Model Predictive Spherical Image-Based Visual Servoing On SO(3) for Aggressive Aerial Tracking

Chao Qin¹, Qiuyu Yu², and Hugh H.T. Liu¹

Abstract—This paper presents an image-based visual servo control (IBVS) method for a first-person-view (FPV) quadrotor to conduct aggressive aerial tracking. There are three major challenges to maneuvering an underactuated vehicle using IBVS: (i) finding a visual feature representation that is robust to large rotations and is suited to be an optimization variable; (ii) keeping the target visible without sacrificing the robot's agility; and (iii) compensating for the rotational effects in the detected features. We propose a complete design framework to address these problems. First, we employ a rotation on SO(3) to represent a spherical image feature on S^2 to gain singularity-free and second-order differentiable properties. To ensure target visibility, we formulate the IBVS as a nonlinear model predictive control (NMPC) problem with three constraints taken into account: the robot's physical limits, target visibility, and time-to-collision (TTC). Furthermore, we propose a novel attitude-compensation scheme to enable formulating the visibility constraint in the actual image plane instead of a virtual fix-orientation image plane. It guarantees that the visibility constraint is valid under large rotations. Extensive experimental results show that our method can track a fast-moving target stably and aggressively without the aid of a localization system.

Index Terms— Visual servoing, aerial systems: mechanics and control, model predictive control.

I. INTRODUCTION

While human pilots employ visual information from a front-looking camera of a drone to accomplish complicated missions such as racing and chasing [1], most vision-based unmanned aerial vehicles (UAVs) require an intermediate localization module to transform image data to position feedback [2], [3]. Such transformation not only increases time delay but also makes the system vulnerable to image blurs, poor lighting conditions, and modeling errors [4]. Moreover, it will take up large amounts of computational resources that may not be affordable for small UAVs. Therefore, it is of great importance to investigate image-based visual servo control (IBVS) that can directly exploit visual information for robot control [5].

However, IBVS exhibits a degraded performance on a quadrotor since the 6-DOF commands from a classic IBVS method [6] cannot be perfectly executed by an underactuated platform with only 4 DOF [7]. Furthermore, since the camera is rigidly connected to the vehicle, the coupling of the quadrotor's translational and rotational motions can significantly hinder the target visibility. To mitigate these effects,

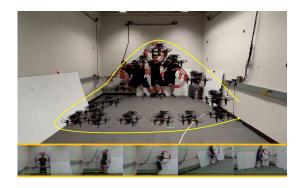


Fig. 1. Our IBVS method can track a fast-moving target and keep it visible during large rotations. We can ensure that the produced aggressive maneuvers will not jeopardize the target visibility. In this experiment, the circular racing gate is detected by a deep-learning-based object detector (blue box), and its center is kept inside the specified image bound (green box) at all times.

previous methods operate the quadrotor in a near-hover state [7]–[9], which seriously restricts its agility, or implicitly assume that controlling the heading direction suffices to keep the target visible [10], which is impractical under large pitching motions. This paper does not rely on any assumption regarding target visibility, because it can be satisfied by fully considering the impact of the quadrotor dynamics on the motion of the image feature. As shown in Fig. 1, the target can be kept inside the camera field of view (FOV) under aggressive maneuvers.

Recently, formulating IBVS on a virtual fixed-orientation image plane has been validated in autonomous landing tasks [8], [9], [11]. However, these methods will fail in agile flights since their visibility constraints will become useless when the camera rotation is large. An example is illustrated in Fig. 2. Additionally, they demand an extra position [4], velocity [12], or attitude [8], [9] controller which has no guarantee to fulfill the perceptive constraints. This paper solves the invalid visibility constraint problem and removes the need for a position/velocity/attitude controller.

In this paper, we present a spherical IBVS method based on nonlinear model predictive control (NMPC) to enable aggressive aerial tracking with a quadrotor equipped with a front-looking camera and an inertial measurement unit (IMU). To tackle large rotations, we project the 2D image feature to the unit sphere and utilize a rotation as the underlying representation of the resulting 3D unit vector. This feature representation not only enjoys the internal passivity-like property that mitigates the impact of underactuation [13] but also addresses the non-smooth vector field problem as

¹Chao and Hugh H.T. Liu are with University Toronto Institute for Aerospace Studies. Toronto. Canada chao.qin@mail.utoronto.ca, hugh.liu@utoronto.ca

²Qiuyu Yu is with the School of Astronautics and Aeronautics, Shanghai Jiao Tong University, Shanghai, China joeyyu@sjtu.edu.cn

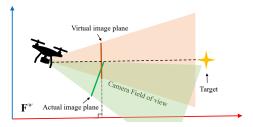


Fig. 2. The underlying problem of formulating visibility constraints in a virtual image plane instead of the actual image plane. In this example, even though the target has already left the camera FOV, the visibility constraint defined in the virtual image plane can still be satisfied, meaning that it is already an invalid constraint.

stated by the "hairy ball theorem" [14] when optimizing a state variable (3D unit vector) on the S^2 manifold. Then, an optimal control problem is constructed which takes the image kinematics and quadrotor dynamics as equality constraints and takes the actuation limits and target visibility as inequality constraints. Additionally, we propose a time-to-collision (TTC) constraint to reduce overshoots caused by high tracking speeds.

To sum up, the novel contributions of this paper are:

- We present a robust IBVS algorithm for agile underactuated robots. The target visibility can be guaranteed without any assumption on the robot motion. Our work will be available online ¹
- We introduce a rotation-based feature representation that is suited for model predictive IBVS, a.k.a. visual predictive control (VPC).
- We propose a novel attitude-compensation scheme to enable formulating the visibility constraint in the actual image plane.

II. RELATED WORKS

Aerial tracking algorithms can be divided into positionbased methods and image-based methods. In this section, we restrict most of our attention to image-based methods.

A. Position-based Methods

Position-based methods assume that the locations of the robot and target are accessible via GPS [16] or a visionbased state estimator [2]. Given these measures, they solve the tracking problem in two steps: trajectory planning and position tracking. To improve flight safety, many cost functions and constraints are raised which include target visibility [3], collision avoidance [17], and distance keeping [2]. Positionbased visual servo control (PBVS) is a special class of position-based methods which utilize relative pose estimate between the robot and the target for target tracking [18]. Recently, trajectory planning has also been widely applied in PBVS to boost the tracking performance [4], [19]. However, motion planning and control in the Cartesian space requires accurate model parameters of the camera and robot [10]. Furthermore, it is challenging to obtain high-accuracy position estimate from low-quality images.

B. Image-based Methods

Fortunately, we are still able to identify some objects or special patterns from low-quality images. These visual features can be used in image-based methods to continue the flight mission, and strong robustness to camera calibration errors and distance errors can be achieved [5]. However, features captured by an onboard camera contain both spatial information about the objects and the robot's attitude, and one needs to decouple these two effects to achieve global convergence [10]. Image-based methods can be classified into two categories based on different camera-rotation decoupling schemes: invariant feature approaches and virtual camera approaches.

Invariant feature approaches are based on the observation that rotational motions will not change the shape of an object projected on the unit sphere [20]. Consequently, if the shape information is modeled as a feature, its kinematics will be invariant to the camera's rotation. Tahri et al. [20] provided a systematic analysis of a control law using invariant features. Fomena et al. [21] utilized the Cartesian distance between the spherical projections of three points to design a controller that is robust to point-range errors. Guo et al. [10] showed that the invariant features together with the heading angle can serve as the flat outputs of the differentially-flat vision-based quadcopter system. Based on this property, a geometric tracking controller can be developed to accomplish a multipleopenings traversing task. However, the visibility constraint is not considered in these methods since it is difficult to map invariant features to points in the image plane.

Virtual camera approaches compensate for the camera's rotation using the roll and pitch estimates from an IMU. Therefore, features defined in the virtual image plane can be rotation-invariant, so as their kinematics [7], [9], [11]. Li et al. [7] present an adaptive backstepping controller to regulate the quadrotor's translation and heading relative to a planar target with arbitrary orientation. Zhang et al. [9] took image moments defined in a level virtual image plane as state variables for NMPC. Similarly, Mcfadyen et al. [8] constructed an optimal control problem on a virtual spherical camera model with a single-point feature. The major advantage of virtual camera approaches is the convenience of formulating the visibility constraint. However, previous methods define the visibility constraint in the virtual image plane which fails to prevent the feature to escape the camera FOV in large rotations as shown in Fig. 2. To make visibility constraint effective in aggressive maneuvers, we propose to keep the detected feature in the actual image plane and manage attitude compensation in the objective function.

C. Feature Representation and Parametrization

Classic IBVS methods represent features in the homogenous coordinate [6]. Although this representation is simple and intuitive, it cannot generate the optimal Cartesian path for large rotation around the optical axis [22], which results in the well-known camera retreat problem [23]. An effective solution is to transform the feature in different coordinate systems. Chaumette et al. [24] expressed features in the polar

¹https://github.com/ChaoqinRobotics

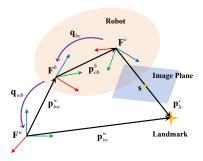


Fig. 3. A schematics representing the world frame \mathscr{F}^w , body frame \mathscr{F}^b , and camera frame \mathscr{F}^c . The standard basis is colored as $\{\mathbf{x}, \mathbf{y}, \mathbf{z}\}$. The relative pose from the camera frame to the body frame is expressed as $(\mathbf{p}^b_{bb}, \mathbf{q}_{bc})$. Similarly, $(\mathbf{p}^w_{bw}, \mathbf{q}_{wb})$ denotes the pose from the body frame to the world frame. A landmark point l located at \mathbf{p}^w_{lw} is projected onto image coordinate at \mathbf{s} .

coordinate to obtain stable performance in large rotations. Projecting features to the spherical coordinate can also handle the camera retreat problem [23]. Hamel et al. [13] proved that this projection is amendable for underactuated robots since it preserves the passivity-like property of the image kinematics. As a result, spherical IBVS has become one of the most popular controllers for quadrotors [8], [25].

In spherical IBVS, there are multiple ways to parameterize a point on the spherical surface, but not all of them are suited to be an optimization variable. For example, Corke [5] and Mcfadyen et al. [8] used 2D-angles which corresponds to the minimal representation. However, this parametrization induces a singularity in the image kinematics around certain angles. Hamel et al. [13] and Mebarki et al. [25] applied a 3D unit vector to over-parameterize features on the unit sphere. The underlying problem of this parametrization is the non-smooth vector fields on S^2 , a consequence of the "hairy ball theorem" [14]. Since this property is not desired for optimization solvers, Bloesch et al. [26] proposed to use a rotation to parameterize a 3D unit vector to get a smooth vector field, and they validated this methodology in vision-based pose estimation. In this paper, we introduce this feature parametrization [26] to NMPC-based spherical IBVS to obtain stable optimization performance.

III. SPHERICAL IMAGE KINEMATICS ON SO(3)

In this section, we introduce spherical imaging and the rotation-based feature parametrization.

A. Notation

As shown in Fig. 3, three coordinate frames are used throughout the paper: the world frame, \mathscr{F}^w , the body frame \mathscr{F}^b , and the camera frame, \mathscr{F}^c . Let $\mathbf{p}_{lw}^w \in \mathbb{R}^3$ be the global position of a landmark l, which reads "the vector from the origin of frame \mathscr{F}^w to point l, expressed in frame \mathscr{F}^{w} ". A rotation is expressed as a quaternion $\mathbf{q} \in SO(3)$ with corresponding rotation matrix denoted as $\mathbf{C}(\mathbf{q}) \in \mathbb{R}^{3\times 3}$. We use both \mathbf{q}_{wc} and $\mathbf{C}(\mathbf{q}_{wc})$ to represent the rotation from frame \mathscr{F}^c to frame \mathscr{F}^w . An operator \otimes is used to denote the Hamilton product of two quaternions. A point in the image plane expressed in the homogeneous coordinate is denoted

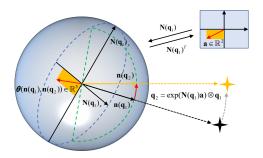


Fig. 4. Representation of a point on the spherical coordinate following [26]. A 3D unit vector ρ_1 on the unit sphere pointing toward to the landmark is transformed into an equivalent form $\mathbf{n}(\mathbf{q}_1)$ where \mathbf{q}_1 is its underlying rotation representation. The matrix $\mathbf{N}(\mathbf{q}_1) = [\mathbf{N}(\mathbf{q}_1)_x, \mathbf{N}(\mathbf{q}_1)_y]$ determines the orthonormal vector that spans the tangent space such that operators \boxplus and \boxminus can be uniquely defined on the manifold. The operator \boxminus inputs two 3D unit vectors and outputs their difference $\mathbf{a} \in \mathbb{R}^2$. Conversely, the operator \boxminus takes an element from \mathbb{R}^2 and rotates \mathbf{q}_1 to \mathbf{q}_2 . The new location of the point in the unit sphere can be obtained by $\rho_2 = \mathbf{n}(\mathbf{q}_2)$.

by $\mathbf{s} = (u, v, 1)$. The skew-symmetric matrix of a 3D vector is written by $(\cdot)^{\times}$.

B. Spherical Imaging

Spherical imaging applies a unit sphere as the image "plane" [5]. Assume that the intrinsic parameter matrix [27], $\mathbf{K} \in \mathbb{R}^{3 \times 3}$, of the pinhole camera is known. A detected feature \mathbf{s} can be projected on the unit sphere by using $\rho = \mathbf{v}/||\mathbf{v}||$ where $\mathbf{v} = \mathbf{K}^{-1}\mathbf{s} \in \mathbb{R}^3$.

C. Rotation-Based Feature Representation

A rotation $\mathbf{q} \in SO(3)$ is utilized as the underlying representation of a 3D unit vector $\rho \in S^2$. We first define the following quantities:

$$\mathbf{n}(\mathbf{q}) = \mathbf{C}(\mathbf{q})\mathbf{e}_z,\tag{1}$$

$$\mathbf{N}(\mathbf{q}) = [\mathbf{C}(\mathbf{q})\mathbf{e}_x, \mathbf{C}(\mathbf{q})\mathbf{e}_y], \tag{2}$$

where $\mathbf{I} = [\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z] \in \mathbb{R}^{3 \times 3}$. We can compute the 3D unit vector $\boldsymbol{\rho}$ from \mathbf{q} by rotating \mathbf{e}_z by \mathbf{q} , i.e., $\boldsymbol{\rho} = \mathbf{n}(\mathbf{q})$. The matrix $\mathbf{N}(\mathbf{q})$ contains the rotated \mathbf{e}_x and \mathbf{e}_y vectors that spans the tangent space around $\boldsymbol{\rho}$ as shown in Fig. 4. Given this tangent space, we are able to define the following operators:

$$\boxplus :SO(3) \times \mathbb{R}^2 \to SO(3),$$
(3)

$$\mathbf{q}, \mathbf{a} \mapsto \exp(\mathbf{N}(\mathbf{q})\mathbf{a}) \otimes \mathbf{q},$$
 (4)

$$\exists : SO(3) \times SO(3) \to \mathbb{R}^2,$$
 (5)

$$\mathbf{q}_1, \mathbf{q}_2 \mapsto \mathbf{N}(\mathbf{q}_2)^T \theta(\mathbf{n}(\mathbf{q}_1), \mathbf{n}(\mathbf{q}_2)),$$
 (6)

where $\exp(\cdot)$ is the exponential map of SO(3); \mathbf{q}_1 , $\mathbf{q}_2 \in SO(3)$, $\mathbf{a} \in \mathbf{R}^2$, and the function $\theta(\cdot, \cdot)$ returns the minimal rotation vector between two unit vectors:

$$\theta(\rho_1, \rho_2) = \arccos(\rho_1^T \rho_2) \frac{\rho_1 \times \rho_2}{||\rho_1 \times \rho_2||} \in \mathbb{R}^3.$$
 (7)

D. Spherical Image Kinematics

Consider a stationary 3D landmark point l. Let \mathbf{q} be the corresponding detected feature in the camera frame and r be the range from the camera origin to point l. Our goal is to

obtain the time derivatives of \mathbf{q} denoted as $\dot{\mathbf{q}}$ and the time derivatives of r denoted as \dot{r} .

From Fig. 3, we know $\mathbf{p}_{lw}^{w} = \mathbf{C}(\mathbf{q}_{wc})\mathbf{p}_{lc}^{c} + \mathbf{p}_{cw}^{w}$. Differentiating this equation with respect to time on both sides, inserting $\mathbf{p}_{lc}^{c} = \mathbf{n}(\mathbf{q})r$, and applying the chain rule yield:

$$\mathbf{0} = \frac{d}{dt}(\mathbf{p}_{lw}^{w}) = \frac{d}{dt}(\mathbf{C}(\mathbf{q}_{wc})(\mathbf{n}(\mathbf{q})r) + \mathbf{p}_{cw}^{w})
= \dot{\mathbf{p}}_{cw}^{w} + \dot{\mathbf{C}}(\mathbf{q}_{wc})\mathbf{n}(\mathbf{q})r + \mathbf{C}(\mathbf{q}_{wc})\frac{\partial \mathbf{n}(\mathbf{q})}{\partial \mathbf{q}}\dot{\mathbf{q}}r + \mathbf{C}(\mathbf{q}_{wc})\mathbf{n}(\mathbf{q})\dot{r}.$$
(8)

Some of the derivatives in (8) can be obtained via three-dimensional geometry [27]:

$$\dot{\mathbf{p}}_{cw}^{w} = \mathbf{v}_{cw}^{w},\tag{9}$$

$$\dot{\mathbf{C}}(\mathbf{q}_{wc}) = \boldsymbol{\omega}_{cw}^{w^{\times}} \mathbf{C}(\mathbf{q}_{wc}), \tag{10}$$

where \mathbf{v}_{cw}^{w} denotes the velocity of the robot expressed in frame \mathscr{F}^{w} and ω_{wc}^{w} denotes the angular velocity from frame \mathscr{F}^{w} to frame \mathscr{F}^{c} , expressed in frame \mathscr{F}^{w} . The most tricky one, $\frac{\partial}{\partial \mathbf{q}} \mathbf{n}(\mathbf{q})$, can be derived as follows:

$$\frac{\partial}{\partial \mathbf{q}} \mathbf{n}(\mathbf{q}) = \lim_{\varepsilon \to 0} \begin{bmatrix} \left(\frac{\mathbf{n}(\mathbf{q} \boxplus (e_1 \varepsilon)) - \mathbf{n}(\mathbf{q})}{\varepsilon}\right)^T \\ \left(\frac{\mathbf{n}(\mathbf{q} \boxplus (e_2 \varepsilon)) - \mathbf{n}(\mathbf{q})}{\varepsilon}\right)^T \end{bmatrix}^T \\
\approx \lim_{\varepsilon \to 0} \begin{bmatrix} \left(\frac{(\mathbf{I} + (\mathbf{N}(\mathbf{q}) e_1 \varepsilon)^{\times}) \mathbf{C}(\mathbf{q}) e_z - \mathbf{C}(\mathbf{q}) e_z}{\varepsilon}\right)^T \\ \left(\frac{(\mathbf{I} + (\mathbf{N}(\mathbf{q}) e_2 \varepsilon)^{\times}) \mathbf{C}(\mathbf{q}) e_z - \mathbf{C}(\mathbf{q}) e_z}{\varepsilon}\right)^T \end{bmatrix}^T \\
= -\mathbf{n}(\mathbf{q})^{\times} \mathbf{N}(\mathbf{q}) \in \mathbb{R}^{3 \times 2}.$$
(11)

where $\varepsilon \in \mathbf{R}$ is the small-angle perturbation and $\mathbf{e}_{1/2} \in \mathbb{R}^2$ are orthonormal basis vectors. Here the identity $\mathbf{v}_1^{\times} \mathbf{v}_2 = -\mathbf{v}_2^{\times} \mathbf{v}_1$ is used where $\mathbf{v}_{1/2} \in \mathbb{R}^3$. The small-angle approximation is also used: $\mathbf{C}(\exp(\Delta \psi) \otimes \mathbf{q}) \approx (\mathbf{I} + \Delta \psi^{\times}) \mathbf{C}(\mathbf{q})$ where $\Delta \psi \in \mathbb{R}^3$.

We define $\mathbf{v}^c = \mathbf{C}(\mathbf{q}_{wc})^T \mathbf{v}_{cw}^w$ and $\boldsymbol{\omega}^c = \mathbf{C}(\mathbf{q}_{wc})^T \boldsymbol{\omega}_{cw}^w$. Inserting (9-11) and $(\mathbf{v}^c, \boldsymbol{\omega}^c)$ into (8), followed by left multiplying $\mathbf{C}(\mathbf{q}_{wc})^T$ on both sides, we get:

$$\mathbf{0} = \mathbf{v}^c - \mathbf{n}(\mathbf{q})^{\times} \boldsymbol{\omega}^c r - \mathbf{n}(\mathbf{q})^{\times} \mathbf{N}(\mathbf{q}) \dot{\mathbf{q}} r + \mathbf{n}(\mathbf{q}) \dot{r}. \tag{12}$$

Here the identity $(\mathbf{C}(\mathbf{q}_{wc})\mathbf{n}(\mathbf{q}))^{\times} = \mathbf{C}(\mathbf{q}_{wc})\mathbf{n}(\mathbf{q})^{\times}\mathbf{C}(\mathbf{q}_{wc})^{T}$ is used to simplify the right-hand-side of (8).

Finally, we obtain the following the image kinematics as well as the range kinematics:

$$\dot{\mathbf{q}} = \mathbf{N}(\mathbf{q})^T (-\mathbf{n}(\mathbf{q})^{\times} \frac{\mathbf{v}^c}{r} - \boldsymbol{\omega}^c), \tag{13}$$

$$\dot{r} = -\mathbf{n}(\mathbf{q})^T \mathbf{v}^c. \tag{14}$$

by pre-multiplying (12) with $\frac{\mathbf{N}(\mathbf{q})^T\mathbf{n}(\mathbf{q})^\times}{r}$ and $\mathbf{n}(\mathbf{q})^T$, respectively. Here the identity $\mathbf{N}(\mathbf{q})^T\mathbf{n}(\mathbf{q})^\times\mathbf{n}(\mathbf{q})^\times=-\mathbf{N}(\mathbf{q})^T$ is used for element eliminations. Note that there is a slight difference in the image kinematics compared to [26]. This is because we adopt a different robotics convention which is more commonly used in quadrotor control.

The discrete-time feature prediction can be performed by using $\mathbf{q}_{k+1} = \mathbf{q}_k \boxplus \left(\mathbf{N}(\mathbf{q})^T (-\mathbf{n}(\mathbf{q})^{\times} \frac{\mathbf{v}^c}{r} - \boldsymbol{\omega}^c) \Delta t \right)$, which can also be extended as

$$\mathbf{q}_{k+1} = \exp((-\mathbf{n}(\mathbf{q})^{\times} \frac{\mathbf{v}^{c}}{r} - (\mathbf{I} - \mathbf{n}(\mathbf{q})\mathbf{n}(\mathbf{q})^{T})\boldsymbol{\omega}^{c})\Delta t) \otimes \mathbf{q}_{k}, (15)$$

where k is the time step and Δt is the time gap. Here the identity $\mathbf{N}(\mathbf{q})\mathbf{N}(\mathbf{q})^T = (\mathbf{I} - \mathbf{n}(\mathbf{q})\mathbf{n}(\mathbf{q})^T)$ is used.

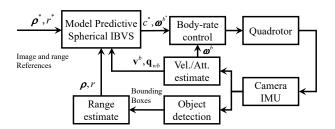


Fig. 5. Control diagram of the the proposed IBVS algorithm.

IV. MODEL PREDICTIVE VISUAL SERVOING

This section introduces the NMPC framework for our spherical IBVS. The overall system is illustrated in Fig. 5.

A. Optimal Control Problem

The non-linear optimization problem can be formulated as:

$$\min_{\mathbf{x}, \mathbf{u}, \mathbf{z}} \int_{t_0}^{t_f} (\mathcal{L}_a(\mathbf{x}, \mathbf{u}) + \mathcal{L}_p(\mathbf{x}, \mathbf{u}) + \mathcal{L}_z(\mathbf{z})) dt$$
s.t. $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}),$

$$\mathbf{h}(\mathbf{x}, \mathbf{u}, \mathbf{z}) \leq \mathbf{0},$$
(16)

where t_0 is the start time and t_f is the end time. \mathbf{x} , \mathbf{u} , and \mathbf{z} represent state variables, control inputs, and slack variables, respectively. The state dynamics $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$ consists of the image kinematics (13), range kinematics (14), and quadrotor dynamics introduced below. The inequality constraint $\mathbf{h}(\mathbf{x}, \mathbf{u}, \mathbf{z}) \leq \mathbf{0}$ includes the robot's physical constraint, targetvisibility constraint, and TTC constraint. Three objective functions are designed to fulfill the flight mission: the action objectives $\mathcal{L}_a(\mathbf{x}, \mathbf{u})$, the perception objectives $\mathcal{L}_p(\mathbf{x}, \mathbf{u})$, and the slack penalties $\mathcal{L}_z(\mathbf{z})$.

B. State Dynamics

The state and control input vectors are defined as:

$$\mathbf{x} = [\mathbf{v}^{w}, \mathbf{q}_{wb}, \mathbf{q}, r]^{T}, \tag{17}$$

$$\mathbf{u} = [c, \boldsymbol{\omega}^b]^T, \tag{18}$$

where \mathbf{v}^w is the simplified notation of \mathbf{v}^w_{cw} ; $\boldsymbol{\omega}^b$ is the simplified notation of $\boldsymbol{\omega}^b_{bw}$, a.k.a. the body rates of the quadrotor; \mathbf{q}_{wb} denotes the quadrotor orientation; and c denotes the mass-normalized thrust. We assume that all the state variables in (17) can be measured or estimated from the input image.

The full state dynamics can be constructed as:

$$\dot{\mathbf{v}}^{w} = \mathbf{C}(\mathbf{q}_{wb})\mathbf{c} + \mathbf{g}^{w},
\dot{\mathbf{q}}_{wb} = \frac{1}{2} \begin{bmatrix} 0 \\ \omega^{b} \end{bmatrix} \otimes \mathbf{q}_{wb},$$
(19)

$$\dot{\mathbf{q}} = \mathbf{N}(\mathbf{q})^{T} (-\frac{1}{r} \mathbf{n}(\mathbf{q})^{\times} \mathbf{C}(\mathbf{q}_{bc})^{T} (\mathbf{C}(\mathbf{q}_{wb})^{T} \mathbf{v}^{w} + \omega^{b^{\times}} \mathbf{p}_{cb}^{b}) - \mathbf{C}(\mathbf{q}_{bc})^{T} \omega^{b}),$$

$$\dot{r} = -\mathbf{n}(\mathbf{q})^{T} \mathbf{C}(\mathbf{q}_{bc})^{T} (\mathbf{C}(\mathbf{q}_{wb})^{T} \mathbf{v}^{w} + \omega^{b^{\times}} \mathbf{p}_{cb}^{b})$$
(20)

where (19) corresponds to the quadrotor dynamics [28] and (20) corresponds to the image and range kinematics after mapping \mathbf{v}^c to frame \mathscr{F}^w and ω^c to frame \mathscr{F}^b , respectively; $\mathbf{g}^w = (0,0,-9.81)$ denotes the gravity vector expressed in frame \mathscr{F}^w ; and $\mathbf{c} = (0,0,c)^T$.

C. Objective Functions

To keep a specified position relative to the target, we need to retrieve positional information from the current target-feature measurement \mathbf{q}_0 as well as its future location \mathbf{q}_k predicted at time step k=1,...,N where N is the horizon of NMPC. This can be done by compensating for the time-varying quadrotor rotation \mathbf{q}_{wb_k} predicted at the corresponding time step. Let \mathbf{p}^* , \mathbf{v}^* , \mathbf{q}^*_{wb} , and \mathbf{u}^* be the reference relative-position, velocity, orientation, and control inputs, respectively. We define the following action objective function to fulfill quadrotor stabilization, rotation compensation, and target tracking:

$$\mathcal{L}_{a}(\mathbf{x}, \mathbf{u}) = \|\mathbf{p}^{*} - \mathbf{p}_{b_{k}l}^{w}\|_{\mathbf{Q}_{p}}^{2} + \|\mathbf{v}^{*} - \mathbf{v}_{k}^{w}\|_{\mathbf{Q}_{v}}^{2} + \|\mathbf{q}_{wb}^{*} - \mathbf{q}_{wb_{k}}\|_{\mathbf{Q}_{a}}^{2} + \|\mathbf{u}^{*} - \mathbf{u}_{k}\|_{\mathbf{R}}^{2},$$
(21)

where

$$\mathbf{p}_{b_{t}l}^{w} = \mathbf{C}(\mathbf{q}_{wb_{k}})(\mathbf{C}(\mathbf{q}_{bc})\mathbf{n}(\mathbf{q}_{k})r_{k} + \mathbf{p}_{cb}^{b}). \tag{22}$$

What we do here can be summarized as predicting the feature motion using the state kinematics, aligning the predicted features to the inertial frame \mathscr{F}^w , and then formulating position error $\|\mathbf{p}^* - \mathbf{p}_{b_k l}^w\|_{\mathbf{Q}_p}^2$ like PBVS to achieve global convergence. The reason we prefer a position error instead of an image error with a range error, e.g., $\|\rho^* - \mathbf{C}(\mathbf{q}_{wb_k} \otimes \mathbf{q}_{bc})\mathbf{n}(\mathbf{q}_k)\|_{\mathbf{Q}_p}^2 + \|r^* - r_k\|_{\mathbf{Q}_r}^2$, is that we want to account for the unit mismatch between the image error and range error; by scaling the image error with a range prediction, we obtain a 3D position metric with each element sharing the same unit. This can effectively facilitate the tuning of the optimization weights. Moreover, by replacing a fixed \mathbf{p}^* with a timevarying $\mathbf{p}^*(t)$, we can benefit from a path-planning algorithm, and a larger region of convergence can be gained.

We define the following perception objective to force the camera to face the target:

$$\mathscr{L}_p(\mathbf{x}, \mathbf{u}) = \|\mathbf{n}(\mathbf{q}_k)_x / \mathbf{n}(\mathbf{q}_k)_z - 0\|_{\mathbf{Q}_u}^2, \tag{23}$$

where $\mathbf{n}(\mathbf{q}_k)_{x/y/z}$ corresponds to the element in the x/y/z-axis. Here we first project the feature from the spherical coordinate to the homogeneous coordinate and then define an error in the horizontal image axis between the current feature and the image center. Note that we do not add a similar objective function in the vertical image axis as in [28] because it will result in a conflict between the action objective and the perception objective. Lastly, readers can refer to [29] for detailed formulation of the slack penalty $\mathcal{L}_z(\mathbf{z})$.

D. Visibility Constraint

To keep the feature inside the specified image bound, we model the visibility constraint as

$$-\mathbf{s}_{max} - z \leq \mathbf{n}(\mathbf{q}_k)_x / \mathbf{n}(\mathbf{q}_k)_z \leq \mathbf{s}_{max} + z, -\mathbf{s}_{max} - z \leq \mathbf{n}(\mathbf{q}_k)_y / \mathbf{n}(\mathbf{q}_k)_z \leq \mathbf{s}_{max} + z,$$
(24)

where z denotes the slack variable. Since this is a soft constraint and it is still possible for the feature to exceed the bound, $\mathbf{s}_{max} > 0$ should be smaller than the actual image size to maintain a safe margin to the image edge. The main reason for not using a hard constraint is to reduce sensitivity to noise. Our practices show that it is very easy for a hard visibility constraint to cause an optimization failure when the target is close and the camera rotation is large. This is because a small rotation can result in a large feature motion if r is small according to (13) and any noise may lead to an infeasible solution.

E. Time-to-Collision Constraint

Humans can perceive an informational variable called time to collision—the required time to collide with an object if the current velocity is maintained—to avoid potential collision [15]. TTC can be expressed as:

$$t_c = r/\dot{r}. (25)$$

We model TTC as a soft constraint to prevent the drone to be too close to the target. Let $\mathcal{T}_c = \{t_c \in \mathbb{R} | t_c < 0 \text{ or } t_c \geqslant t_{c_{min}}\}$ be the safe t_c set where $t_{c_{min}}$ is the minimum TTC. It can be interpreted as the flight is safe if the robot is leaving the object $(t_c < 0)$ or it takes a time longer than $t_{c_{min}}$ to hit the object. To keep t_c inside the feasible set \mathcal{T}_c , we define the TTC constraint as:

$$\dot{r}/r \leqslant 1/t_{c_{min}} + z. \tag{26}$$

Compared with the distance (range) constraint as in [2], the proposed TTC constraint can provide an additional speed regulation which is very effective in reducing tracking overshoot in a high-speed flight.

F. Implementation Details

The system is implemented in C++ and runs in ROS environment. The NMPC framework is based on ACADO with qpOASES as the solver [30]. We set the time gap as $\Delta t = 0.05$ s with a horizon of N = 20. To account for the drifts in feature prediction, our framework works in a receding-horizon fashion by iteratively solving the optimization problem. In order to deal with the intermittent object-detection outage, we store all the control inputs after each iteration and broadcast them in order if the new detection result is not received on time.

V. SIMULATION

We simulate a hummingbird quadrotor using RotorS [31]. A proportional-integral-derivative (PID) controller is developed for low-level body rate tracking. The target is simulated as a movable 3D point and a pinhole camera model with an image size of $752 \times 480 \text{ px}^2$ is used to generate image feedback.

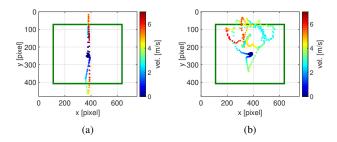


Fig. 6. Observed features in the image plane. The specified image bound is visualized as a green rectangle. The color of the feature point indicate the speed of the robot when the feature is being captured. (a) Features observed in the stationary target tracking scenario. (b) Features observed in the moving target tracking scenario.

A. Stationary Target Tracking

In this scenario, the quadrotor is commanded to reach a stationary target from 20 m away. The target's position is (20,0,1) m and the quadrotor's initial position is (0,0,1) m with velocity and attitude both being zeros. The desired range is set to be $r^* = 2$ m and the reference feature is the image center $\rho^* = (0,0,1)^T$. The image bound is specified as a rectangle centered at the middle of the image with a size of 557×336 px². The minimum TTC is set to be $t_{c_{min}} = 2$ s.

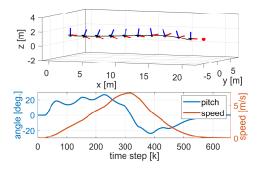


Fig. 7. Quadrotor trajectory in the stationary target tracking scenario with the target visualized as a red ball. The body frame is indicated by $\{x, y, z\}$. The time evolution of the pitch angle and the speed of the quadrotor is plot in the bottom figure.

Fig. 6(a) shows the observed feature in the image plane during the whole task and Fig. 7 provides the tracking trajectory of the quadrotor. As expected, the robot had successfully reached the reference state without any feature leaving the FOV. In addition, we do observe that some features are outside the image bound (green box). This is because the visibility constraint is a soft constraint that can be violated temporally if the controller prioritizes aggressiveness in certain stages such as an acceleration stage at the beginning and a deceleration stage at the end. We can also let the controller strictly obey the visibility constraint by setting a large weight for the slack variable. According to Fig. 7, we know that the maximum pitch angle of the quadrotor reaches 26.78° with a maximum speed of 6.88 m/s. This result suffices to show that our method can produce aggressive quadrotor flights and can be applicable for highspeed tasks.

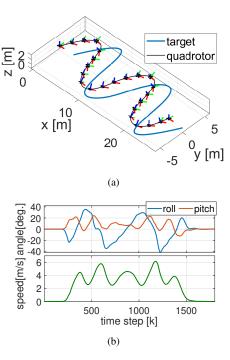


Fig. 8. (a) 3D trajectory of the quadrotor (black line) in the moving target tracking scenario with the blue line representing the target trajectory. The body frame is indicated by $\{x, y, z\}$. (b) The time evolution of the roll, pitch, and speed of the quadrotor.

B. Moving Target Tracking

In this scenario, we let the robot track a target that moves along an "S" shape trajectory. The maximum speed of the target is 6 m/s. The desired range is set to be $r^* = 3$ m and the reference image feature is the image center $\rho^* = (0,0,1)^T$. In the beginning, the range error and image error are both zeros

The trajectories of the quadrotor and the target are shown in Fig. 8(a). It shows that the robot had successfully followed the target. From Fig. 8(b), we know that the maximum roll angle is 41.62°, the maximum pitch angle is 24.26°, and the maximum tracking speed is 6.15 m/s. This result confirms that the resulting flight trajectory is aggressive enough. From Fig. 6(b), we see that under such a large rotation and a high speed, the target feature can still be visible at all times, showing that the produced aggressive maneuvers would not jeopardize the target visibility and the visibility constraint can take effect in large rotations. Another interesting finding in Fig. 6(b) is that there are a lot of features lying around the image's upper bound (green line at the top). It shows that the agility of the quadrotor is mostly restricted by the visibility constraint.

C. TTC Constraint vs. Distance Constraint

To verify the efficacy of the TTC constraint in reducing overshoot during a high-speed flight, we test the controller in the stationary target tracking scenario as in V-A with and without the TTC constraint and make a comparison with the distance (range) constraint $r \geqslant 0$. Other conditions are the same except for the type of constraint. We focus on their

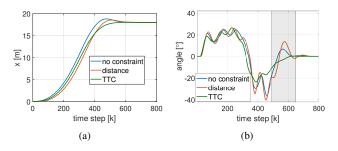


Fig. 9. Comparison of the tracking performance with different constraint strategies. (a) The time evolution of the position in x-axis. (b) The time evolution of the pitch angle with the terminal stage highlighted in grey.

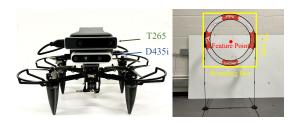


Fig. 10. Hardware for real-world experiment. The left figure is our quadrotor platform. The right figure is the target object, a racing gate. The gate can be detected by a deep-learning-based detector which outputs a bounding box as shown in the yellow rectangle. The center point of the bounding box will serve as the feature point.

respective control performances in terms of overshoot and pitch angles around the terminal stage (when the quadrotor is sufficiently close to the target).

The time evolution of the quadrotor's position in the xaxis and pitch angles are shown in Fig. 9(a) and Fig. 9(b), respectively. The maximum speeds for all tests are larger than 6.0 m/s. We see that large overshoots occur when either the distance constraint or no constraint is applied, whereas there is almost no overshoot when the TTC constraint is applied. More specifically, the overshoot for the TTC-constraint case is 0.02 m, which is much lower than 0.82 m of the noconstraint case and 0.55 m of the distance-constraint case. The results suggest that the TTC constraint is effective to decrease the tracking overshoot. Moreover, we see from Fig. 9(b) that the pitch angle is varying intensively around the terminal stage if the distance constraint is used, which will cause large camera shakes and hinder the target visibility. In comparison, the TTC constraint contributes to a smooth pitch-angle trajectory such that the feature can move in the image plane slowly and smoothly.

VI. EXPERIMENT

A. Hardware

The quadrotor platform, shown in Fig. 10, is equipped with an Intel RealSense D435i depth camera (only the left camera is used) with an image size of $1280 \times 640 \,\mathrm{px^2}$ and an IntelRealsense T265 tracking camera to provide velocity estimate. The onboard computer is a Jetson Xavier NX with its CPU running the control system and its GPU running a deep-

learning-based object detector, Yolov5². The target object is a circular racing gate with a diameter of $d_{gate} = 75$ cm. The detector can output a bounding box of the gate as shown in the yellow box in Fig. 10. Using d_{gate} and the size of the detected bounding box, (w_{box}, h_{box}) , we can estimate the range to the gate by using $r = ||\rho \cdot d_{gate}/\max(\frac{w_{box}}{f_{cam}}, \frac{h_{box}}{f_{cam}})||_2$ where ρ is the center of the bounding-box and f_{cam} is the camera's focal length. The low-level body rate controller is based on the Pixhawk4 autopilot.

B. Real-World Moving Gate Tracking

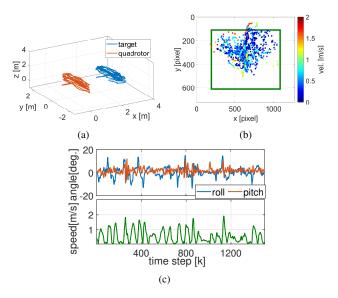


Fig. 11. Experiment results of the moving-gate tracking task. (a) Trajectories of the quadrotor and the target. (b) Observed features in the image plane. The specified image bound is visualized as a green rectangle. The color of the feature point indicate the speed of the robot when the feature is being captured. (c) The time evolution of the roll, pitch, and speed of the quadrotor.

This experiment aims to show that our approach can accomplish target tracking stably with pure onboard sensing and computing. The environment is a 5 m \times 4.3 m \times 3 m indoor space as shown in Fig. 1. The desired range is set to be $r^* = 2$ m and the reference feature is the image center $\rho^* = (0,0,1)^T$. Once the experiment starts, we will move the gate back and forth with a speed up to 2.0 m/s and evaluate the tracking performance.

The trajectory of the quadrotor and the observed feature in the image plane are displayed in Fig. 11(a) and Fig. 11(b), respectively. It shows that the tracking was stable and the target gate can be kept inside the camera FOV even when it was moving rapidly. Fig. 11(c) plots the roll and pitch angles as well as the speed of the quadrotor, from which we know that the maximum roll angle is 14.93°, the maximum pitch-angle is 11.99°, and the maximum speed is 1.92 m/s. Although the resulting trajectory is not as aggressive as the simulation (mostly limited by the size of the indoor area), the achieved speed is still higher than the existing IBVS

²https://github.com/ultralytics/yolov5

methods for a quadrotor [8], [9]. More importantly, we verify that our framework can empower IBVS to conduct safe and aggressive flights of a quadrotor, which, to some extent, breaks the stereotype that IBVS can only fly a drone in a low-speed and near-hover state.

VII. CONCLUSION

In this paper, we propose a model predictive spherical IBVS method to conduct aerial tracking with a quadrotor equipped with a front-looking camera and an IMU. We point out the problem of formulating the visibility constraint in the virtual image plane and propose a complete framework to enable modeling the visibility constraint in the actual image plane. Then, we introduce a rotation as an underlying representation of a feature in the spherical coordinate to account for the non-smooth vector field issues as well as the singularity issue. An optimal control problem is formulated in which the image kinematics, range kinematics, and quadrotor dynamics are taken into account. Additionally, we verify the efficacy of the TTC constraint in reducing the tracking overshoot. Extensive simulations and real-world experiments validate the robustness of the proposed controller in moving-target tracking tasks. In the future, we will incorporate collision avoidance to realize image-based aerial tracking in an obstacle-rich environment.

REFERENCES

- C. Pfeiffer and D. Scaramuzza, "Human-piloted drone racing: Visual processing and control," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3467–3474, 2021.
- [2] J. Ji, N. Pan, C. Xu, and F. Gao, "Elastic tracker: A spatio-temporal trajectory planner for flexible aerial tracking," in 2022 International Conference on Robotics and Automation (ICRA). IEEE, 2022, pp. 47–53.
- [3] Q. Wang, Y. Gao, J. Ji, C. Xu, and F. Gao, "Visibility-aware trajectory optimization with application to aerial tracking," in 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2021, pp. 5249–5256.
- [4] C. Potena, D. Nardi, and A. Pretto, "Effective target aware visual navigation for uavs," in 2017 European Conference on Mobile Robots (ECMR). IEEE, 2017, pp. 1–7.
- [5] P. I. Corke, "Spherical image-based visual servo and structure estimation," in 2010 IEEE international conference on Robotics and Automation. IEEE, 2010, pp. 5550–5555.
- [6] F. Chaumette and S. Hutchinson, "Visual servo control. i. basic approaches," *IEEE Robotics & Automation Magazine*, vol. 13, no. 4, pp. 82–90, 2006.
- [7] J. Li, H. Xie, K. H. Low, J. Yong, and B. Li, "Image-based visual servoing of rotorcrafts to planar visual targets of arbitrary orientation," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7861–7868, 2021
- [8] A. Mcfadyen, L. Mejias, P. Corke, and C. Pradalier, "Aircraft collision avoidance using spherical visual predictive control and single point features," in 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2013, pp. 50–56.
- [9] K. Zhang, Y. Shi, and H. Sheng, "Robust nonlinear model predictive control based visual servoing of quadrotor uavs," *IEEE/ASME Trans*actions on Mechatronics, vol. 26, no. 2, pp. 700–708, 2021.
- [10] D. Guo and K. K. Leang, "Image-based estimation, planning, and control for high-speed flying through multiple openings," *The Inter*national Journal of Robotics Research, vol. 39, no. 9, pp. 1122–1137, 2020.

- [11] H. Sheng, E. Shi, and K. Zhang, "Image-based visual servoing of a quadrotor with improved visibility using model predictive control," in 2019 IEEE 28th international symposium on industrial electronics (ISIE). IEEE, 2019, pp. 551–556.
- (ISIE). IEEE, 2019, pp. 551–556.
 P. Roque, E. Bin, P. Miraldo, and D. V. Dimarogonas, "Fast model predictive image-based visual servoing for quadrotors," in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2020, pp. 7566–7572.
- [13] T. Hamel and R. Mahony, "Visual servoing of an under-actuated dynamic rigid-body system: an image-based approach," *IEEE Trans*actions on Robotics and Automation, vol. 18, no. 2, pp. 187–198, 2002
- [14] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006
- [15] H. Zhang, B. Cheng, and J. Zhao, "Optimal trajectory generation for time-to-contact based aerial robotic perching," *Bioinspiration & biomimetics*, vol. 14, no. 1, p. 016008, 2018.
- [16] M. Zhang and H. H. Liu, "Vision-based tracking and estimation of ground moving target using unmanned aerial vehicle," in *Proceedings* of the 2010 American Control Conference. IEEE, 2010, pp. 6968– 6973.
- [17] Z. Han, R. Zhang, N. Pan, C. Xu, and F. Gao, "Fast-tracker: A robust aerial system for tracking agile target in cluttered environments," in 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2021, pp. 328–334.
- [18] M. G. Popova and H. H. Liu, "Position-based visual servoing for target tracking by a quadrotor uav," in AIAA guidance, navigation, and control conference, 2016, p. 2092.
- [19] M. Sheckells, G. Garimella, and M. Kobilarov, "Optimal visual servoing for differentially flat underactuated systems," in 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2016, pp. 5541–5548.
- [20] O. Tahri, H. Araujo, F. Chaumette, and Y. Mezouar, "Robust image-based visual servoing using invariant visual information," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1588–1600, 2013.
- [21] R. T. Fomena, O. Tahri, and F. Chaumette, "Distance-based and orientation-based visual servoing from three points," *IEEE Transactions on Robotics*, vol. 27, no. 2, pp. 256–267, 2011.
- [22] S. Liu and J. Dong, "Robust online model predictive control for image-based visual servoing in polar coordinates," *Transactions of the Institute of Measurement and Control*, vol. 42, no. 4, pp. 890–903, 2020.
- [23] P. I. Corke and O. Khatib, *Robotics, vision and control: fundamental algorithms in MATLAB*. Springer, 2011, vol. 73.
- [24] F. Chaumette and S. Hutchinson, "Visual servo control. ii. advanced approaches [tutorial]," *IEEE Robotics & Automation Magazine*, vol. 14, no. 1, pp. 109–118, 2007.
- [25] R. Mebarki, V. Lippiello, and B. Siciliano, "Nonlinear visual control of unmanned aerial vehicles in gps-denied environments," *IEEE Transactions on Robotics*, vol. 31, no. 4, pp. 1004–1017, 2015.
- [26] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct ekf-based approach," in 2015 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, 2015, pp. 298–304.
- [27] T. D. Barfoot, "State estimation for robotics." Cambridge University Press, 2017, pp. 173–196.
- [28] D. Falanga, P. Foehn, P. Lu, and D. Scaramuzza, "Pampc: Perception-aware model predictive control for quadrotors," in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018, pp. 1–8.
- [29] R. Verschueren, G. Frison, D. Kouzoupis, J. Frey, N. v. Duijkeren, A. Zanelli, B. Novoselnik, T. Albin, R. Quirynen, and M. Diehl, "acados—a modular open-source framework for fast embedded optimal control," *Mathematical Programming Computation*, vol. 14, no. 1, pp. 147–183, 2022.
- [30] B. Houska, H. J. Ferreau, and M. Diehl, "Acado toolkit—an opensource framework for automatic control and dynamic optimization," *Optimal Control Applications and Methods*, vol. 32, no. 3, pp. 298– 312, 2011.
- [31] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart, "Rotors—a modular gazebo may simulator framework," in *Robot operating system (ROS)*. Springer, 2016, pp. 595–625.