraints on the currently detected ones to ensure the state estimation. Such exploratory behavior could be enabled by the introduction of a probabilistic distribution of markers, e.g. with a rasterization of the environment in cells which may or may not contain feature. Finally, a better perception model for the markers could be defined, accounting for their orientation and possible occlusions.

References

[1] A. Alcántara, J. Capitán, A. Torres-González, R. Cunha, and A. Ollero, "Autonomous execution of cinematographic shots with multiple drones," *IEEE Access*, vol. 8, pp. 201 300–201 316, 2020.

[2] P. Petracek, V. Kratky, and M. Saska, "Dronument: System for reliable deployment of micro aerial vehicles in dark areas of large historical monuments," *IEEE Robotics and Automation Letters*, vol. 5, pp. 2078–2085, Apr. 2020.

[3] P. Zhang, Y. Zhong, and X. Li, "Slimyolov3: Narrower, faster and better for real-time uav applications," in *2020 IEEE/CVF Int. Conf. on Computer Vision*, 2019.

[4] Y. Akbari, N. Almaadeed, S. Al-maadeed, and O. Elharrouss, "Applications, databases and open computer vision research from drone videos and images: a survey," *Artificial Intelligence Review*, vol. 54, no. 5, pp. 3887–3938, 2021.

[5] J. Delmerico and D. Scaramuzza, "A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots," in *2018 IEEE Int. Conf. on Robotics and Automation*, 2018, pp. 2502–2509.

[6] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Trans. on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[7] D. Falanga, P. Foehn, P. Lu, and D. Scaramuzza, "PAMPC: Perception-aware model predictive control for quadrotors," in *2018 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Oct. 2018.

[8] B. Penin, P. R. Giordano, and F. Chaumette, "Minimum-time trajectory planning under intermittent measurements," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 153–160, 2018.

[9] M. Jacquet and A. Franchi, "Motor and perception constrained NMPC for torque-controlled generic aerial vehicles," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 518–525, Apr. 2021.

[10] G. Li, A. Tunchez, and G. Loianno, "PCMPC: Perception-constrained model predictive control for quadrotors with suspended loads using a single camera and IMU," in *2021 IEEE Int. Conf. on Robotics and Automation*, Jun. 2021.

[11] M. Greeff, T. D. Barfoot, and A. P. Schoellig, "A perception-aware flatness-based model predictive controller for fast vision-based multi-rotor flight," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 9412–9419, 2020.

[12] M. Jacquet, M. Kivits, H. Das, and A. Franchi, "Motor-level N-MPC for cooperative active perception with multiple heterogeneous UAVs," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2063–2070, Apr. 2022.

[13] D. Bicego, J. Mazzetto, R. Carli, M. Farina, and A. Franchi, "Nonlinear model predictive control with enhanced actuator model for multi-rotor aerial vehicles with generic designs," *Journal of Intelligent & Robotics Systems*, vol. 100, no. 3, pp. 1213–1247, 2020.

[14] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct ekf-based approach," in *2015 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2015, pp. 298–304.

[15] J. Solà, "Quaternion kinematics for the error-state kalman filter," Tech. Rep., 2017.

[16] D. Tedaldi, A. Pretto, and E. Menegatti, "A robust and easy to implement method for IMU calibration without external equipments," in *2014 IEEE Int. Conf. on Robotics and Automation*, 2014, pp. 3042–3049.

[17] J. Solà, J. Deray, and D. Atchuthan, "A micro Lie theory for state estimation in robotics," Tech. Rep., 2018.

[18] J. Thomas, J. Welde, G. Loianno, K. Daniilidis, and V. Kumar, "Autonomous flight for detection, localization, and tracking of moving targets with a small quadrotor," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1762–1769, 2017.

[19] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.

[20] T. Collins and A. Bartoli, "Infinitesimal plane-based pose estimation," *International journal of computer vision*, vol. 109, no. 3, pp. 252–286, 2014.

[21] M. Fourmy, D. Atchuthan, N. Mansard, J. Solà, and T. Flayols, "Absolute humanoid localization and mapping based on imu lie group and fiducial markers," in *2019 IEEE-RAS 19th International Conference on Humanoid Robots*, 2019, pp. 237–243.

[22] D. He, W. Xu, and F. Zhang, "Kalman filters on differentiable manifolds," *arXiv preprint*, 2021.

[23] D. Q. Huynh, "Metrics for 3D rotations: Comparison and analysis," *Journal of Mathematical Imaging and Vision*, vol. 35, no. 2, pp. 155–164, 2009.

[24] A. Mallet, C. Pasteur, M. Herrb, S. Lemaignan, and F. Ingrand, "GenoM3: Building middleware-independent robotic components," in *2010 IEEE Int. Conf. on Robotics and Automation*, 2010, pp. 4627–4632.

[25] Y. Chen, M. Bruschetta, E. Picotti, and A. Beghi, "MATMPC - a MATLAB based toolbox for real-time nonlinear model predictive control," in *2019 European Control Conference*, Jun. 2019, pp. 3365–3370.