Flying through Moving Gates without Full State Estimation

Ralf Römer[1], Tim Emmert[1], and Angela P. Schoellig[1]

***Abstract*— Autonomous drone racing requires powerful perception, planning, and control and has become a benchmark and test field for autonomous, agile flight. Existing work usually assumes static race tracks with known maps, which enables offline planning of time-optimal trajectories, performing localization to the gates to reduce the drift in visual-inertial odometry (VIO) for state estimation or training learning-based methods for the particular race track and operating environment. In contrast, many real-world tasks like disaster response or delivery need to be performed in unknown and dynamic environments. To make drone racing more robust against unseen environments and moving gates, we propose a control algorithm that operates without a race track map or VIO, relying solely on monocular measurements of the line of sight to the gates. For this purpose, we adopt the law of proportional navigation (PN) to accurately fly through the gates despite gate motions or wind. We formulate the PN-informed vision-based control problem for drone racing as a constrained optimization problem and derive a closed-form optimal solution. Through simulations and real-world experiments, we demonstrate that our algorithm can navigate through moving gates at high speeds while being robust to different gate movements, model errors, wind, and delays.**

I. Introduction

Autonomous drone racing has become popular in the research community due to the availability of fast and agile quadrotors [1], [2]. Since drone racing requires high-performance algorithms for perception, state estimation, planning, and control [3], [4], [5], it is considered a challenging benchmark and a driver for progress in autonomous flight [2]. The field has made tremendous progress recently, with maximum speeds evolving from $2\,\mathrm{m/s}$ in 2018 [6] to more than $25\,\mathrm{m/s}$ in 2023, even matching human pilots [4].

Prior work has almost exclusively considered the race track, i.e., the poses and sizes of the gates, to be known and static. This has allowed for planning and tracking of time-optimal paths and trajectories [5], [7], [8] or training learning-based state-estimation and control algorithms for the particular track [4], [9]. Moving gates are rarely addressed, with only one notable exception [9]. Even then, the perception/planning module cannot generalize to substantial changes in the race track or operating environment. Onboard state estimation is commonly addressed via visual-inertial odometry (VIO) [10]. As VIO tends to accumulate large drift in situations involving motion blur, low-texture environments, and high dynamic range [8], VIO algorithms for high-speed drone racing use a race track map and perform localization relative to the gates to reduce drift [8], [9].

[1]Technical University of Munich, Germany; TUM School of Computation, Information and Technology, Department of Computer Engineering, Learning Systems and Robotics Lab (learnsyslab.org); Munich Institute of Robotics and Machine Intelligence (MIRMI).

Corresponding Email: {ralf.roemer; tim.emmert}@tum.de.

This work has been supported by the Robotics Institute Germany, funded by BMBF grant 16ME0997K.

R. R. gratefully acknowledges the support of the research group ConVeY funded by the German Research Foundation under grant GRK 2428.

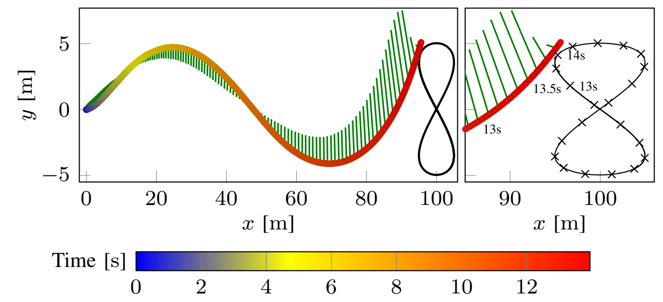

Fig. 1: Quadrotor (colored) and gate (black) trajectories and the acceleration commanded by our control algorithm (green) for flying through a fast-moving gate ($7\,\mathrm{m/s}$ top speed) solely based on line-of-sight (LOS) estimates. It can be seen that the normal acceleration, which is proportional to the LOS rate, increases as the quadrotor gets closer to the gate.

The common assumptions of static and known operating environments differ from many real-world applications like search and rescue [11], inspection of large structures, and delivery. In such scenarios, an accurate map is often unavailable, and drones can often only perceive their environment with monocular cameras. Also, operating environments are often dynamic and subject to wind, requiring vision-based algorithms that can react to strong environmental changes and disturbances. The gaps between indoor drone racing on known and static tracks and applications in unknown and dynamic environments limit the applicability of solutions developed for drone racing on such real-world tasks.

Therefore, our goal in this work is to develop a drone racing algorithm that does not require a race track map and is robust to changes in the environment, particularly moving gates. Specifically, we do not rely on VIO to provide position and velocity information. As a result, the relative position and velocity of the gate with respect to the quadrotor are unknown in our setup. The only information about the gates we assume to be available to the control algorithm is an estimate of the gate's bounding box in the camera frame of a monocular camera mounted on the quadrotor, which can be obtained using standard detection algorithms [12], [13].

To fly through a moving gate with a quadrotor, the two trajectories must closely meet at a certain point in time. We ensure this by using the concept of proportional navigation (PN) [14], which was originally developed as a missile guidance law for intercepting moving objects [15]. The PN law cannot be directly applied to quadrotors due to several differences between missile and quadrotor control. Also, the

PN law requires constant and known relative velocity, which does not match our objective of drone racing (i.e., minimizing total time) without velocity information. To address these gaps, we propose a novel PN-informed quadrotor control algorithm that is the first to pass moving gates using only monocular line-of-sight (LOS) and IMU measurements. We visualize an example of flying through a fast-moving gate in Fig. 1. In summary, our main contributions are:

- We adapt the PN law for an unknown relative velocity and introduce the PN frame to derive a state-space model for PN-informed quadrotor control.
- We formulate PN-informed drone racing without full state estimation and with moving gates as a constrained optimization problem and derive a closed-form solution.
- We present a Bayesian optimization (BO) based hyperparameter optimization strategy to achieve robust performance for different objectives with our approach.
- We demonstrate through simulation and hardware experiments that our algorithm can pass moving gates based solely on LOS and IMU measurements while being robust to different gate motions, wind, and delays.

II. Related Work

A. Autonomous Drone Racing

Many drone racing approaches track a pre-planned path or trajectory that avoids obstacles and satisfies physical constraints [5], [16], [17], [18], [19]. Common approaches for attitude control and trajectory tracking include nonlinear techniques [20], differential-flatness-based control [3] and model predictive control (MPC) [16], [21]. Perception-aware planning and control approaches [22], [23], [24] can also consider the quality of the state estimates. Planning-based methods require an accurate state estimate through motion capture or VIO [10]. Without reliable estimation of the pose and velocity relative to the gate [8], [9], which requires static gates at known locations, learning-based methods that skip the planning stage can be used [2], [4]. In [9], a vision-based imitation learning policy is trained for trajectory tracking in a static environment, and the method can generalize to some gate motion after training. Learning-based MPC with motion capture has been used to fly through a fast-moving gate [25]. Swift [4] uses a reinforcement learning (RL)-based controller [26] to generate rate commands and VIO [27] for local position estimation on the track. The RL controller is trained for the particular race track, and the VIO algorithm requires a race track map and certain environmental conditions, which makes generalizing to changing or unseen operating environments challenging. Recently, [28] demonstrated agile flight at medium speeds without explicit state estimation, using the segmented gate edges as an input for an RL policy. Despite recent advances in learning-based drone racing [4], [8], [28], these approaches have been limited to known and static race tracks in controlled environments.

B. Proportional Navigation

Proportional navigation (PN) [14], [15] is a guidance law for intercepting moving objects based on keeping the bearing angle between the velocity vectors constant. Several recent studies [29], [30], [31] address the adaptation of PN to quadrotors, but they rely on an accurate estimation of the relative distance and velocity, either via GPS [29], [31] or ultrasonic sensors [30]. Therefore, these approaches are unsuitable for drone racing with moving gates based solely on LOS measurements.

III. Problem Setup

We consider the control problem of racing through moving gates without full state estimation. The gates are detected with a monocular camera mounted on the quadrotor, and a (potentially noisy) estimate of the next gate's bounding box in the camera frame obtained via standard object detection algorithms [12], [13] is available. Since objects in the real world often have unknown dimensions, we do not assume knowledge of the gate sizes, which could otherwise be used to estimate the relative distance to the gate. Also, we do not use VIO for state estimation due to its reliance on known gate locations in high-speed drone racing scenarios [8], [9]. Consequently, the relative position and velocity between the quadrotor and the gate are assumed to be unknown. As the lack of position and velocity information and the gate motion render the drone racing problem very difficult, we make three assumptions: 1) The gates' top speeds are lower than the quadrotor's maximum speed. 2) The gate opening initially points towards the quadrotor, and the gate only exhibits translational motion. 3) After the quadrotor has passed a gate, the next gate is within the camera's field of view. We can then break down the problem of completing the entire race track into sequentially flying through the next gate in the field of view.

IV. Methodology

A. Preliminaries

1) Proportional Navigation: The PN law [14], [15] was originally derived for use in interceptor missiles, but we consider a quadrotor and a moving gate in the following. The PN law describes the planar kinematics in an inertial frame, as shown in Fig. 2. We denote the quadrotor and gate velocity at time t by $\boldsymbol{v}(t)$ and $\boldsymbol{v}_{\mathrm{g}}(t)$, but we omit the dependence on t for brevity. The vector from the quadrotor to the gate center is denoted by $\boldsymbol{r} = [x,\, y]^{\mathsf{T}}$ and their distance by $r = \|\boldsymbol{r}\|$. We define the line-of-sight (LOS) as $\boldsymbol{l} = \frac{\boldsymbol{r}}{r}$ and denote by λ the LOS angle between the x-axis $\boldsymbol{e}_x$ and $\boldsymbol{l}$. The derivation of the PN law assumes $x >> y$ and a constant relative velocity $v_{\mathrm{rel}} = \dot{r}$, i.e., $\ddot{r} = 0$. Then, we have $\sin(\lambda) \approx \lambda = \frac{y}{r}$, and the LOS angle evolves via $\dot{\lambda} = \frac{1}{r}(\dot{y} - \lambda\dot{r})$, $\ddot{\lambda} = \frac{1}{r}(\ddot{y} - 2\dot{\lambda}\dot{r} - \lambda\ddot{r})$, where $\ddot{y} = a_{\mathrm{g},y} - a_y$ is the difference between the accelerations $a_{\mathrm{g},y}$ and a_y of the gate and the quadrotor in y-direction. Defining the state as $\boldsymbol{x} = [x_1,\, x_2]^{\mathsf{T}} = [\lambda,\, \dot{\lambda}]^{\mathsf{T}}$, the control input as $u = a_y$ and the noise as $w = a_{\mathrm{g},y}$ and using $\ddot{r} = 0$ allows writing the LOS angle dynamics as

$$\begin{bmatrix} \dot{\lambda} \\ \ddot{\lambda} \end{bmatrix} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} \dot{x}_2 \\ \frac{1}{r}(w - u - 2x_2 v_{\mathrm{rel}}) \end{bmatrix}. \tag{1}$$

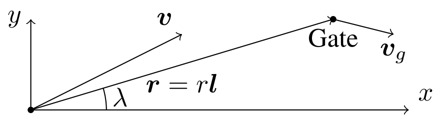

Fig. 2: Kinematics in the derivation of the PN law.

According to the parallel navigation principle [15], the quadrotor will pass the gate center for a constant LOS angle λ and positive relative velocity, i.e., if $x_2 = 0$ and $\dot{r} > 0$. Assuming zero gate acceleration in y-direction, i.e., $a_{g,y} = 0$, a Lyapunov function $V(\boldsymbol{x}) = \frac{1}{2}x_2^2$ can be defined [32]. Its time derivative satisfies $\dot{V}(t) = x_2\frac{1}{r}(-u - 2x_2 v_{\text{rel}}) < 0$ if

$$u(t) = k_{\text{PN}} v_{\text{rel}}(t) x_2(t), \quad k_{\text{PN}} > 2, \tag{2}$$

for all t, where k_{PN} is called the navigation constant. In summary, the PN law (2) commands an acceleration approximately normal to the LOS that guarantees stability of (1).

The PN-guidance approach can be generalized to 3D [15]. In this case, the LOS-rate is defined as $\dot{\lambda} = ||\boldsymbol{\omega_l}||$, where $\boldsymbol{\omega_l}$ is the angular velocity of $\boldsymbol{l}$. The commanded acceleration is perpendicular to both $\boldsymbol{\omega_l}$ and $\boldsymbol{l}$ and proportional to the norm of the relative velocity between the quadrotor and the gate. We adopt the PN law for drone racing to ensure that a quadrotor flies precisely through and not past moving gates.

2) Quadrotor Dynamics: We define the quadrotor state as $\boldsymbol{x} = \left(\boldsymbol{p}^{\text{W}}, \boldsymbol{q}_{\text{B}}^{\text{W}}, \boldsymbol{v}^{\text{W}}, \boldsymbol{\omega}^{\text{B}}\right)$, where $\boldsymbol{p}^{\text{W}}$ and $\boldsymbol{v}^{\text{W}}$ are the position and velocity in the world frame $\mathcal{O}_{\text{W}}$, $\boldsymbol{q}_{\text{B}}^{\text{W}}$ is the quaternion from the body frame $\mathcal{O}_{\text{B}}$ to $\mathcal{O}_{\text{W}}$, and $\boldsymbol{\omega}^{\text{B}}$ are the body rates. The control input $\boldsymbol{u} = (f_{\text{th}}, \boldsymbol{q}_{\text{c}})$ consists of the total thrust f_{th}, acting in the positive z-direction $\boldsymbol{e}_z$ of $\mathcal{O}_{\text{B}}$, and the desired orientation $\boldsymbol{q}_{\text{c}}$. We use the attitude controller [33] and approximate the PID rate controller [34] as a first-order system with time constant τ_ω. This yields the dynamics

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) = \begin{bmatrix} \dot{\boldsymbol{p}}^{\text{W}} \\ \dot{\boldsymbol{q}}_{\text{B}}^{\text{W}} \\ \dot{\boldsymbol{v}}^{\text{W}} \\ \dot{\boldsymbol{\omega}}^{\text{B}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{v}^{\text{W}} \\ \boldsymbol{q}_{\text{B}}^{\text{W}} \cdot \begin{bmatrix} 0 \\ \frac{1}{2}\boldsymbol{\omega}^{\text{B}} \end{bmatrix} \\ \frac{1}{m}\boldsymbol{R}_{\text{B}}^{\text{W}}\left(f_{\text{th}}\boldsymbol{e}_z - \boldsymbol{f}_{\text{aero}}^{\text{B}}\right) + \boldsymbol{g}^{\text{W}} \\ \frac{1}{\tau_\omega}\left(\boldsymbol{\omega}_c - \boldsymbol{\omega}^{\text{B}}\right) \end{bmatrix}, \tag{3}$$

where m is the mass, $\boldsymbol{R}_{\text{B}}^{\text{W}}$ is the rotation matrix from $\mathcal{O}_{\text{B}}$ to $\mathcal{O}_{\text{W}}$, $\boldsymbol{g} = [g_x,\, g_y,\, g_z]^\top$ is the gravity vector, $\boldsymbol{f}_{\text{aero}}$ are the aerodynamic forces, and $\boldsymbol{\omega}_{\text{c}} = \frac{2}{\tau}\text{sgn}(\boldsymbol{q}_{e,0})\mathbf{q}_{e,1:3}$ with the error quaternion $\boldsymbol{q}_{\text{e}} = \left(\boldsymbol{q}_{\text{B}}^{\text{W}}\right)^{-1} \cdot \boldsymbol{q}_{\text{c}}$ [33].

B. Monocular Line of Sight Estimation

Adapting the PN law (2) for drone racing requires estimating the LOS $\boldsymbol{l}$ and the LOS rate $\dot{\lambda}$. We use a monocular camera facing forward with zero camera pitch, i.e., the optical axis $\boldsymbol{o}_{\text{cam}}$ can be expressed in the body frame as $\boldsymbol{o}_{\text{cam}}^{\text{B}} = \boldsymbol{e}_x$. We detect the gate's bounding box in the camera frame $\mathcal{O}_{\text{C}}$ at a frequency f_{est}. As these estimates are noisy due to motion blur, vibration, and tracking errors, we apply a low-pass filter. We calculate the LOS in $\mathcal{O}_{\text{C}}$ from the bounding box center via the camera intrinsics and the LOS rate as

$$\dot{\lambda} = f_{\text{est}} \arccos\left(\boldsymbol{l}^\top \boldsymbol{l}_{\text{prev}}\right), \tag{4}$$

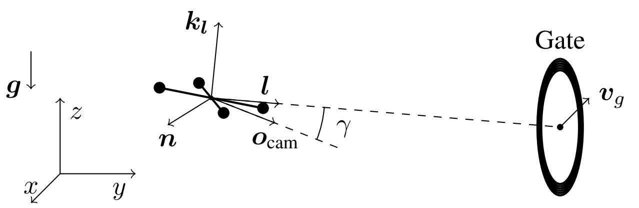

Fig. 3: Proposed control approach for flying through moving gates solely based on measuring the line-of-sight (LOS) angle γ between the optical axis of the camera $\boldsymbol{o}_{\text{cam}}$ and the LOS $\boldsymbol{l}$. To ensure that the gate is not missed, we command a PN-informed acceleration $\boldsymbol{a}_n$ in the direction $\boldsymbol{n}$ normal to $\boldsymbol{l}$ and the LOS rotation axis $\boldsymbol{k_l}$ via (5). Moreover, we maximize the acceleration towards the gate, i.e., in the direction $\boldsymbol{l}$, while enforcing an upper bound on γ to ensure that the gate always stays within the field of view.

where $\boldsymbol{l}_{\text{prev}}$ is the LOS estimate at the previous timestep. We also compute the rotation axis of the LOS as $\boldsymbol{k_l} = \boldsymbol{l} \times \boldsymbol{l}_{\text{prev}}$ to determine the direction in 3D space in which the acceleration given by the PN law must be applied, as explained below.

C. Proportional Navigation without Velocity Measurements

In the original PN law (2), the relative velocity v_{rel} is assumed to be constant and known. Since the quadrotor in our setup can and should accelerate toward the gate to minimize time, and velocity estimates are unavailable, we need to modify (2). Recall that the LOS angle dynamics (1) are stable if $k_{\text{PN}} > 2$. Since the relative velocity v_{rel} is unknown in our setup, we define the lumped PN parameter $k_v = k_{\text{PN}} v_{\text{rel}}$, which needs to satisfy the condition $k_v > 2v_{\text{rel}}$ for all v_{rel}. We denote the maximum speed of the quadrotor as $\bar{v}$ and consider the gate velocity to be upper bounded by $\bar{v}_{\text{g}}$, which implies $v_{\text{rel}} \leq \bar{v} + \bar{v}_{\text{g}} =: \bar{v}_{\text{rel}}$. Then, the condition $k_v > 2\bar{v}_{\text{rel}}$ is sufficient for stability. The acceleration given by the PN law needs to be applied approximately normal to the LOS. Moreover, the derivation of the PN law considers planar kinematics. To satisfy these conditions, we calculate the acceleration vector required for flying through the gate as

$$\boldsymbol{a}_n = k_{\text{PN}} \bar{v}_{\text{rel}} \dot{\lambda} \boldsymbol{n}, \qquad \boldsymbol{n} = \boldsymbol{l} \times \boldsymbol{k_l}, \tag{5}$$

where the acceleration is applied in the direction $\boldsymbol{n}$ normal to the LOS $\boldsymbol{l}$ and the LOS rotation axis $\boldsymbol{k_l}$, as visualized in Fig. 3. In the following, we discuss how to generate this acceleration with a quadrotor at high speeds while meeting other requirements for vision-based drone racing.

D. PN-Informed Quadrotor Control for Drone Racing

1) PN Frame: The quadrotor PN law (5) commands an acceleration normal to the LOS. For drone racing, maximizing the speed towards the gate, i.e., along the LOS, is also desirable. For a unified formulation of these objectives, we define the PN frame $\mathcal{O}_{\text{PN}}$ as

$$\boldsymbol{R}_{\text{PN}}^{\text{W}} = \begin{bmatrix} \boldsymbol{l}^{\text{W}} & \boldsymbol{e}_z \times \boldsymbol{l}^{\text{W}} & \boldsymbol{l}^{\text{W}} \times \left(\boldsymbol{e}_z \times \boldsymbol{l}^{\text{W}}\right) \end{bmatrix}. \tag{6}$$

As a result, we can express the LOS vector $\boldsymbol{l}$ and the normal vector $\boldsymbol{n}$ in (5) in $\mathcal{O}_{\text{PN}}$ as

$$\boldsymbol{l}^{\text{PN}} = \boldsymbol{R}_{\text{W}}^{\text{PN}} \boldsymbol{l}^{\text{W}} = \boldsymbol{e}_x, \qquad \boldsymbol{n}^{\text{PN}} = \begin{bmatrix} 0 & n_y & n_z \end{bmatrix}^\top. \tag{7}$$

We assume that, as the quadrotor is approaching the gate, the aerodynamic forces act approximately parallel to the LOS, i.e., $\boldsymbol{f}_{\text{aero}} \approx -f_{\text{aero}}\boldsymbol{l}$, where $f_{\text{aero}} = \|\boldsymbol{f}_{\text{aero}}\|$. Then, the quadrotor acceleration in the PN frame is given by

$$\boldsymbol{a}^{\text{PN}} = \frac{1}{m}\left(\boldsymbol{R}_{\text{B}}^{\text{PN}} f_{\text{th}} \boldsymbol{e}_z - f_{\text{aero}} \boldsymbol{e}_x\right) + \boldsymbol{g}^{\text{PN}}. \tag{8}$$

This assumption is reasonable if the quadrotor is approximately on course to fly through the gate and if the gate velocity normal to the LOS is significantly smaller than the quadrotor velocity towards the gate.

2) Constrained Optimization Problem: We aim to optimize the total thrust f_{th} and the commanded orientation $\boldsymbol{q}_{\text{c}}$ with regard to the following conditions:

1) The acceleration normal to the LOS must satisfy the quadrotor PN law (5) to fly through the moving gate.
2) The quadrotor should achieve high speed towards the gate to minimize lap time.
3) While flying towards the gate, the gate must always stay in the camera's field of view.

We also consider an upper thrust limit $\bar{f}_{\text{th}}$ and formulate the above requirements as a constrained optimization problem

$$f_{\text{th}}^*, \boldsymbol{q}_{\text{c}}^* = \underset{f_{\text{th}}, \boldsymbol{q}_{\text{c}}}{\arg\max} \quad a_{\text{los}} \tag{9a}$$

$$\text{s.t.} \quad \boldsymbol{a}^{\text{PN}} = \begin{bmatrix} a_{\text{los}} & n_y a_n & n_z a_n \end{bmatrix}^{\mathsf{T}} \tag{9b}$$

$$f_{\text{th}} \in \left[0, \bar{f}_{\text{th}}\right] \tag{9c}$$

$$\boldsymbol{q}_{\text{c}} \in \mathcal{Q}_{\text{fov}}, \tag{9d}$$

where $\boldsymbol{a}^{\text{PN}}$ is given by (8), $a_n = \|\boldsymbol{a}_n\|$ with $\boldsymbol{a}_n$ given by (5), and $\mathcal{Q}_{\text{fov}}$ contains all orientations keeping the gate within the field of view. To simplify this nonlinear optimization problem, we use the fact that accelerating towards the gate requires pitching, and applying the normal acceleration (5) requires rolling, both with respect to the PN frame $\mathcal{O}_{\text{PN}}$. Consequently, we can decompose the rotation from the body frame $\mathcal{O}_{\text{B}}$ to $\mathcal{O}_{\text{PN}}$ into two rotations as

$$\boldsymbol{R}_{\text{B}}^{\text{PN}} = \boldsymbol{R}_y(\phi)\boldsymbol{R}_x(\theta), \tag{10}$$

where ϕ and θ are the pitch and roll angle, respectively. By inserting (10) into (8), we can rewrite (9) as

$$\underset{f_{\text{th}}, \phi, \theta}{\arg\max} \quad f_{\text{th}} \sin(\phi)\cos(\theta) \tag{11a}$$

$$\begin{bmatrix} n_y a_n \\ n_z a_n \end{bmatrix} = \begin{bmatrix} -f_{\text{th}} \sin(\theta) \\ f_{\text{th}} \cos(\phi)\cos(\theta) + g_z^{\text{PN}} \end{bmatrix} \tag{11b}$$

$$\text{(9c), (9d),}$$

where we have used that $g_y^{\text{PN}} = 0$ due to the definition of $\mathcal{O}^{\text{PN}}$ in (6) and that f_{aero} and g_x^{PN} are independent of f_{th}, ϕ and θ. Solving (11b) for the roll and pitch angle yields

$$\theta(f_{\text{th}}) = \arcsin\left(-\frac{n_y a_n}{f_{\text{th}}}\right) \tag{12a}$$

$$\phi(f_{\text{th}}) = \arccos\left(\frac{n_z a_n - g_z^{\text{PN}}}{\sqrt{f_{\text{th}}^2 - (n_y a_n)^2}}\right), \tag{12b}$$

which we can use to rewrite the objective (11a) as

$$f_{\text{th}} \sin(\phi)\cos(\theta) = \sqrt{f_{\text{th}}^2 - n_y^2 a_n^2 - (n_z a_n - g_z^{\text{PN}})^2}. \tag{13}$$

To keep the gate in the field of view, we impose an upper bound $\bar{\gamma} \in (0, \gamma_{\text{cam}}]$ on the angle γ between the LOS $\boldsymbol{l}$ and the optical axis $\boldsymbol{o}_{\text{cam}}$ of the camera, where γ_{cam} is the camera's angle of view. Increasing $\bar{\gamma}$ increases the risk that due to the gate motion, model errors, or disturbances, the gate leaves the field of view at some point and can no longer be detected. It follows from $\boldsymbol{o}_{\text{cam}}^{\text{B}} = \boldsymbol{e}_x$ and (10) that the optical axis in the PN frame $\mathcal{O}_{\text{PN}}$ is given by

$$\boldsymbol{o}_{\text{cam}}^{\text{PN}} = \boldsymbol{R}_{\text{B}}^{\text{PN}} \boldsymbol{o}_{\text{cam}}^{\text{B}} = \boldsymbol{R}_y(\phi)\boldsymbol{R}_x(\theta)\boldsymbol{e}_x. \tag{14}$$

Hence, we can calculate γ in $\mathcal{O}_{\text{PN}}$ via

$$\cos(\gamma) = \boldsymbol{l}^{\mathsf{T}} \boldsymbol{o}_{\text{cam}} = \boldsymbol{e}_x^{\mathsf{T}} \boldsymbol{R}_y(\phi)\boldsymbol{R}_x(\theta)\boldsymbol{e}_x = \cos(\phi)\cos(\theta).$$

By inserting (12), we can express $\gamma < \bar{\gamma}$ as

$$\frac{n_z a_n - g_z^{\text{PN}}}{f_{\text{th}}} \geq \cos(\bar{\gamma}). \tag{15}$$

It follows from (12), (13) and (15) that the optimal control input solving the optimization problem (11) is given by

$$f_{\text{th}}^* = \min\left(\bar{f}_{\text{th}}, \frac{n_z a_n - g_z^{\text{PN}}}{\cos(\bar{\gamma})}\right), \quad \phi^* = \phi(f_{\text{th}}^*), \quad \theta^* = \theta(f_{\text{th}}^*).$$

We can see that increasing the upper bound $\bar{\gamma}$ also makes the control towards the gate more aggressive.

E. Bayesian Hyperparameter Optimization

The racing performance of our control algorithm is affected by two key hyperparameters: The navigation constant k_{PN} and the upper bound $\bar{\gamma}$ on the angle between the camera's optical axis and the LOS. To reduce the tuning effort, we use Bayesian optimization (BO) [35], a common approach for optimizing black-box functions that are expensive to evaluate. We quantify the performance of a set of hyperparameters via two metrics: The time to reach the gate t_{gate} and the distance d_{center} to the gate center at the moment of flying through it. The choice to minimize d_{center} is made for two reasons: First, passing the gate close to its center increases the robustness against colliding with the gate boundaries. Second, due to the lack of position information and the gate movements, we cannot track an optimal trajectory [5], [36], which would not necessarily go through the gate centers. To obtain optimal hyperparameters with respect to the two metrics, we define a reward function

$$R(k_{\text{PN}}, \bar{\gamma}) = \max(0,\ c_t/t_{\text{gate}} - c_d d_{\text{center}} + c), \tag{16}$$

where $c_t \geq 0$ and $c_d \geq 0$ are weighting factors to trade off the two objectives, and $c \geq 0$ is an offset. In each iteration of BO, we first fit a Gaussian process (GP) model [37] to the available parameter-reward samples. Then, we determine the next set of hyperparameters $\boldsymbol{\theta}$ to evaluate by maximizing the upper confidence bound [35] of the fitted GP by utilizing its mean and variance. After a few iterations, we obtain optimized hyperparameters for our control algorithm.

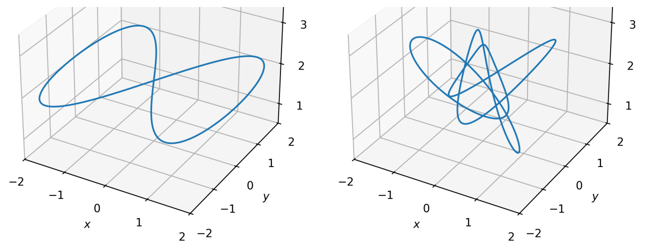

Fig. 4: Planar (2D) and knot (3D) gate motions used in our experiments.

Gate motion	Success rate		Time [s]	
	Rew. $R_{d,t}$	Rew. R_d	Rew. $R_{d,t}$	Rew. R_d
Stationary	1.00 (1.00)	1.00 (1.00)	4.11	5.28
Linear slow	1.00 (1.00)	1.00 (1.00)	4.03	5.35
Linear fast	0.67 (0.75)	0.92 (1.00)	5.41	5.80
Planar	1.00 (1.00)	0.50 (1.00)	4.01	5.76
Knot	1.00 (1.00)	1.00 (1.00)	4.03	5.41
Total	**0.93** (0.95)	0.88 (**1.00**)	**4.32**	5.52

TABLE I: Performance of our algorithm for racing through moving gates for two different formulations of the reward (16). The success rates of flying through the gates are reported for a gate radius of $1\,\mathrm{m}$ and $2\,\mathrm{m}$ (in brackets).

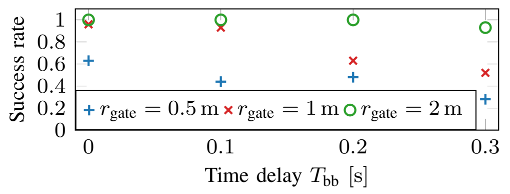

Fig. 5: Impact of delays in the vision-based LOS estimation (e.g., due to camera latencies, inference times) on the success rate of passing the gates.

V. Evaluation

A. Simulation

1) Setup: We utilize four parameterizations of the gate position over time: Stationary (0D), linear (1D), planar (2D), and knot (3D). We use two different constant velocities for the linear motion, referred to as "slow" ($2.5\,\mathrm{m/s}$) and "fast" ($5\,\mathrm{m/s}$). The planar and knot motions are defined by

$$\boldsymbol{p}_{\text{plan}}(t) = \boldsymbol{p}_0 + d_{\text{plan}} \begin{bmatrix} \mathrm{c}_{\omega t} & \frac{1}{2}\mathrm{c}_{2\omega t} & 0 \end{bmatrix}^\mathsf{T},$$
$$\boldsymbol{p}_{\text{knot}}(t) = \boldsymbol{p}_0 + d_{\text{knot}} \begin{bmatrix} \mathrm{s}_{3\omega t}\left(1 + \frac{1}{2}\mathrm{c}_{2\omega t}\right) & \mathrm{s}_{3\omega t}\mathrm{s}_{2\omega t} & \mathrm{s}_{4\omega t} \end{bmatrix}^\mathsf{T},$$

where $\mathrm{c}_\alpha = \cos(\alpha)$, $\mathrm{s}_\alpha = \sin(\alpha)$, and shown in Fig. 4. Unless otherwise stated, we set $d_{\text{plan}} = 2\,\mathrm{m}$, $d_{\text{knot}} = 1\,\mathrm{m}$, resulting in a top speed and acceleration of $3.1\,\mathrm{m/s}$ and $4.0\,\mathrm{m/s^2}$ for the planar and $1.2\,\mathrm{m/s}$ and $0.9\,\mathrm{m/s^2}$ for the knot motion.

We combine the software-in-the-loop simulation of PX4 [34] and Gazebo to simulate the quadrotor dynamics, including rotors [38], [39], low-level control, and aerodynamic forces, as well as our drone racing algorithm for the x500 quadrotor [34] with a mass of $2\,\mathrm{kg}$. The simulation provides simulated sensor measurements by applying noise to the ground truth acceleration, angular velocity, and air pressure for the IMU and barometer sensors. We use the quadrotor's true position and attitude to project the gate's ground truth position into the camera frame based on its calibration. To evaluate robustness, imperfections in the vision system can be simulated by adding noise or introducing a delay to the bounding box measurements. We use an estimation and control frequency of $30\,\mathrm{Hz}$ in all experiments. The camera angle of view is $\gamma_{\text{cam}} = 33.5°$. The gates are circular with a radius of $0.5\,\mathrm{m}$ to $2\,\mathrm{m}$. Unless otherwise stated, we consider initial distances from the quadrotor to the gate between $20\,\mathrm{m}$ and $30\,\mathrm{m}$, set the initial velocity to $10\,\mathrm{m/s}$ and randomize the angle between the initial velocity vector and the LOS between $-10\,°$ and $10\,°$. For the planar and knot gate motion, we randomize the initial LOS angle by initializing the gate position randomly on the gate trajectory.

2) Bayesian Hyperparameter Optimization: We optimize k_{PN} and $\bar{\gamma}$ via BO [40], as described in Section IV-E. For the GP, we use a Matérn kernel [37] with smoothness parameter $\nu = 2.5$. The estimate of the maximum relative speed used in (5) is $\bar{v}_{\text{rel}} = 15\,\mathrm{m/s}$. We set the search spaces for the hyperparameters to $\bar{\gamma} \in [5°, 30°]$ and $k_{\text{PN}} \in [0.3, 3.0]$. We consider two reward formulations. The "distance + time" reward $R_{d,t}$ uses $c_t = 10$, $c_d = 3$ and $c = 0$ in (16), aiming to strike a balance between robustness against collisions with the gate and minimizing time. The "distance" reward R_d reward uses $c_t = 0$, $c_d = 5$, and $c = 3$, corresponding to a conservative "racing" approach where the primary goal is to successfully fly through the gates. We perform 25 iterations for BO, each consisting of 2 runs per gate motion (excluding the stationary gate as it is the easiest to fly through). The optimized hyperparameter values are $k_{\text{PN}} = 2.10$ and $\bar{\gamma} = 21.05°$ for the reward $R_{d,t}$ and $k_{\text{PN}} = 1.11$ and $\bar{\gamma} = 8.91°$ for the reward R_d. This shows that the combined reward $R_{d,t}$ results in a more aggressive control behavior and allows having the gate closer to the boundary of the field of view.

3) Results and Discussion: We simulate 60 runs with different initial conditions for both reward formulations, evenly spread across all gate motions. The success rate of flying through the gates and the time to reach the gates t_{gate} are provided in Table I. Our approach achieves a high success rate of at least 88%, and the same hyperparameters perform well for different gate motions. The linear fast and planar gate motions are the most challenging, likely because their high speeds violate some of the assumptions in the derivation of the PN law in Section IV-A.1. Although $k_{\text{PN}} \leq 2$ for the reward R_d, high performance is achieved due to overestimating the relative velocity. The combined reward $R_{d,t}$ leads to shorter times to pass the gates and higher top speeds ($14.6\,\mathrm{m/s}$ vs. $11.0\,\mathrm{m/s}$), and we use it in the following.

We also evaluate the robustness of our algorithm against time delays in the perception, using the stationary, linear fast, and planar gate motions. For this, we consider delays of up to $300\,\mathrm{ms}$ in the LOS estimation. As shown in Fig. 5, even a large delay of $100\,\mathrm{ms}$ has little effect on performance for gate radii of $1\,\mathrm{m}$ and $2\,\mathrm{m}$.

One of the key properties of our control algorithm is that it does not require knowledge of the relative speed between the quadrotor and the gate and instead uses a fixed overestimation $\bar{v}_{\text{rel}}$. We compare this approach to the ideal case when the relative speed is known. To obtain very different relative speeds, we consider initial distances to the gate of $30\,\mathrm{m}$ and $150\,\mathrm{m}$, leading to top speeds of

Top speed [m/s]	Knowledge about v_{rel}	d_{center} [m]
15	Estimate: $\bar{v}_{\text{rel}} = 30\,\text{m/s}$	4.72 ± 1.03
	Estimate: $\bar{v}_{\text{rel}} = 20\,\text{m/s}$	1.01 ± 0.22
	Estimate: $\bar{v}_{\text{rel}} = 10\,\text{m/s}$	1.02 ± 0.34
	Perfect knowledge	$\mathbf{0.42 \pm 0.19}$
25	Estimate: $\bar{v}_{\text{rel}} = 30\,\text{m/s}$	$\mathbf{0.65 \pm 0.05}$
	Estimate: $\bar{v}_{\text{rel}} = 20\,\text{m/s}$	0.70 ± 0.35
	Estimate: $\bar{v}_{\text{rel}} = 10\,\text{m/s}$	4.84 ± 0.45
	Perfect knowledge	1.19 ± 0.43

TABLE II: Comparing our approach to overestimate the relative velocity v_{rel} to be $\bar{v}_{\text{rel}}$ with the idealized case that v_{rel} is known at all times.

Gate motion	PN-based accel. contr. [31]	PN-informed contr. (ours)
Linear	0.35	**0.69**
Planar	0.41	**0.56**

TABLE III: Comparison against [31], which assumes velocity information to be available, in terms of the success rate of passing a $0.5\,\text{m}$ radius gate.

Runs	Distance to gate center [m]	Top speed [m/s]	Time [s]
1	0.39	13.36	4.21
2	0.30	12.31	4.50
3	0.39	12.51	4.58

TABLE IV: Real-world experimental results with simulated gate visuals.

around $15\,\text{m/s}$ and $25\,\text{m/s}$. We use the knot gate motion and set $d_{\text{knot}} = 5$ to match the increased initial distance. The results for different estimates $\bar{v}_{\text{rel}}$ are provided in Table II. We observe that performance is poor if the maximum relative speed is vastly over- or underestimated. However, if the estimate is reasonable, our algorithm performs comparably to the idealized case, highlighting the robustness of our approach to different gate motions. We observe that using the true relative speed performs worse for the higher top speed, likely because that approach violates the assumption of a constant factor $k_{\text{PN}} v_{\text{rel}}$ in the original PN law (2).

The only baseline conducting experiments comparable to ours is [31], which directly commands the acceleration from the PN law to the PX4 acceleration controller. In contrast to our approach, [31] assumes a velocity estimate to be available via GPS and controls the speed to a relatively low, constant value of $5\,\text{m/s}$. For a fair comparison, we constrain the pitch angle by setting $\bar{\gamma} = 3.5°$ to achieve a similar top speed and use the same gate radius of $0.5\,\text{m}$ and gate motion parameterizations as [31], with gates moving at 1.25 to $5\,\text{m/s}$. The success rates of flying through the gates given in Table III show that our PN-informed controller clearly outperforms the approach [31] although we assume strictly fewer parts of the state estimate to be available.

B. Hardware Experiments

1) Setup: We conduct real-world experiments in an uncontrolled outdoor environment with a quadrotor with $2\,\text{kg}$ mass and $25\,\text{m/s}$ maximum speed. We implement the LOS calculation and the control algorithm in ROS2 [41] and use a UXRCE-DDS bridge for communication with PX4.

2) Simulated Gate Visuals: To evaluate our algorithm in a repeatable way, we first simulate the gate visuals using position estimates obtained via GPS (not used for control). We simulate a point cloud for the gate, project it into the camera frame, and use its bounding box to calculate the LOS.

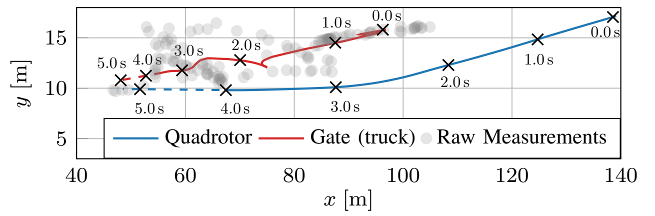

Fig. 6: Trajectories in our end-to-end hardware experiments. We emulate the gate motion using a truck and manually pull up the quadrotor right before impact. The extrapolation (dashed) based on the last velocity results in a predicted distance to the gate center of about $0.9\,\text{m}$.

This setup allows us to accurately compute the distance to the gate center as the relative position of the quadrotor and the gate are known. We use $k_{\text{PN}} = 2$ and $\bar{\gamma} = 15°$ and conduct three runs with the linear slow gate motion, each time starting from a stable hovering state. During the experiments, a wind speed of about $3\,\text{m/s}$ is measured. All runs are successful, and the results provided in Table IV demonstrate consistent performance at high speeds of more than $12\,\text{m/s}$.

3) End-to-End Experiment: Due to the challenges of moving a gate at a desired speed without motion capture, we emulate the gate movement with a truck. We use YOLO [42] for detection and Nanotrack [43], [44] for tracking and use $k_{\text{PN}} = 2$ and $\bar{\gamma} = 20°$. We smooth the tracking estimates to prevent LOS jumps that could destabilize the algorithm. The perception pipeline has a latency of about $60\,\text{ms}$. The truck drives at a speed of 8 to $10\,\text{m/s}$, and the quadrotor is initially $40\,\text{m}$ away. Once the truck is successfully detected and tracked, we start our drone racing algorithm. We pull up the quadrotor right before impact to avoid damage. By extrapolating the quadrotor trajectory based on the last velocity before pulling up, we can predict a distance of $0.9\,\text{m}$ to the center of the truck's bounding box; see Fig. 6. The duration of the approach, including the extrapolation, is $4.9\,\text{s}$, and the quadrotor reaches a speed of $23.6\,\text{m/s}$, which is only slightly below the speed achieved in [4] under controlled conditions. The experiment demonstrates that our control algorithm can be integrated with a computer vision stack on real hardware and achieve high speeds while being robust to sensor noise, delays in perception and control, wind, and unmodeled aerodynamic effects.

VI. Conclusion and Future Work

We have proposed a control algorithm for drone racing with moving gates based solely on monocular LOS and IMU measurements. We demonstrate through simulation and real-world experiments that our approach is effective at flying through moving gates at high speeds without knowing their relative position and velocity and robust to different gate motions, noise, wind, and delays. While we have shown that keeping the gate within the field of view can be enforced with our controller, the necessity for this condition is still a limitation of our method. Our experiments are focused on drone racing, but we consider this work an important step toward vision-based agile flight in dynamic environments without relying on accurate maps and full state estimation.

References

[1] P. Foehn, E. Kaufmann, A. Romero, R. Penicka, S. Sun, L. Bauersfeld, T. Laengle, G. Cioffi, Y. Song, A. Loquercio *et al.*, "Agilicious: Open-source and open-hardware agile quadrotor for vision-based flight," *Science robotics*, vol. 7, no. 67, 2022.

[2] D. Hanover, A. Loquercio, L. Bauersfeld, A. Romero, R. Penicka, Y. Song, G. Cioffi, E. Kaufmann, and D. Scaramuzza, "Autonomous Drone Racing: A Survey," *IEEE Transactions on Robotics*, vol. 40, pp. 3044–3067, 2024.

[3] E. Tal and S. Karaman, "Accurate Tracking of Aggressive Quadrotor Trajectories using Incremental Nonlinear Dynamic Inversion and Differential Flatness," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 3, pp. 1203–1218, 2020.

[4] E. Kaufmann, L. Bauersfeld, A. Loquercio, M. Müller, V. Koltun, and D. Scaramuzza, "Champion-level drone racing using deep reinforcement learning," *Nature*, vol. 620, no. 7976, pp. 982–987, 2023.

[5] C. Qin, M. S. J. Michet, J. Chen, and H. H.-T. Liu, "Time-Optimal Gate-Traversing Planner for Autonomous Drone Racing," in *IEEE Internat. Conference on Robotics and Automation (ICRA)*, 2024, pp. 8693–8699.

[6] H. Moon, J. Martinez-Carranza, T. Cieslewski, M. Faessler, D. Falanga, A. Simovic, D. Scaramuzza, S. Li, M. Ozo, C. De Wagter *et al.*, "Challenges and implemented technologies used in autonomous drone racing," *Intelligent Service Robotics*, vol. 12, pp. 137–148, 2019.

[7] S. Li, E. van der Horst, P. Duernay, C. De Wagter, and G. C. H. E. de Croon, "Visual Model-predictive Localization for Computationally Efficient Autonomous Racing of a 72-gram Drone," *Journal of Field Robotics*, vol. 37, no. 4, pp. 667–692, 2020.

[8] P. Foehn, D. Brescianini, E. Kaufmann, T. Cieslewski, M. Gehrig, M. Muglikar, and D. Scaramuzza, "Alphapilot: Autonomous drone racing," *Autonomous Robots*, vol. 46, no. 1, pp. 307–320, 2022.

[9] E. Kaufmann, A. Loquercio, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza, "Deep drone racing: Learning agile flight in dynamic environments," in *Conference on Robot Learning*. PMLR, 2018, pp. 133–145.

[10] D. Scaramuzza and Z. Zhang, "Visual-Inertial Odometry of Aerial Robots," in *Encyclopedia of Robotics*. Springer, 2019.

[11] D. C. Schedl, I. Kurmi, and O. Bimber, "An autonomous drone for search and rescue in forests using airborne optical sectioning," *Science Robotics*, vol. 6, no. 55, 2021.

[12] H. X. Pham, I. Bozcan, A. Sarabakha, S. Haddadin, and E. Kayacan, "Gatenet: An efficient deep neural network architecture for gate perception using fish-eye camera in autonomous drone racing," in *IEEE/RSJ Internat. Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 4176–4183.

[13] H. X. Pham, A. Sarabakha, M. Odnoshyvkin, and E. Kayacan, "Pencilnet: Zero-shot sim-to-real transfer learning for robust gate perception in autonomous drone racing," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 847–11 854, 2022.

[14] R. Yanushevsky and W. Boord, "Lyapunov approach to guidance laws design," *Nonlinear Analysis: Theory, Methods & Applications*, vol. 63, no. 5-7, pp. 743–749, 2005.

[15] R. Yanushevsky, *Modern missile guidance*. CRC Press, 2018.

[16] A. Romero, S. Sun, P. Foehn, and D. Scaramuzza, "Model Predictive Contouring Control for Time-Optimal Quadrotor Flight," *IEEE Transactions on Robotics*, vol. 38, no. 6, pp. 3340–3356, 2022.

[17] P. Foehn, D. Falanga, N. Kuppuswamy, R. Tedrake, and D. Scaramuzza, "Fast Trajectory Optimization for Agile Quadrotor Maneuvers with a Cable-Suspended Payload," in *Robotics: Science and Systems*, 2017.

[18] B. Zhou, F. Gao, J. Pan, and S. Shen, "Robust real-time uav replanning using guided gradient-based optimization and topological paths," in *IEEE Internat. Conference on Robotics and Automation (ICRA)*, 2020, pp. 1208–1214.

[19] P. Foehn, A. Romero, and D. Scaramuzza, "Time-Optimal Planning for Quadrotor Waypoint Flight," *Science Robotics*, vol. 6, no. 56, 2021.

[20] D. Brescianini and R. D'Andrea, "Tilt-Prioritized Quadrocopter Attitude Control," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 2, pp. 376–387, 2020.

[21] S. Sun, A. Romero, P. Foehn, E. Kaufmann, and D. Scaramuzza, "A Comparative Study of Nonlinear MPC and Differential-Flatness-Based Control for Quadrotor Agile Flight," *IEEE Transactions on Robotics*, vol. 38, no. 6, pp. 3357–3373, 2022.

[22] M. Greeff, T. D. Barfoot, and A. P. Schoellig, "A Perception-Aware Flatness-Based Model Predictive Controller for Fast Vision-Based Multirotor Flight," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 9412–9419, 2020.

[23] D. Falanga, E. Mueggler, M. Faessler, and D. Scaramuzza, "Aggressive quadrotor flight through narrow gaps with onboard sensing and computing using active vision," in *IEEE Internat. Conference on Robotics and Automation (ICRA)*, 2017, pp. 5774–5781.

[24] B. Penin, R. Spica, P. R. Giordano, and F. Chaumette, "Vision-based minimum-time trajectory generation for a quadrotor UAV," in *IEEE/RSJ Internat. Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 6199–6206.

[25] Y. Song and D. Scaramuzza, "Policy search for model predictive control with application to agile drone flight," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2114–2130, 2022.

[26] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[27] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, "OpenVINS: A Research Platform for Visual-Inertial Estimation," in *IEEE Internat. Conference on Robotics and Automation (ICRA)*, 2020, pp. 4666–4672.

[28] I. Geles, L. Bauersfeld, A. Romero, J. Xing, and D. Scaramuzza, "Demonstrating agile flight from pixels without state estimation," in *Robotics: Science and Systems (RSS)*, 2024.

[29] T. Layman, T. Fields, and O. A. Yakimenko, "Evaluation of Proportional Navigation for Multirotor Pursuit," in *AIAA Scitech 2021 Forum*, 2021.

[30] A. Kumar, A. Ojha, S. Yadav, and A. Kumar, "Real-time interception performance evaluation of certain proportional navigation based guidance laws in aerial ground engagement," *Intelligent Service Robotics*, vol. 15, no. 1, pp. 95–114, Mar. 2022.

[31] A. Bhattacharya, "Toward Increased Airspace Safety: Quadrotor Guidance for Targeting Aerial Objects," *arXiv preprint arXiv:2107.01733*, 2021.

[32] H. K. Khalil, *Nonlinear systems*. Prentice-Hall, 2002.

[33] D. Brescianini, M. Hehn, and R. D'Andrea, "Nonlinear quadrocopter attitude control: Technical report," ETH Zurich, Tech. Rep., 2013.

[34] L. Meier, D. Honegger, and M. Pollefeys, "Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms," in *IEEE Internat. Conference on Robotics and Automation (ICRA)*, 2015, pp. 6235–6240.

[35] E. Brochu, V. M. Cora, and N. De Freitas, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," *arXiv preprint arXiv:1012.2599*, 2010.

[36] A. Romero, R. Penicka, and D. Scaramuzza, "Time-Optimal Online Replanning for Agile Quadrotor Flight," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7730–7737, 2022.

[37] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. Cambridge, Mass: MIT Press, 2006.

[38] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," in *IEEE/RSJ Internat. Conference on Intelligent Robots and Systems (IROS)*, 2004, pp. 2149–2154.

[39] P. Martin and E. Salaun, "The true role of accelerometer feedback in quadrotor control," in *IEEE Internat. Conference on Robotics and Automation (ICRA)*, 2010, pp. 1623–1629.

[40] F. Nogueira, "Bayesian Optimization: Open source constrained global optimization tool for Python," 2014. [Online]. Available: https://github.com/bayesian-optimization/BayesianOptimization

[41] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science robotics*, vol. 7, no. 66, 2022.

[42] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788.

[43] Z. Chen, B. Zhong, G. Li, S. Zhang, and R. Ji, "Siamese box adaptive network for visual tracking," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 6668–6677.

[44] B. Yan, H. Peng, K. Wu, D. Wang, J. Fu, and H. Lu, "Lighttrack: Finding lightweight neural networks for object tracking via one-shot architecture search," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 15 180–15 189.