PAMPC: Perception-Aware Model Predictive Control for Quadrotors

Davide Falanga*, Philipp Foehn*, Peng Lu, and Davide Scaramuzza

Abstract— We present the first perception-aware model predictive control framework for quadrotors that unifies control and planning with respect to action and perception objectives. Our framework leverages numerical optimization to compute trajectories that satisfy the system dynamics and require control inputs within the limits of the platform. Simultaneously, it optimizes perception objectives for robust and reliable sensing by maximizing the visibility of a point of interest and minimizing its velocity in the image plane. Considering both perception and action objectives for motion planning and control is challenging due to the possible conflicts arising from their respective requirements. For example, for a quadrotor to track a reference trajectory, it needs to rotate to align its thrust with the direction of the desired acceleration. However, the perception objective might require to minimize such rotation to maximize the visibility of a point of interest. A model-based optimization framework, able to consider both perception and action objectives and couple them through the system dynamics, is therefore necessary. Our perception-aware model predictive control framework works in a receding-horizon fashion by iteratively solving a non-linear optimization problem. It is capable of running in real-time, fully onboard our lightweight, small-scale quadrotor using a low-power ARM computer, together with a visual-inertial odometry pipeline. We validate our approach in experiments demonstrating (I) the conflict between perception and action objectives, and (II) improved behavior in extremely challenging lighting conditions.

Fig. 1: An example application of our PAMPC, where a quadrotor is asked to fly at $3\,\mathrm{m/s}$ around a region of interest while keeping it visible in the field of view of its camera.

SUPPLEMENTARY MATERIAL

Video: `https://youtu.be/9vaj829vE18`
Code: `https://github.com/uzh-rpg/rpg_mpc`

I. INTRODUCTION

Thanks to the progresses in perception algorithms, the availability of low-cost cameras, and the increased computational power of small-scale computers, vision-based perception has recently emerged as the de facto standard in onboard sensing for micro aerial vehicles. This made it possible to replicate some of the impressive quadrotor maneuvers seen in the last decade [1], [2], [3], [4], which relied on motion-capture systems, using only onboard sensing, such as cameras and IMUs [5], [6], [7].

Cameras have a number of advantages over other sensors in terms of weight, cost, size, power consumption and field of view. However, vision-based perception has severe limitations: it can be intermittent and its accuracy is strongly affected by both the environment (e.g., texture distribution, light conditions) and motion of the robot (e.g., motion blur, camera pointing direction, distance from the scene).

This means that one cannot always replace motion-capture systems with onboard vision, since the motion of a camera can negatively affect the quality of the estimation, posing hard bounds on the agility of the robot. On the other hand, perception can benefit from the robot motion if it is planned considering the necessities and the limitations of onboard vision. For example, to pass through a narrow gap while localizing with respect to it using an onboard camera, it is necessary to guarantee that the gap is visible at all times. Similarly, to navigate through an unknown environment, it is necessary to guarantee that the camera always points towards texture-rich regions.

To fully leverage the agility of autonomous quadrotors, it is necessary to create synergy between perception and action by considering them jointly as a single problem.

A. Contributions

Model Predictive Control (MPC) has become increasingly popular for quadrotor control [8], [9], [10] thanks to its capability of simultaneously dealing with different constraints and objectives through optimization. In this work, we present an MPC algorithm for quadrotors able to optimize both action and perception objectives.

Our framework satisfies the robot dynamics and computes feasible trajectories with respect to the input saturations. Such trajectories are not constrained to specific time or space parametrization (e.g., polynomials in time or splines), and tightly couple perception and action. To do so, perception objectives aimed at rendering vision-based estimation more robust are taken into account in the optimization problem. Such objectives are the visibility of a point of interest the robot needs to maintain in the image, and the minimization of the velocity of its projection onto the image plane. The main challenge in this is to simultaneously cope with action (e.g., dynamics, underactuation, saturations) and perception

objectives, due to the potential conflicts between them.

To solve this problem, we leverage numerical optimization to compute trajectories that are optimal with respect to a cost function considering both the dynamics of the robot and the quality of perception. To fully exploit the agility of a quadrotor, we incorporate perception objectives into the optimization problem not as constraints, but rather as components to be optimized. This results in a perception-aware framework which is intrinsically tailored to agile navigation, since the optimizer can trade off between perception and action objectives (cf. Fig. 1, depicting fast circle flight while adjusting the heading to look at a point of interest). Furthermore, considering perception in the cost function reduces the computation load of the model predictive control pipeline, allowing it to run in real-time on a low-power onboard computer. Our approach does not depend on the task and can potentially provide benefits to a large variety of applications, such as vision-based localization, target tracking, visual servoing, and obstacle detection. We validate our perception-aware model predictive control framework in real-world experiments using a small-scale, lightweight quadrotor platform.

B. Related Work

The aforementioned shift from offboard to onboard sensing based on cameras resulted in an increased number of works trying to connect perception and action.

In [11], the authors proposed a method to compute minimum-time trajectories that take into account the limited field of view of a camera to guarantee visibility of points of interests. Such a method requires the trajectory to be parametrized as a B-spline polynomial, constraining the kind of motion the robot can perform. Also, perception is included in the planning problem as hard constraint, posing an upper-bound to the agility of the robot since such constraints must be satisfied at all times. Furthermore, the velocity of the projection of the points of interest in the image is not taken into account. Finally, the algorithm was not suited for real-time control of a quadrotor, and was only tested in simulations..

In [12], the authors focused on combining visual servoing with active Structure from Motion and proposed a solution to modify the trajectory of a camera in order to increase the quality of the reconstruction. In such a work, a trajectory for the tracked features in the image plane was required, and the null space of the visual servoing task was exploited in order to render it possible for such feature to track the desired trajectory. Furthermore, the authors did not consider the underactuation of the robot, which can significantly lower the performance of the overall task due to potentially conflicting dynamics and perception objectives.

In [13] and [14], information gain was used to bridge the gap between perception and action. In the first work, the authors tackled the problem of selecting trajectories that minimize the pose uncertainty by driving the robot toward regions rich of texture. In the second work, a technique to minimize the uncertainty of a dense 3D reconstruction based on the scene appearance was proposed. In both works, however, near-hover quadrotor flight was considered, and the underactation of the platform was not taken into account.

In [15], a hybrid visual servoing technique for differentially flat systems was presented. A polynomial parameterization of the flat outputs of the system was required, and due to the computational load required by the designed optimization framework, an optimal trajectory was computed in advance and never replanned. This did not allow coping with external disturbances and unmodelled dynamics, which during the execution of the trajectory can lead to behaviours different from the expected one.

In [16] and [17], a real-time motion planning method for aerial videography was presented. In these works, the main goal was to optimize the viewpoint of a pan-tilt camera carried by an aerial robot in order to improve the quality of the video recordings. Both works were mainly targeted to cinematography, therefore they considered objectives such as the size of a target of interest and its visibility. Conversely, we target robotic sensing and consider objectives aimed at facilitating vision-based perception.

In [18], the authors proposed a two-step approach for target-aware visual navigation. First, position-based visual servoing was exploited to find a trajectory minimizing the reprojection error of a landmark of interest. Then, a model predictive control pipeline was used to track such a trajectory. Conversely, we solve the trajectory optimization and tracking within a single framework. Additionally, that work only aimed at rendering the target visible, but did not take into account that, due to the motion of the camera, it might not be detectable because of motion blur. We cope with this problem by considering in the optimization problem the velocity of the projection of the point of interest in the image plane.

C. Structure of the Paper

The remainder of this paper is organized as follows. In Sec. II we provide the general formulation of the problem. In Sec. III we derive the model for the dynamics of the projection of a 3D point into the image plane for the case of a quadrotor equipped with a camera. In Sec. IV we present our perception-aware optimization framework, describing the objectives and the constraints it takes into account. In Sec. V we validate our approach in different real-world experiments showcasing the capability of our framework. In Sec. VI we discuss our approach and provide additional insights and in Sec. VII we draw the conclusions.

II. PROBLEM FORMULATION

For truly autonomous robot navigation, two components are essential: (I) perception, both of the ego-motion and of the surrounding environment; (II) action, meant as the combination of motion planning and control algorithms. A very wide literature is available for both of them. However, they are rarely considered as a joint problem.

The need for coupled perception and action can be easily explained. To guarantee safety, accurate and robust perception is necessary. Nevertheless, the quality of vision-based

perception is strongly affected by the motion of the camera. On the one hand, it can degrade its performance by not making it possible to extract sufficiently accurate information from images. For example, lack of texture or blur due to camera motion can lead to algorithm failure. On the other, the quality of vision-based perception can improve significantly if its limitations and requirements are considered, e.g. by rendering highly-textured areas visible in the image and by reducing motion blur. Therefore it is necessary to create synergy between perception and action.

Let $\mathbf{x}$ and $\mathbf{u}$ be the state and input vectors of a robot, respectively. Assume its dynamics to be described by a set of differential equations $\dot{\mathbf{x}} = \boldsymbol{f}(\mathbf{x}, \mathbf{u})$. Furthermore, let $\mathbf{z}$ be the state vector of the perception system (e.g., 3D points' projection onto the image plane), and σ a vector of parameters characterizing it (e.g., the focal length of the camera or its field of view). The perception state and the robot state are coupled through the robot dynamics, namely $\mathbf{z} = \boldsymbol{f}_p(\mathbf{x}, \mathbf{u}, \sigma)$. Given certain action objectives, we can define an action cost $\mathcal{L}_a(\mathbf{x}, \mathbf{u})$. Similarly, we can define a cost $\mathcal{L}_p(\mathbf{z})$ for the perception objectives.

We can then formulate the coupling of perception and action as an optimization problem:

$$\min_{\mathbf{u}} \quad \int_{t_0}^{t_f} \mathcal{L}_a(\mathbf{x}, \mathbf{u}) + \mathcal{L}_p(\mathbf{z})\, dt$$
$$\text{subject to} \qquad \mathbf{r}(\mathbf{x}, \mathbf{u}, \mathbf{z}) = 0 \qquad (1)$$
$$\mathbf{h}(\mathbf{x}, \mathbf{u}, \mathbf{z}) \leq 0,$$

where $\mathbf{r}(\mathbf{x}, \mathbf{u}, \mathbf{z})$ and $\mathbf{h}(\mathbf{x}, \mathbf{u}, \mathbf{z})$ represent equality and inequality constraints that the solution should satisfy for perception, action, or both of them simultaneously.

III. METHODOLOGY

Any computer vision algorithm aimed at providing a robot with the information necessary for navigation (e.g., pose estimation, obstacle detection, etc) has two fundamental requirements. First, the points of interest used by the algorithm to provide the aforementioned information must be visible in the image. For example, such points can be the landmarks used for pose estimation by visual odometry algorithms, or the points belonging to an object for obstacle detection. If such points are not visible while the robot is moving, there is no way the algorithm can cope with the absence of information. Second, such points of interest must be clearly recognizable in the image. Depending on the motion of the camera and the distance from the scene, the projection of a 3D point onto an image can suffer from motion blur, making it very complicated, if not impossible, to extract meaningful information. Therefore, the motion of the camera should be thoroughly planned to guarantee robust visual perception.

Based on the considerations above, in this work we consider two perception objectives in our framework: (I) visibility of points of interest, and (II) minimization of the velocity of their projection onto the image plane. In the following, we study the relation between the motion of a quadrotor equipped with an onboard camera and the projection onto the image plane of a point in space. Without loss of generality,
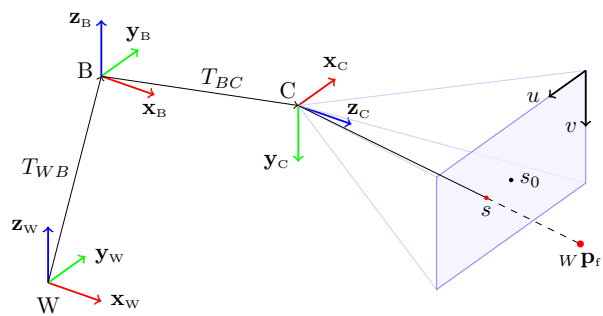

Fig. 2: A schematics representing the world frame W, the body frame B and the camera frame C. The position and orientation of B with respect to W is provided by T_{WB}. The constant rigid body transformation T_{BC} provides the extrinsics of the camera. A feature located at $_W\mathbf{p}_f$ is projected into the image plane onto a point of coordinates $\mathbf{s}$. $\mathbf{s}_0$ represents the principal point.

we consider the case of a single 3D point of interest. Our goal is to couple perception and action into an optimization framework by expressing the dynamics of its projection onto the image plane as a function of the state and input vectors of a quadrotor.

A. Nomenclature

In this work, we make use of a world frame W with orthonormal basis $\{\mathbf{x}_W, \mathbf{y}_W, \mathbf{z}_W\}$. The quadrotor frame B, also referred to as the body frame, has orthonormal basis $\{\mathbf{x}_B, \mathbf{y}_B, \mathbf{z}_B\}$. Finally, we assume the robot to be equipped with a camera, whose reference frame C has orthonormal basis $\{\mathbf{x}_C, \mathbf{y}_C, \mathbf{z}_C\}$. Fig. 2 provides a clear overview about the reference frames.

Throughout this manuscript, we represent vectors as bold quantities having a prefix, representing the frame in which they are expressed, and a suffix, indicating the origin and the end of such a vector. For example, the quantity $_W\mathbf{p}_{WB}$ represents the position of the body frame B with respect to the world frame W, expressed in the world frame. To simplify the notation, if a vector has no prefix, we assume it to be expressed in the first frame reported in the suffix (i.e., the frame where the vectors origin is).

We use quaternions to represent the orientation of a rigid body. The time derivative of a quaternion $\mathbf{q} = (q_w, q_x, q_y, q_z)^\mathsf{T}$ is given by $\dot{\mathbf{q}} = \frac{1}{2}\Lambda(\omega) \cdot \mathbf{q}$, where the skew-symmetric matrix $\Lambda(\omega)$ of a vector $\omega = (\omega_x, \omega_y, \omega_z)^\mathsf{T}$ is defined as:

$$\Lambda(\omega) = \begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{bmatrix}. \qquad (2)$$

Finally, we use the operator $\odot$ to denote the multiplication between a quaternion and a vector. More specifically, multiplying a vector $\mathbf{v}$ with the quaternion $\mathbf{q}$ means rotating $\mathbf{v}$ by the rotation induced by $\mathbf{q}$. By doing so, we obtain a vector

$\mathbf{v}' = \mathbf{v} \odot \mathbf{q} = Q\mathbf{v}$ where:

$$Q = \begin{bmatrix} 1 - 2q_y^2 - 2q_z^2 & 2(q_x q_y + q_w q_z) & 2(q_x q_z - q_w q_y) \\ 2(q_x q_y - q_w q_z) & 1 - 2q_x^2 - 2q_z^2 & 2(q_y q_z + q_w q_x) \\ 2(q_x q_z + q_w q_y) & 2(q_y q_z - q_w q_x) & 1 - 2q_x^2 - 2q_y^2 \end{bmatrix}.$$

B. Quadrotor Dynamics

Let $\mathbf{p}_{WB} = (p_x, p_y, p_z)^\mathsf{T}$ and $\mathbf{q}_{WB} = (q_w, q_x, q_y, q_z)^\mathsf{T}$ be the position and the orientation of the body frame with respect to the world frame W, expressed in world frame, respectively (cf. Fig. 2). Additionally, let $\mathbf{v}_{WB} = (v_x, v_y, v_z)^\mathsf{T}$ be the linear velocity of the body, expressed in world frame, and $\mathbf{\Omega}_B = (\omega_x, \omega_y, \omega_z)^\mathsf{T}$ its angular velocity, expressed in the body frame. Finally, let $\mathbf{c} = (0, 0, c)^\mathsf{T}$ be the mass-normalized thrust vector, where $c = (f_1 + f_2 + f_3 + f_4)/m$, f_i is the thrust produced by the i-th motor, and m is the mass of the vehicle. In this work, we use the dynamical model of a quadrotor proposed in [3]:

$$\begin{aligned} \dot{\mathbf{p}}_{WB} &= \mathbf{v}_{WB} \\ \dot{\mathbf{v}}_{WB} &=_W \mathbf{g} + \mathbf{q}_{WB} \odot \mathbf{c} \\ \dot{\mathbf{q}}_{WB} &= \frac{1}{2}\Lambda(\mathbf{\Omega}_B) \cdot \mathbf{q}_{WB} \end{aligned} \tag{3}$$

where $_W\mathbf{g} = (0, 0, -g)^\mathsf{T}$ is the gravity vector, with $g = 9.81\,\mathrm{m/s}^2$. The state and the input vectors of the system are $\mathbf{x} = [\mathbf{p}_{WB}, \mathbf{v}_{WB}, \mathbf{q}_{WB}]^\mathsf{T}$ and $\mathbf{u} = [c, \mathbf{\Omega}_B^\mathsf{T}]^\mathsf{T}$, respectively.

C. Perception Objectives

Let $_W\mathbf{p}_f = (_W p_{fx}, _W p_{fy}, _W p_{fz})$ be the 3D position of a point of interest (landmark) in the world frame W (cf. Fig. 2). We assume the body to be equipped with a camera having extrinsic parameters described by a constant rigid body transformation $T_{BC} = [\mathbf{p}_{BC}, \mathbf{q}_{BC}]$, where $\mathbf{p}_{BC}$ and $\mathbf{q}_{BC}$ are the position and the orientation of C with respect to B. The coordinates $_C\mathbf{p}_f = (_C p_{fx}, _C p_{fy}, _C p_{fz})^\mathsf{T}$ of $_W\mathbf{p}_f$ in the camera frame C are given by:

$$\begin{aligned} _C\mathbf{p}_f = (\mathbf{q}_{WB}\ \mathbf{q}_{BC})^{-1} \odot \\ (_W\mathbf{p}_f - (\mathbf{q}_{WB} \odot\ \mathbf{p}_{BC} + \mathbf{p}_{WB})). \end{aligned} \tag{4}$$

The point $_C\mathbf{p}_f$ in camera frame is projected into the image plane coordinates $\mathbf{s} = (u, v)^\mathsf{T}$ according to classical pinhole camera model [19]:

$$u = f_x \frac{_C p_{fx}}{_C p_{fz}}, \quad v = f_y \frac{_C p_{fy}}{_C p_{fz}} \tag{5}$$

where f_x, f_y are the focal lengths for pixel rows and columns, respectively.

To guarantee robust vision-based perception, the projection $\mathbf{s}$ of a point of interest $_W\mathbf{p}_f$ should be as close as possible to the center of the image for two reasons. First, keeping its projection in the center of the image results in the highest safety margins against external disturbances. The second reason comes from the fact that the periphery of the image is typically characterized by a non-negligible distortion, especially for large field of view cameras. A number of models for such distortion are available in the literature, as well as techniques to estimate their parameters to compensate the effects of the distortion. However, such

a compensation is never perfect and this can degrade the accuracy of the estimates.

As previously mentioned, in addition to rendering the point of interest visible in the image, we are interested in reducing the velocity of its projection onto the image plane. We assume the point of interest to be static, but similar considerations apply to the case where such a point of interest moves with respect to the world frame. To express the projection velocity as a function of the quadrotor state and input vectors, we can differentiate (5) with respect to time:

$$\begin{aligned} \dot{u} &= f_x \frac{_C\dot{p}_{fx}\ _Cp_{fz} -\ _Cp_{fx}\ _C\dot{p}_{fz}}{_Cp_{fz}^2}, \\ \dot{v} &= f_y \frac{_C\dot{p}_{fy}\ _Cp_{fz} -\ _Cp_{fy}\ _C\dot{p}_{fz}}{_Cp_{fz}^2}. \end{aligned} \tag{6}$$

Eq. (6) can be written in a compact form as:

$$\dot{\mathbf{s}} = \begin{bmatrix} \dot{u} \\ \dot{v} \\ 0 \end{bmatrix} = \begin{bmatrix} 0 & -\frac{f_x}{_Cp_{fz}^2} & 0 \\ \frac{f_y}{_Cp_{fz}^2} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} (_C\mathbf{p}_f \times_C \dot{\mathbf{p}}_f). \tag{7}$$

To compute the term $_C\dot{\mathbf{p}}_f$, we can differentiate (4) with respect to time:

$$_C\dot{\mathbf{p}}_f = -\frac{1}{2}\Lambda(\mathbf{\Omega}_C)\ _C\mathbf{p}_f -_C \mathbf{v}_{WC}, \tag{8}$$

where:

$$\begin{aligned} _C\mathbf{v}_{WC} &= (\mathbf{q}_{WB}\ \mathbf{q}_{BC})^{-1} \odot \\ &\quad \left(\frac{1}{2}\Lambda(\mathbf{\Omega}_B)\mathbf{q}_{WB} \odot \mathbf{p}_{BC} + \mathbf{v}_{WB}\right), \\ \mathbf{\Omega}_C &= \mathbf{q}_{BC}^{-1} \odot \mathbf{\Omega}_B. \end{aligned} \tag{9}$$

D. Action Objectives

For a quadrotor to execute a desired task (e.g., reach a target position in space), a suitable trajectory has to be planned. In this regard, for a quadrotor two objectives should be considered.

The first comes from the bounded inputs available to the system. The thrust each motor can produce has both an upper and a lower bound, leading to a limited input vector $\mathbf{u}$. Therefore, denoting the subset of the allowed inputs as $\mathcal{U}$, the planned trajectory should be such that the condition $\mathbf{u}(t) \in \mathcal{U}\ \forall t$ can be satisfied.

The second objective to be considered comes from the underactuated nature of a quadrotor. In the most common configuration, all the rotors point in the same direction, typically along the axis $\mathbf{z}_B$ of the body. This means that the robot can accelerate only in this direction. Therefore, to move in the 3D space, it is necessary to exploit the system dynamics (3) by coupling the translational and the rotational motions of the robot to follow the desired trajectory.

E. Challenges

The perception (Sec. III-C) and the action (Sec. III-D) objectives previously described are both necessary for vision-based quadrotor navigation. Considering them simultaneously is challenging due to the possible conflict among them.

Indeed, for a quadrotor to track a reference trajectory, it needs to rotate to align its thrust with the direction of the desired acceleration. However, the perception objective might require to minimize such rotation to maximize the visibility of a point of interest. A model-based optimization framework able to consider both perception and action objectives and couple them through the system dynamics is therefore necessary.

IV. MODEL PREDICTIVE CONTROL

Formulating coupled perception and action as an optimization problem has the advantages of being able to satisfy the underactuated system dynamics and actuator constraints (i.e., input boundaries) and to minimize the predicted costs along a time horizon. In contrast, classical control schemes are incapable of predicting costs and the corresponding trajectory (e.g., PID controllers) and guaranteeing input boundaries (PID, LQR).

The basic formulation of such an optimization is given in (1), which in our case results in a non-linear program with quadratic costs. This can then be approximated by a sequential quadratic program (SQP) where the solution of the non-linear program is iteratively approximated and used as a model predictive control (MPC). To this regard, for the MPC to be effective, the optimization scheme has to run in real-time, at the desired control frequency. To achieve this, we first discretize the system dynamics with a time step dt for a time horizon t_h into $\mathbf{x}_i \; \forall i \in [1, N]$ and $\mathbf{u}_i \; \forall i \in [1, N-1]$. We define the time-varying state cost matrix as $\mathcal{Q}_{x,i} \; \forall i \in [1, N]$. Furthermore, the time-varying perception and input cost matrices are defined as $\mathcal{Q}_{p,i}$ and $\mathcal{R}_i$, $\forall i \in [1, N-1]$, respectively. Finally, let $\mathbf{z} = [\mathbf{s}, \dot{\mathbf{s}}]$ be the perception function. It is important to recall that $\mathbf{z}$ is a function of the quadrotor's state and input variables, as remarked in Eq. (4) to (9). The resulting cost function we consider is:

$$
\begin{aligned}
\mathcal{L} = \; & \bar{\mathbf{x}}_N^{\mathsf{T}} \mathcal{Q}_{x,N} \bar{\mathbf{x}}_N + \\
& + \sum_{i=1}^{N-1} \left(\bar{\mathbf{x}}_i^{\mathsf{T}} \mathcal{Q}_{x,i} \bar{\mathbf{x}}_i + \bar{\mathbf{z}}_i^{\mathsf{T}} \mathcal{Q}_{p,i} \bar{\mathbf{z}}_i + \bar{\mathbf{u}}_i^{\mathsf{T}} \mathcal{R}_i \bar{\mathbf{u}}_i \right),
\end{aligned}
\tag{10}
$$

where the values $\bar{\mathbf{x}}, \bar{\mathbf{z}}, \bar{\mathbf{u}}$ refer to the difference with respect to the reference of each value. In our case, the reference value for $\mathbf{z}$ is the null vector (i.e., center of the image and zero velocity) and the reference for the states and inputs are given by a target pose or a precomputed trajectory (that neglects the perception objectives).

The inputs $\mathbf{u}$, consisting of c and $\mathbf{\Omega}_B$, as well as the velocity $\mathbf{v}_{WB}$ are limited by the constraints:

$$
c_{min} \leq c \leq c_{max}, \tag{11}
$$
$$
-\Omega_{max} \leq \mathbf{\Omega}_B \leq \Omega_{max}, \tag{12}
$$
$$
-v_{max} \leq \mathbf{v}_{WB} \leq v_{max}, \tag{13}
$$

where $c_{min}, c_{max}, \Omega_{max}, v_{max} \in \mathcal{R}_+$.

To include the dynamics as in (3), we use multiple shooting as transcription method and a Runge-Kutta integration
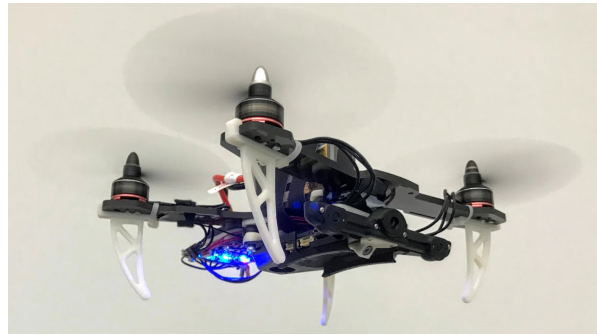

Fig. 3: The quadrotor used for the experiments.

scheme. We refer the reader to [20] and [21] for more details on the transcription of the dynamics for optimization.

We approximate the solution of the optimization problem by executing one iteration at each control loop and use as initial state the most recent available estimate $\mathbf{x}_{est}$ provided by a Visual-Inertial Odometry pipeline running onboard the vehicle (see Sec. V-A). To achieve good approximations, it is important to run these iterations significantly faster than the discretization time of the problem and to keep the previous solution as initialization trajectory of the next optimization. Such a SQP scheme leads to a fast convergence towards the exact solution, since the system is always close to the last linearization, and the deviation of each state $\mathbf{x}_i$ between two iterations is very small.

V. EXPERIMENTS

In order to show the potential of our perception-aware model predictive control, we ran our approach onboard a small, vision-based, autonomous quadrotor. We refer the reader to the attached video showcasing the experiments.

A. Experimental Setup

We used a small and lightweight quadrotor platform to achieve high agility through high torque-to-inertia and thrust-to-weight ratios, and improve simplicity and safety for the user (cf. Fig 3). The quadrotor had a take-off weight of $420\,\mathrm{g}$, a thrust-to-weight ratio of ~ 2, and a motor-to-motor diagonal of $220\,\mathrm{mm}$. We used a Qualcomm Snapdragon Flight board with a quad-core ARM processor at up to $2.26\,\mathrm{GHz}$ and $2\,\mathrm{GB}$ of RAM, paired with a Qualcomm Snapdragon Flight ESC. The board was equipped with an Inertial Measurement Unit and a forward-looking, wide field-of-view global-shutter camera tilted down by $45°$ for visual-inertial odometry (VIO) using the Qualcomm mvSDK. It ran ROS on Linux and our self-developed flight stack. We setup the optimization with ACADO and used qpOASES as solver. As discretization step, we chose $dt = 0.1\,\mathrm{s}$ with a time horizon of $t_h = 2\,\mathrm{s}$ and ran one iteration step in each control loop with a frequency of $100\,\mathrm{Hz}$. Therefore, the iteration ran roughly $10\times$ faster than the discretization time, resulting in small deviations of the predicted state vector between iterations and facilitating convergence. The code developed in this work is publicly available as open-source software.
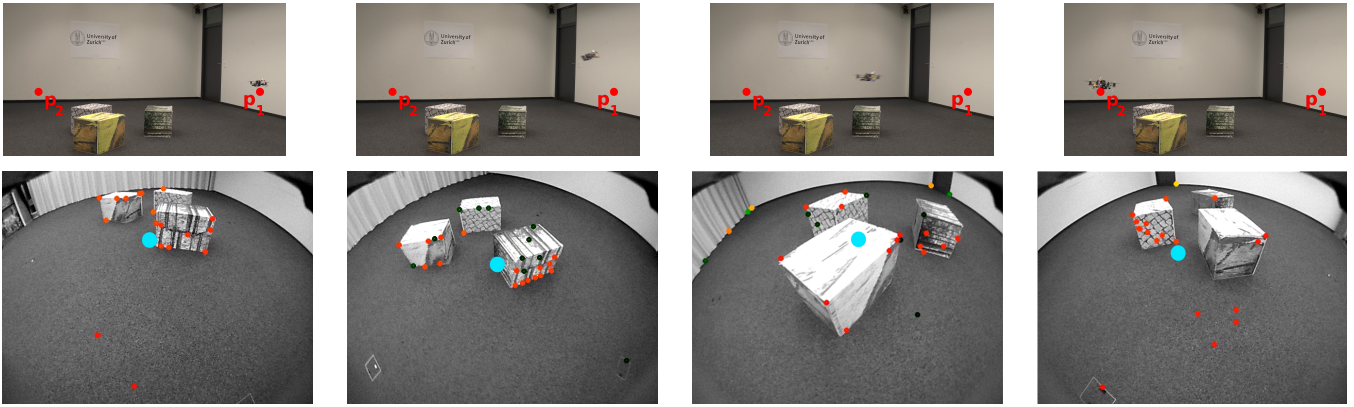
Fig. 4: A sequence of the visibility experiment for the hover-to-hover flight experiment, with time progressing from left to right. The quadrotor performs a maneuver to fly to a new reference pose, exploiting additional height to pitch less and keep the point of interest (centroid of the vision features, marked as cyan circle) in the center of the image. The corresponding footage is available in the accompanying video.
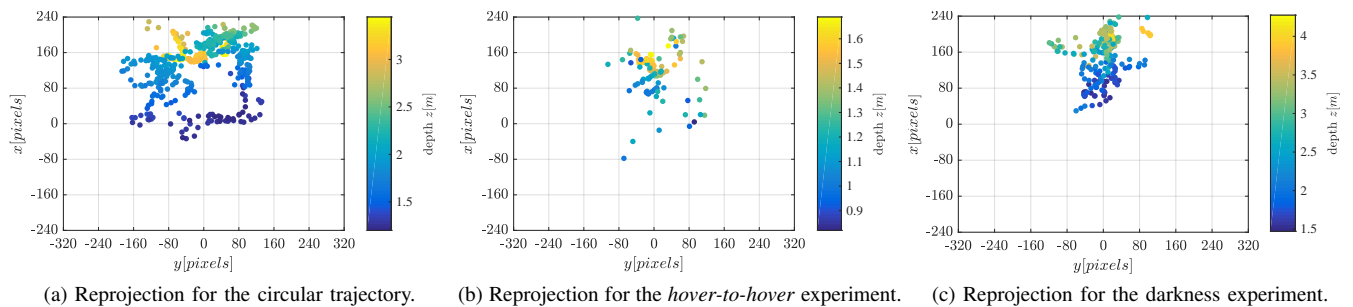

(a) Reprojection for the circular trajectory. (b) Reprojection for the *hover-to-hover* experiment. (c) Reprojection for the darkness experiment.

Fig. 5: Reprojection of the point of interest in image plane, colored according to the depth with respect to the camera frame.

B. Experiment Description and Results

To prove the functionality and importance of our PAMPC, we ran three experiments. In the first experiment, the controller modified a circular trajectory to improve the visibility of a point of interest. In the second experiment, the controller handled *hover-to-hover* flight by deviating from a straight line trajectory to keep the point of interest visible. In the third experiment, it enabled vision-based flight in an extremely challenging scenario. All the experiments were conducted with onboard VIO and onboard computation of the PAMPC, without any offline computation and without any motion-capture system.

1) Circular Flight: We setup a small pile of boxes in the middle of a room otherwise poor of texture. We did this to force the VIO pipeline to use such boxes as features for state estimation. The centroid of these features was set as our point of interest. We provided the robot with a circular reference trajectory around the aforementioned boxes and asked it to fly along such a trajectory while maintaining the boxes visible in the center of the image (cf. Fig. 1). We evaluated the performance of our framework for speeds along the circle from $1\,\mathrm{m/s}$ to $3\,\mathrm{m/s}$.

The results of one run of the circular flight experiments at $3\,\mathrm{m/s}$ are depicted in Fig. 6. Despite the agility of the maneuver, which requires large deviations from the hover conditions, the robot is able to keep the point of interest

visible in the onboard image. Fig. 5a reports the reprojection in the image plane for such point of interest.

2) Hover-To-Hover Flight: In this experiment within the same scenario as in Sec. V-B.1, we showed the capabilities of our framework for *hover-to-hover* flight. More specifically, we requested a pose jump from a position $\mathbf{p}_1$ to $\mathbf{p}_2$ at equal height (cf. Fig. 4). During that maneuver, the quadrotor had to pitch down to reach the desired acceleration, but controversially should pitch as little as possible to keep the point of interest visible. A sequence of this experiment is visible in Fig. 4.

One can easily see that, despite the start and end positions are at the same height, the quadrotor not only pitches to go towards the new reference in an horizontal motion, but also accelerates upward (i.e., in positive z, cf. Fig. 7). This results in a smaller pitch angle and a higher thrust to reach the same y-acceleration, which is helpful for perception since it brings the features towards the center of the frame due to the higher altitude. If perception objectives were not considered, the resulting trajectory would have not required any height change, potentially leading to a poor visibility of the point of interest. The full motion of the quadrotor is depicted in Fig. 7, where the exploitation of the added height and the orientation of the camera frame can be seen. Finally, in Fig. 5b we show the reprojection in the image plane for the point of interest.
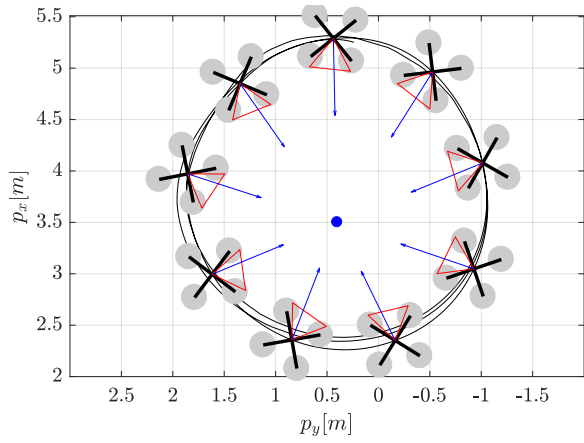
Fig. 6: Executed trajectory with quadrotor heading while the arrow points toward the point of interest (blue).
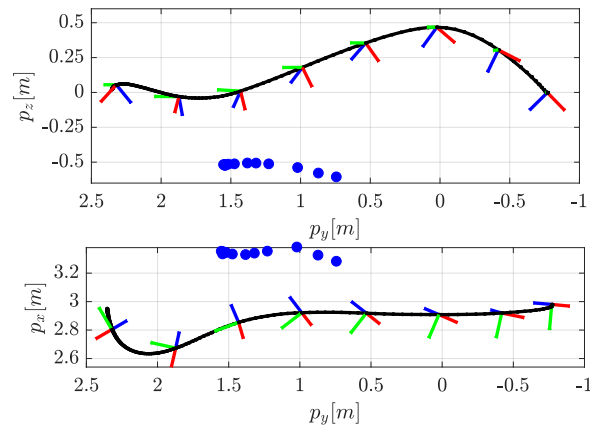

Fig. 7: Quadrotor path in hover-to-hover, looking towards the centroid of tracked features (blue), with the camera frame indicated by $\{x_C, y_C, z_C\}$.
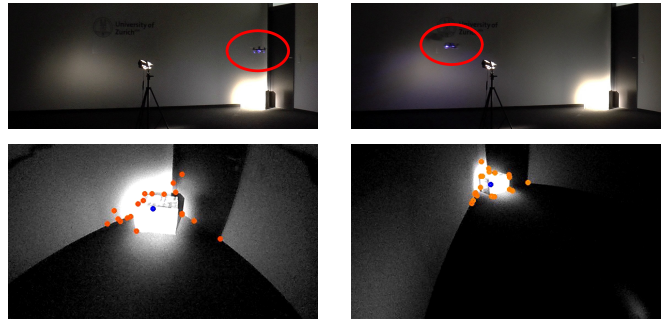

Fig. 8: A sequence of the darkness experiment with time progressing from left to right. The quadrotor, highlighted by a red circle in the figures in the first row, tracks a trajectory and adjusts its heading to keep the point of interest (centroid of the vision features) in field of view.
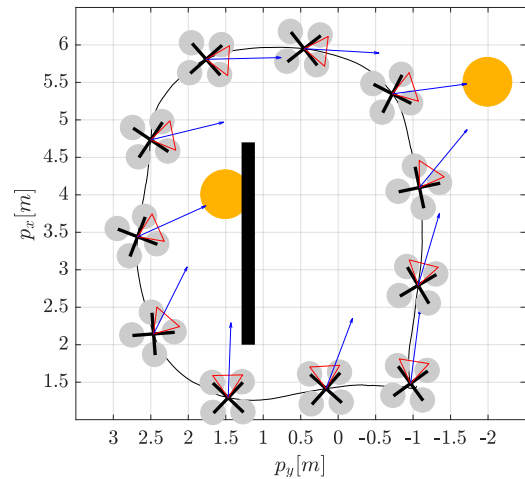

Fig. 9: Path of the quadrotor, looking towards light spots (yellow), with camera direction (red triangle) and point-of-interest direction (blue arrow).

3) Darkness Scenario: This experiment was targeted towards extremely challenging scenarios, such as flight in a very dark environment, or otherwise difficult illumination conditions (cf. Fig. 8). To demonstrate the performance in such a scenario, we flew the vehicle several times in a dark room with two illuminated spots. If the illuminated spots left the field of view for a moment, the VIO pipeline would drift quickly or even completely loose track, potentially leading to a crash. Therefore, in such scenarios it was of immense importance to keep the few available features always visible. The flown path was given by a trajectory passing through four waypoints forming a rectangle, but without any heading reference. The quadrotor correctly adjusts its heading to keep the illuminated spot in its field of view, because this is the only source of trackable features. Fig. 9 visualizes a setup with two spotlights and a cardboard wall in between, where the quadrotor first focuses on the upper right illuminated spot, and further down the track switches to the second illuminated spot behind the wall. The reprojection of the point of interest in the image plane is shown in Fig. 5c.

VI. DISCUSSION

A. Choice of the optimizer

To implement the optimization problem, we chose to use ACADO because of two main reasons: (I) it is capable of transcribing system dynamics with single- and multiple-shooting and integration schemes, as well as provide an interface to a solver; (II) it generates c++ code, which then is compiled directly on the executing platform, which allows it to use accelerators and optimizations tailored to the platform.

B. Convexity of the problem

Our state and input space is a convex domain, hence also any quadratic cost in those is convex. The perception costs could be argued to be non-convex due to the division by $_Cp_{fz}$ in the projection (5). However, on examination of the projection one will notice that the denominator $_Cp_{fz}$ is always positive, since the pinhole camera projection model does not allow negative or zero depths. We can therefore constrain $_Cp_{fz}$ to be positive, rendering all possible solutions in the positive halfplane $\mathcal{R}^+$ and therefore recover convexity.

C. Choice of point of interest

In our experiments, we used the centroid of detected features as our point of interest. Assuming that all the features are equally important, instead of optimizing for each individually, we can summarize them as their centroid, which results in the same optimal solution.

D. PAMPC Parameters

We chose a discretization of $dt = 0.1\,\text{s}$ and a time horizon of $t_h = 2\,\text{s}$. One could always argue that a longer time horizon and a shorter discretization step are beneficial, but they also increase the computation time by roughly $\mathcal{O}(N^2)$ with the number of discretization nodes $N = \frac{t_h}{dt}$. In our experience, we could not identify any significant gain from smaller discretization steps nor from a longer time horizon.

E. Computation Time

Since the computation time must be low enough to execute the optimization in a real time scheme, we show that it is significantly lower than the one required by the controller frequency of $100\,\text{Hz}$. Indeed, our PAMPC requires on average $3.53\,\text{ms}$. It is interesting to note that this is the case for both an idle CPU and while running the full pipeline with VIO and our full control pipeline. This is due to the quad-core ARM CPU and the fact that our full pipeline without the PAMPC takes up only 3 cores leaving one free for the PAMPC. However, the standard deviation increases significantly if the CPU is under load (from $0.155\,\text{ms}$ to $0.354\,\text{ms}$), even though the maximal execution time always stays below $5\,\text{ms}$.

F. Drawbacks of a Two-Step Approach

An alternative approach to the problem tackled in this work is to use the differential flatness as in [2] to plan a translational trajectory connecting the start and end positions, and subsequentially plan the yaw angle to point the camera towards the point of interest. After planning, a suitable controller could be used to track the desired reference trajectory. Although possible, such a solution would lead to sub-optimal results because of the following reasons: (I) the roll and pitch angles of the quadrotor would be planned without considering the visibility objective, therefore might render the point of interest not visible in the image despite the yaw control; (II) because of the split between planning and control, even if the first would provide guarantees about visibility, these could not be preserved during the control stage due to deviations from the nominal trajectory; (III) it would be challenging to provide guarantees about the respect of the input saturations. Therefore, our proposed approach considering perception, planning and control as a single problem leads to superior results.

VII. Conclusions

In this work, we presented a perception-aware model predictive control (PAMPC) algorithm for quadrotors able to optimize both action and perception objectives. Our framework computes trajectories that satisfy the system dynamics and inputs limits of the platform. Additionally, it optimizes perception objectives by maximizing the visibility of a point of interest in the image and minimizing the velocity of its projection into the image plane for robust and reliable sensing. To fully exploit the agility of a quadrotor, we incorporated perception objectives into the optimization problem not as constraints, but rather as components in the cost function to be optimized. Our algorithm is able to run in real-time on an onboard ARM processor, in parallel with a VIO pipeline, and is used to directly control the robot. We validated our approach in real-world experiments using a small-scale, lightweight quadrotor platform.

References

[1] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," in *Int. Symp. Experimental Robotics*, 2010.

[2] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *IEEE Int. Conf. Robot. Autom.*, 2011.

[3] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient motion primitive for quadrocopter trajectory generation," *IEEE Trans. Robot.*, vol. 31, no. 6, 2015.

[4] D. Brescianini, M. Hehn, and R. D'Andrea, "Quadrocopter pole acrobatics," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2013.

[5] D. Falanga, E. Mueggler, M. Faessler, and D. Scaramuzza, "Aggressive quadrotor flight through narrow gaps with onboard sensing and computing," in *IEEE Int. Conf. Robot. Autom.*, 2017.

[6] G. Loianno, C. Brunner, G. McGrath, and V. Kumar, "Estimation, control, and planning for aggressive flight with a small quadrotor with a single camera and IMU," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, 2017.

[7] K. Su and S. Shen, "Catching a flying ball with a vision-based quadrotor," in *Int. Symp. Experimental Robotics*, 2016.

[8] M. Kamel, M. Burri, and R. Siegwart, "Linear vs nonlinear MPC for trajectory tracking applied to rotary wing micro aerial vehicles," *arXiv*, 2016. [Online]. Available: http://arxiv.org/abs/1611.09240

[9] M. Neunert, C. de Crousaz, F. Furrer, M. Kamel, F. Farshidian, R. Siegwart, and J. Buchli, "Fast nonlinear model predictive control for unified trajectory optimization and tracking," in *IEEE Int. Conf. Robot. Autom.*, 2016.

[10] M. Bangura and R. Mahony, "Real-time model predictive control for quadrotors," *IFAC World Congress*, 2014.

[11] B. Penin, R. Spica, P. Robuffo Giordano, and F. Chaumette, "Vision-Based Minimum-Time Trajectory Generation for a Quadrotor UAV," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2017.

[12] R. Spica, P. Robuffo Giordano, and F. Chaumette, "Coupling Active Depth Estimation and Visual Servoing via a Large Projection Operator," *Int. J. Robot. Research*, vol. 36, no. 11, 2017.

[13] G. Costante, J. Delmerico, M. Werlberger, P. Valigi, and D. Scaramuzza, "Exploiting photometric information for planning under uncertainty," *Robotics Research: Volume 1*, pp. 107–124, 2018.

[14] C. Forster, M. Pizzoli, and D. Scaramuzza, "Appearance-based active, monocular, dense depth estimation for micro aerial vehicles," in *Robotics: Science and Systems*, 2014.

[15] M. Sheckells, G. Garimella, and M. Kobilarov, "Optimal visual servoing for differentially flat underactuated systems," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2016.

[16] T. Nägeli, J. Alonso-Mora, A. Domahidi, D. Rus, and O. Hilliges, "Real-time motion planning for aerial videography with dynamic obstacle avoidance and viewpoint optimization," *IEEE Robot. Autom. Lett.*, vol. 2, no. 3, 2017.

[17] T. Nägeli, L. Meier, A. Domahidi, J. Alonso-Mora, and O. Hilliges, "Real-time planning for automated multi-view drone cinematography," in *SIGGRAPH*, 2017.

[18] C. Potena, D. Nardi, and A. Pretto, "Effective target aware visual navigation for uavs," in *Eur. Conf. Mobile Robots*, 2017.

[19] R. Szeliski, *Computer Vision: Algorithms and Applications*, 2010.

[20] B. Houska, H. Ferreau, and M. Diehl, "An Auto-Generated Real-Time Iteration Algorithm for Nonlinear MPC in the Microsecond Range," *Automatica*, vol. 47, no. 10, 2011.

[21] B. Houska, H. Ferreau, and M. Diehl, "ACADO Toolkit – An Open Source Framework for Automatic Control and Dynamic Optimization," *Optimal Control Applications and Methods*, vol. 32, no. 3, 2011.