ACRORL: Learning Aggressive Quadrotor Inversion using Bidirectional Thrust

Gabriel Rodriguez[1], **Henri Sayag**[1], **Abhishek Rathod**[1],
John Stecklein[1], **Siddharth Saha**[2], **Christopher Barngrover**[2] **and Wennie Tabib**[1]
[1] Carnegie Mellon University, [2] Shield AI
{gar2, hsayag, arathod2, jsteckle, wtabib}@andrew.cmu.edu
{siddharth.saha, chris.barngrover}@shield.ai

Abstract: Bidirectional thrust grants quadrotors a second equilibrium condition and increased control authority, expanding the envelope of possible aggressive maneuvers and enabling inverted flight, perching, and sensing. Prior geometric control approaches extend differential flatness through Hopf fibration-based attitude representations to support bidirectional thrust, but struggle with actuator saturation and motor reversal delay during inversions, requiring heuristic thrust posture scheduling and waypoint tuning. We propose a learning-based framework that modulates a constant reference trajectory to perform *compact*, position-constrained quadrotor inversions while remaining compatible with traditional trajectory generation and tracking across flight regimes. Separate policies are trained via reinforcement learning for nominal-to-inverted and inverted-to-nominal transitions. In JAX-based simulation, the proposed method achieves the lowest position deviation and settling time across all evaluated baselines, reducing position root mean square error (RMSE) by 32% and settling time by 57% relative to the strongest optimization-based baseline. Hardware experiments demonstrate successful inversion across multiple yaw configurations with position RMSE below $0.35\,\mathrm{m}$, and compatibility with downstream trajectory generation and control through circular flight in both regimes. Additionally, we provide an open-source implementation of the proposed framework.

Keywords: Reinforcement Learning, Geometric Control, Aerial Robotics

1 Introduction

Traditionally, quadrotors operate around a single equilibrium: upright hover. Bidirectional electronic speed controllers (ESCs) enable a second equilibrium: inverted hover. Operating near and transitioning between the two, unlocks a range of desirable capabilities: perching on inclined surfaces, inverted perching, recovery from large disturbances in either flight regime, and the reuse of a single fixed payload for both ground and overhead tasks, thereby maximizing low size, weight, and power (SWaP) field efficiency.

Yet, executing the inversion maneuver remains challenging. Traditional differentially flat trajectory generation methods, while powerful within either flight regime, struggle with the motor reversal delays introduced by bidirectional ESCs when switching motor operating regimes due to control signal jumping [1, 2]. Prior works have demonstrated the inversion maneuver but incur significant position deviations [1, 3, 4, 5].

We address this gap in the state of the art by developing aggressive, *compact* inversion and flight. We introduce the following contributions: i) a learned reference modulation policy for aggressive inversion maneuvers leveraging bidirectional thrust, ii) a steady-state and stochastic transient thrust

model for asymmetric propellers, and iii) extensive simulation and hardware evaluation[1] alongside an open-source release[2] of the proposed approach.

2 Related Work

This section provides an overview of recent works in perching and landing, recovery mechanisms, trajectory generation and control, and thrust modeling as these are most aligned with our approach.

Perching and Landing: Recent work has explored leveraging bidirectional thrust for landing and perching tasks, including landing on inclined surfaces [6, 2], inverted perching on ceilings [1, 7], and ceiling effect perching [8]. In contrast to unidirectional thrust approaches [9, 10, 11, 12, 13], systems that exploit bidirectional thrust exhibit increased control authority along the body-frame z-axis, enabling greater contact forces during surface interaction and increased braking and acceleration capabilities during approach. However, existing inverted perching maneuvers rely on the vehicle's approach momentum to complete inversion, resulting in large position deviations. Such maneuvers are dangerous in constrained and cluttered environments, common when perching on tree limbs and branches [14, 15] or near powerlines [16]. The proposed approach addresses this limitation by developing a strategy for *compact*, in-place inversion maneuvers to and from inverted hover.

Safety and Recovery Mechanisms: Recovery after collisions [17], external disturbances [18], or throw-and-go initializations [19] is critical for mission execution. Prior works have leveraged reverse thrust to recover from single-motor failures [20] or improve quadrotor survivability under external disturbances [21, 22, 23]. However, none of these prior works intentionally utilize the inverted flight envelope, using reversible thrust only as a mechanism for recovery.

Trajectory Generation and Control: Existing works utilize a variety of control architectures for bidirectional thrust quadrotors, including model predictive control [24, 25, 26], fractional-order PID control [27, 28], cascaded quaternion-based control [5], universal UAV control [29, 4], and geometric control based on either Gram–Schmidt axis composition [1, 7] or the Hopf fibration [3]. Importantly, Watterson et al. [3] avoid singularities present in the Gram–Schmidt axis composition, critical for inversions and downstream tasks. Yet, existing trajectory generation methods are ill-suited for in-place inversions, relying on thrust posture heuristics [30], or optimizing for time while omitting key transient actuator dynamics [31]. In contrast, our policy explicitly optimizes over the *full* system dynamics, learning to minimize position deviation directly.

Thrust Modeling: Existing works have explicitly utilized symmetric propellers (i.e. 3D or bidirectional propellers) to guarantee near-symmetric performance across operating regimes [29, 4, 32, 3, 5], assumed symmetric performance despite unspecified propeller geometries [6, 23, 22], or explicitly captured asymmetry [21, 20]. Non-ideal transient dynamics during motor reversal have been represented using a deterministic reversal delay [6], a dead-zone [4, 29], a nonlinear state-dependent rate constraint [24] and, more recently, a stochastic reversal delay [32]. While Ji et al. [32] treat motor reversal delay as a stochastic variable, the approach requires complex log-normal modeling and specialized parameter identification. Similarly, the deterministic model of Bass and Desbiens [6] relies on genetic optimization to tune model parameters. In contrast, we propose a model that captures asymmetric behavior, accounts for stochastic motor reversal delay, and can be fit using limited experimental data.

3 Methodology

Notation: We adopt the Hamilton convention for quaternions with imaginary basis elements $(\mathbf{i}, \mathbf{j}, \mathbf{k})$ satisfying $(\mathbf{ij} = \mathbf{k})$ and represent quaternions as $\boldsymbol{q} = [q_w, q_x, q_y, q_z]^\top$, where $\overline{\boldsymbol{q}}$ denotes the quaternion conjugate. Scalars are written in italic typeface (m, g), while vectors are represented using lowercase bolded typeface ($\boldsymbol{r}$, $\boldsymbol{\omega}$), and matrices using uppercase bolded typeface ($\boldsymbol{R}$, $\boldsymbol{P}$). Unit basis

[1] A video of the experiments is available at https://youtu.be/7pPTKY5KKtU.

[2] The open-source `acrorl` software is available at https://github.com/rislab/acrorl.

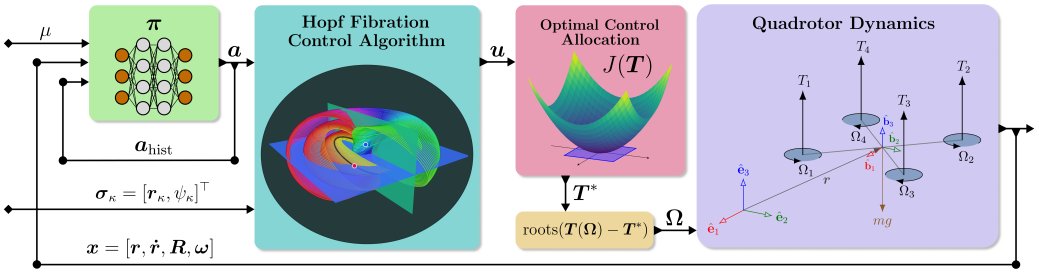


Figure 1: Overview of the proposed method. A reference modulation policy $\boldsymbol{\pi}$ (Section 3.2), activated with inversion flag μ, observes the robot's state $\boldsymbol{x}$ and a finite action history $\boldsymbol{a}_{\text{hist}}$ to produce a position reference modulation and thrust posture, $\boldsymbol{a} = [\boldsymbol{r}_{\delta\kappa}, \eta]$. These outputs and differentially flat reference $\boldsymbol{\sigma}_\kappa$ are mapped through a Hopf fibration-based control algorithm (Section 3.3) to generate the control input $\boldsymbol{u} = [f_c, \boldsymbol{\tau}]$, which is then passed to a box-constrained optimal control allocation (Section 3.4) that computes optimal thrust commands $\boldsymbol{T}^*$. Finally, these are converted to motor rates $\boldsymbol{\Omega}$ via an asymmetric thrust model and executed on the quadrotor dynamics (Section 3.1).

vectors of coordinate frames are specifically denoted with a hat $(\hat{\mathbf{e}}_i, \hat{\mathbf{b}}_i)$. The Hamiltonian quaternion product is denoted by $\otimes$, $\boldsymbol{R}(\boldsymbol{q})$ denotes the rotation matrix associated with $\boldsymbol{q}$, and $\boldsymbol{R}_x(\cdot)$, $\boldsymbol{R}_y(\cdot)$, $\boldsymbol{R}_z(\cdot)$ denote the elementary rotation matrices about the x-, y-, and z-axes, respectively, such that the ZYX Euler angle rotation matrix is $\boldsymbol{R}(\psi, \theta, \phi) = \boldsymbol{R}_z(\psi)\boldsymbol{R}_y(\theta)\boldsymbol{R}_x(\phi)$.

3.1 System Model

Quadrotor Dynamics: We model the dynamics of the quadrotor as a 3D rigid body with control inputs $\boldsymbol{u} = [f_c, \boldsymbol{\tau}]$ and state $\boldsymbol{x} = [\boldsymbol{r}, \dot{\boldsymbol{r}}, \boldsymbol{R}, \boldsymbol{\omega}]$, where f_c is the collective thrust, $\boldsymbol{\tau}$ are body torques, $\boldsymbol{r}$ is the position, $\boldsymbol{R}$ is the body-to-world rotation matrix, and $\boldsymbol{\omega}$ are body angular velocities. We define an inertial world frame $\mathcal{W}$ with axes $\hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2$, and $\hat{\mathbf{e}}_3$, where $\hat{\mathbf{e}}_3$ points opposite the direction of gravity. The body frame $\mathcal{B}$ is centered at the quadrotor's center of mass with basis vectors $\hat{\mathbf{b}}_1, \hat{\mathbf{b}}_2$, and $\hat{\mathbf{b}}_3$ comprising its axes. The pose of $\mathcal{B}$ in $\mathcal{W}$ is defined by the translation $\boldsymbol{r}$ and rotation $\boldsymbol{R}$. Letting $\boldsymbol{\mathcal{I}}$ be the inertial matrix, g the acceleration due to gravity, and m the mass of the quadrotor, its equations of motion are given by:

$$m\ddot{\boldsymbol{r}} = -mg\hat{\mathbf{e}}_3 + f_c\hat{\mathbf{b}}_3, \qquad \boldsymbol{\mathcal{I}}\dot{\boldsymbol{\omega}} = -\boldsymbol{\omega} \times \boldsymbol{\mathcal{I}}\boldsymbol{\omega} + \boldsymbol{\tau}. \tag{1}$$

Individual rotor thrusts $\boldsymbol{T}$ are mapped to the control vector $\boldsymbol{u}$ by the relationship $\boldsymbol{u} = \boldsymbol{M}(\boldsymbol{\Omega})\boldsymbol{T}$, where $\boldsymbol{M}(\boldsymbol{\Omega})$ is the mixer matrix, defined as a function of the motor rate vector $\boldsymbol{\Omega}$:

$$\boldsymbol{M}(\boldsymbol{\Omega}) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ y_1 & y_2 & y_3 & y_4 \\ -x_1 & -x_2 & -x_3 & -x_4 \\ -m_{s_1}(\operatorname{sgn}\Omega_1) & -m_{s_2}(\operatorname{sgn}\Omega_2) & m_{s_3}(\operatorname{sgn}\Omega_3) & m_{s_4}(\operatorname{sgn}\Omega_4) \end{bmatrix}, \tag{2}$$

where x_i, y_i are the positions of the motors in the body frame and m_{s_i} is the ratio of the reaction torque to rotor thrust [33], whose magnitude is dependent on motor spinning direction due to asymmetric rotor drag characteristics across positive $(+)$ and negative $(-)$ operating regimes of the propeller.

Thrust Modeling: Executing quadrotor inversions necessitates a thrust model that encompasses both the $(+)$ and $(-)$ operating regimes of the propeller. Contrary to prior works limited to the small symmetric propellers available on the market [29, 4, 32, 3, 5], we intentionally utilize larger asymmetric propellers to maximize peak thrust capability, sacrificing thrust symmetry in exchange for increased control authority and versatility. Without sacrificing model fidelity by reducing coefficient number [21] or increasing model order [20], we propose the following second order, asymmetric thrust model, where c_{T_j} is the thrust coefficient for the j-th order term [33].

$$\begin{aligned} T_i(\Omega_i) &= c_{T_{2,i}}\Omega_i|\Omega_i| + c_{T_{1,i}}\Omega_i + c_{T_{0,i}} \\ \tau_{z,i}(\Omega_i) &= m_{s_i}T_i(\Omega_i) \end{aligned} \qquad (c_{T_{j,i}}, m_{s_i}) = \begin{cases} c_{T_{j,i,(+)}}, m_{s_{i,(+)}} & \Omega_i \geq 0 \\ c_{T_{j,i,(-)}}, m_{s_{i,(-)}} & \Omega_i < 0 \end{cases} \tag{3}$$

Transient dynamics for asymmetric propellers are also non-trivial and asymmetric, especially during regime transitions where the dynamics are dominated by stochastic motor reversal delay (or dead-zone) [32, 4]. Our objective is not to explicitly model the exact timing of the reversal delay, but to capture its dominant effect on control authority.

Therefore, we adapt the piecewise first-order model presented in [4] to account for trans-regime transient dynamics variability, rather than attempting to explicitly model stochastic timing of reversal. By defining distinct operating regime specific slew rates $\alpha_{(\pm)}$ and a switching logic $\mathcal{S}$ based on a switching threshold Ω_0 and dead-zone width Δ_Ω, we devise a computationally efficient model that remains structurally simple and easy to fit to little experimental data. Conditioned on $\mathcal{S}$, the target equilibrium Ω_{set} is defined as either the true desired motor rate Ω_d, or the threshold Ω_0 to model the exponential approach to the dead-zone.

$$\mathcal{S} = (\Omega > \Omega_0 + \Delta_\Omega \wedge \Omega_\text{d} < \Omega_0) \vee (\Omega < \Omega_0 - \Delta_\Omega \wedge \Omega_\text{d} > \Omega_0) \tag{4}$$

$$\alpha = \begin{cases} \alpha_{(+)}, & \Omega \geq \Omega_0 \\ \alpha_{(-)}, & \Omega < \Omega_0 \end{cases}, \quad \Omega_{\text{set}} = \begin{cases} \Omega_0, & \mathcal{S} \\ \Omega_\text{d}, & \text{otherwise} \end{cases}, \quad \dot{\Omega} = \alpha\,(\Omega_{\text{set}} - \Omega) \tag{5}$$

To bridge the sim2real gap, we treat the steady-state and transient thrust model parameters as random variables sampled from independent uniform distributions $\mathcal{U}$ during training; namely,

$$\alpha \sim \mathcal{U}(0.75,\, 1.25)\,\alpha_{\text{nom}}, \qquad c_{T_i} \sim \mathcal{U}(0.9,\, 1.1)\,c_{T_i,\text{nom}}, \tag{6}$$

$$\Omega_0 \sim \mathcal{U}(\Omega_{0,\min},\, \Omega_{0,\max}), \qquad \Delta_\Omega \sim \mathcal{U}(\Delta_{\Omega,\min},\, \Delta_{\Omega,\max}), \tag{7}$$

where $(\cdot)_{\text{nom}}$ denotes the experimentally measured parameter. This domain randomization approach, grounded in the real-world stochasticity of the transient dynamics (see Fig. 8), yields control policies that are robust to variability in the motor response without needing complex stochastic modeling [32] or genetic algorithms for model fitting [6].

3.2 Learned Inversion Policies (π)

The quadrotor inversion task is formulated as a Markov decision process (MDP), represented by the tuple $\langle \mathcal{O}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$. The observation space $\mathcal{O}$ comprises the quadrotor state $\boldsymbol{x}$, and a finite action history $\boldsymbol{a}_{\text{hist}}$, allowing the policy to infer actuator dynamics and constraint effects. The action space $\mathcal{A}$ provides a modulation $\boldsymbol{r}_{\delta\kappa}$ to a constant flat reference $\boldsymbol{\sigma}_\kappa$, resulting in a position reference $\tilde{\boldsymbol{r}}_\kappa = \boldsymbol{r}_\kappa + \boldsymbol{r}_{\delta\kappa}$ and thrust posture η. $\mathcal{P}$ denotes the transition probability and $\mathcal{R} : \mathcal{O} \times \mathcal{A} \to \mathbb{R}$ is the reward function, with future rewards discounted by γ. For training, we define a cost function $\mathcal{C} = -\mathcal{R}$. $\mathcal{C}$ is designed to produce the desired positionally *compact* inversion while maintaining smoothness in control action for favorable sim2real transfer:

$$\mathcal{C} = \underbrace{w_{\boldsymbol{r}}\|\boldsymbol{r}\|_1 + w_{\dot{\boldsymbol{r}}} L_H(\dot{\boldsymbol{r}})}_{\text{Translational Penalties}} + \underbrace{w_{\boldsymbol{g}_b}\|\boldsymbol{g}_b - \boldsymbol{g}_{b,d}\|_2 + w_\omega L_H(\boldsymbol{\omega})}_{\text{Orientation Penalties}} + \underbrace{w_\eta\left(1-\eta^2\right) + w_{\dot{\boldsymbol{r}}_{\delta\kappa}}\|\boldsymbol{r}_{\delta\kappa,t} - \boldsymbol{r}_{\delta\kappa,t-1}\|_2}_{\text{Action Penalties}}, \tag{8}$$

where L_H is the Huber loss function [34], and $\boldsymbol{g}_b = -\boldsymbol{R}^\top \hat{\mathbf{e}}_3$ denotes the gravity vector projected into the body frame. The orientation penalty is formed using $\boldsymbol{g}_b$ to allow the Hopf Fibration-Based Control Algorithm (HFCA) to handle the yaw alignment and focus the policy's reasoning on inversion. Separate policies $\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{a}_t | \boldsymbol{x}, \boldsymbol{a}_{\text{hist}})$, parameterized by $\boldsymbol{\theta}$ and implemented as multi-layer perceptrons (MLPs), are trained using Proximal Policy Optimization (PPO) [35] for nominal-to-inverted (NTI) and inverted-to-nominal (ITN) transitions with separate training weights $\langle \boldsymbol{w}_{\text{NTI}}, \boldsymbol{w}_{\text{ITN}} \rangle$, respectively, in our JAX-based simulator `acrorl` (an adaptation of the simulator presented in [36]). Separate weighting follows the same principle as Yu et al. [1], where separate control gains are used for different flight regimes.

The action space $\mathcal{A}$ modulates a constant reference rather than providing collective thrust and body rates, or direct motor commands. This design allows the policy to explore aggressive maneuvers while anchored by an almost globally stable reference controller. To enhance exploration under the constrictions of the control structure, the action limits of $\boldsymbol{r}_{\delta\kappa}$ far exceed the controller's immediate tracking window. The policy outputs therefore act as *feedforward* commands that exploit the underlying controller, rather than as a true reference for direct tracking. Furthermore, including the thrust

posture η in $\mathcal{A}$ allows the policy to determine thrust reversal timing, removing the need for heuristics present in baseline classical methods.

3.3 Hopf Fibration-Based Control Algorithm (HFCA)

Position Control: The HFCA [37, 3] tracks a specified hybrid-mode differentially flat reference trajectory $\boldsymbol{\sigma}_\kappa(t) = [\boldsymbol{r}_\kappa(t), \psi_\kappa(t)]^\top$, conditioned on a thrust posture $\eta \in \{\pm 1\}$, where $\boldsymbol{r}_\kappa(t) \in C^3$. We define the translation error vectors $\boldsymbol{e_r} = \boldsymbol{r} - \boldsymbol{r}_\kappa$, $\boldsymbol{e_{\dot r}} = \dot{\boldsymbol{r}} - \dot{\boldsymbol{r}}_\kappa$, and $\boldsymbol{e_{\ddot r}} = \ddot{\boldsymbol{r}} - \ddot{\boldsymbol{r}}_\kappa$ when $\boldsymbol{\pi}$ is inactive, and redefine the position error as $\boldsymbol{e_r} = \boldsymbol{r} - \tilde{\boldsymbol{r}}_\kappa$ when $\boldsymbol{\pi}$ is activated by the inversion flag μ (see Fig. 1). The desired acceleration $\ddot{\boldsymbol{r}}_d$ and jerk $\dddot{\boldsymbol{r}}_d$ in the world frame $\mathcal{W}$ are given by:

$$\ddot{\boldsymbol{r}}_d = -\boldsymbol{K_r e_r} - \boldsymbol{K_{\dot r} e_{\dot r}} + \ddot{\boldsymbol{r}}_\kappa + g\hat{\boldsymbol{e}}_3, \qquad \dddot{\boldsymbol{r}}_d = -\boldsymbol{K_r e_{\dot r}} - \boldsymbol{K_{\dot r} e_{\ddot r}} + \dddot{\boldsymbol{r}}_\kappa. \tag{9}$$

The collective thrust is computed as $f_c = m\hat{\mathbf{b}}_3 \cdot \ddot{\boldsymbol{r}}_d$. The desired thrust axis $\boldsymbol{s}$ and its derivative $\dot{\boldsymbol{s}}$ are:

$$\boldsymbol{s} = \frac{\ddot{\boldsymbol{r}}_d}{\|\ddot{\boldsymbol{r}}_d\|_2}, \qquad \dot{\boldsymbol{s}} = \frac{1}{\|\ddot{\boldsymbol{r}}_d\|_2} \boldsymbol{P}\, \dddot{\boldsymbol{r}}_d, \tag{10}$$

where $\boldsymbol{P} = \boldsymbol{I} - \boldsymbol{s}\boldsymbol{s}^\top$ is the projection matrix onto the tangent space of $\mathbb{S}^2$. Given the bidirectional thrust capability, $\hat{\mathbf{b}}_3$ is not restricted to be codirectional with $\boldsymbol{s}$, only colinear. Thus, we define the desired body z-axis as $\hat{\mathbf{b}}_{3,d} = \eta\boldsymbol{s} = [a, b, c]$, flipping the direction of the desired body z-axis in the case of a commanded negative thrust posture.

Attitude Control: To cover all of $\mathrm{SO}(3)$ and handle the negative thrust case, we define 4 coordinate charts, as in [3, 30, 22]. The base quaternions $\boldsymbol{q}_{abc,N}$ and $\boldsymbol{q}_{abc,S}$ are

$$\boldsymbol{q}_{abc,N} = \frac{1}{\sqrt{2(1+c)}} \left[\, 1+c,\ -b,\ a,\ 0 \,\right]^\top, \quad \boldsymbol{q}_{abc,S} = \frac{1}{\sqrt{2(1-c)}} \left[\, -b,\ 1-c,\ 0,\ a \,\right]^\top, \tag{11}$$

while the yaw quaternion is defined as $\boldsymbol{q}_{\text{yaw}}(\psi) = [\cos(\psi/2), 0, 0, \sin(\psi/2)]^\top$, where $\psi \in \{\psi_N, \psi_S\}$, $\psi_N = \psi_\kappa$, and $\psi_S = 2\,\mathrm{atan2}(a, b) + \psi_N$. The desired orientation is then defined as $\boldsymbol{q}_d = \boldsymbol{q}_{abc,N} \otimes \boldsymbol{q}_{\text{yaw}}(\psi_N)$ in the first chart, or $\boldsymbol{q}_d = \boldsymbol{q}_{abc,S} \otimes \boldsymbol{q}_{\text{yaw}}(\psi_S)$ in the second chart. The desired angular velocity is then $\boldsymbol{\omega}_d = 2\,\mathrm{vec}\left(\boldsymbol{q}_d^{-1} \otimes \dot{\boldsymbol{q}}_d\right)$. The two other charts arise from the case that $\eta = -1$, where $\psi_N = -\psi_\kappa$ because yaw is now applied about an inverted axis. To track the desired orientation and angular velocity, we define the orientation error functions $\boldsymbol{e_R} = \mathrm{Log}(\boldsymbol{q}_d^{-1} \otimes \boldsymbol{q})$, and $\boldsymbol{e_\omega} = \boldsymbol{\omega} - \boldsymbol{R}(\boldsymbol{q}_d^{-1} \otimes \boldsymbol{q})^\top \boldsymbol{\omega}_d$, where $\boldsymbol{q}$ is the quaternion associated with $\boldsymbol{R}$.

Importantly, diverging from previous works, we use the quaternionic error function as opposed to the skew-symmetric error function [18] because the latter produces slow convergence when the error is high [38], as during an inversion. Finally, the orientation feedback is defined as

$$\boldsymbol{\tau} = -\boldsymbol{J}_R^{-\top}(\boldsymbol{e}_R)\boldsymbol{K_R e_R} - \boldsymbol{K_\omega e_\omega}, \tag{12}$$

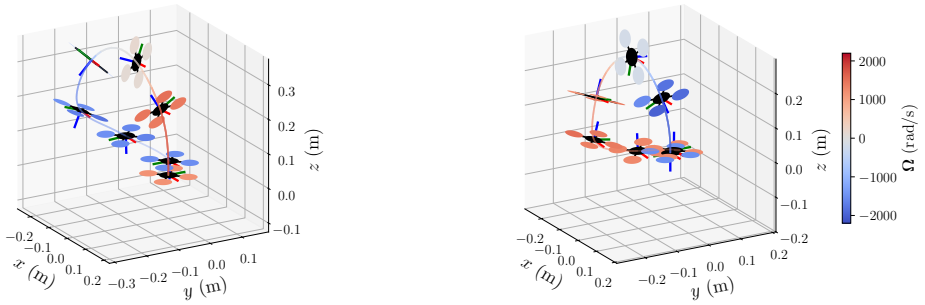
where $\boldsymbol{J}_R^{-\top}$ is the inverse right-Jacobian implemented as in [39], defining $\boldsymbol{u} = [f_c, \boldsymbol{\tau}]$ illustrated in Fig. 1. We include $\boldsymbol{J}_R^{-\top}$ in the feedback law to account for the curvature of the $\mathrm{SO}(3)$ manifold.

Chart Switching Logic: For nominal hybrid-mode differentially flat trajectory tracking, we follow the chart switching logic presented in [37, 3], switching charts at the equator ($c = 0$) using the yaw offset $2\,\mathrm{atan2}(a, b)$.

However, step thrust posture commands, such as $c = 1$ to $c = -1$ via a transition in thrust posture sign, generate large instantaneous orientation error that drives high-speed inversion. Such commands often necessitate chart transitions at locations where the offset $2\,\mathrm{atan2}(a, b)$ is undefined, requiring separate logic to handle this case. Hence, we default to no offset, which in the $c = 1$ to $c = -1$ case, corresponds to $\boldsymbol{q}_{abc,S} = \mathbf{i}$. Since $\mathbf{i} \otimes [0, \cos(-\zeta), \sin(-\zeta), 0]^\top \otimes \bar{\mathbf{i}} = [0, \cos(\zeta), \sin(\zeta), 0]^\top$, it follows that no additional yaw offset is introduced, yielding $\psi_S = \psi_N = -\psi_\kappa$.

3.4 Optimal Control Allocation (OCA)

Without accounting for actuator limits, $\boldsymbol{u}$ is nominally mapped to desired motor thrusts $\boldsymbol{T}$ using direct inversion of the mixer matrix $\boldsymbol{M}(\boldsymbol{\Omega})$: $\boldsymbol{T} = \boldsymbol{M}(\boldsymbol{\Omega})^{-1}\boldsymbol{u}$. Jothiraj et al. [4] cast



(a) Nominal to inverted (NTI) transition

(b) Inverted to nominal (ITN) transition

Figure 2: Still-frame visualizations of the learned quadrotor inversion trajectories in simulation. The colors encode the rotor angular rates throughout the trajectory execution (red for positive thrust and blue for negative thrust). Compared to the (a) NTI transition, the (b) ITN transition exhibits a tighter maneuver profile, which is consistent with the asymmetric thrust profile during reversal.

this control allocation problem as a weighted least squares formulation with cost of the form $J(\boldsymbol{T}) = \frac{1}{2}\|\boldsymbol{W}(\boldsymbol{M}(\boldsymbol{\Omega})\boldsymbol{T} - \boldsymbol{u})\|^2$, where $\boldsymbol{W} \in \mathbb{R}^{4\times4}$ is a weighting matrix used to prioritize allocating thrust to roll and pitch body torques for stabilization under saturation conditions. Using the same relative weighting, we augment the cost function with a Tikhonov regularization term and parameter λ: $J(\boldsymbol{T}) = \frac{1}{2}\|\boldsymbol{W}(\boldsymbol{M}(\boldsymbol{\Omega})\boldsymbol{T} - \boldsymbol{u})\|^2 + \frac{\lambda}{2}\|\boldsymbol{T} - \boldsymbol{T}_{\text{prev}}\|^2$, and remove explicit slew-rate constraints as they can impose overly conservative limits on transient actuator response [4]. Expanding the cost and dropping constant terms, we obtain a similar box-constrained quadratic objective [40]:

$$\boldsymbol{T}^* = \arg\min_{\boldsymbol{T}} \tfrac{1}{2}\boldsymbol{T}^\top \boldsymbol{H}\boldsymbol{T} + \boldsymbol{f}^\top \boldsymbol{T} \quad \text{s.t.} \quad \boldsymbol{T}_{\min} \le \boldsymbol{T} \le \boldsymbol{T}_{\max}, \tag{13}$$

where $\boldsymbol{H} = \boldsymbol{M}(\boldsymbol{\Omega})^\top \boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{M}(\boldsymbol{\Omega}) + \lambda \boldsymbol{I}$ and $\boldsymbol{f} = -\left(\boldsymbol{M}(\boldsymbol{\Omega})^\top \boldsymbol{W}^\top \boldsymbol{W}\boldsymbol{u} + \lambda \boldsymbol{T}_{\text{prev}}\right)$. Contrary to [4], we solve this optimization problem using Projected Gradient Descent (PGD) [40] for a fixed k iterations to comply with JAX's static-graph compilation methods, enabling efficient batching during training [41].

4 Experimental Design and Results

Baseline Inversion Strategies: The proposed method was compared against three baselines: step thrust posture command with direct inversion control allocation, step thrust posture with optimal control allocation, and a minimum-snap trajectory baseline.

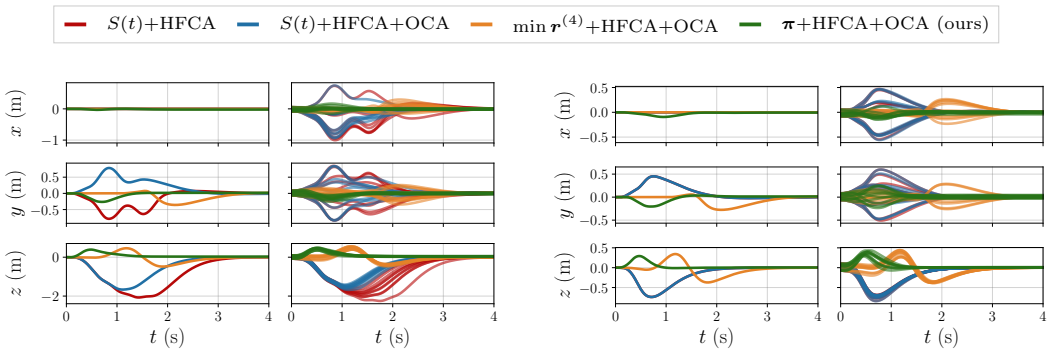
The step thrust posture command consists of a constant position reference and a transition from η_{init} to $-\eta_{\text{init}}$, inverting $\hat{\mathbf{b}}_{3,d}$. The minimum-snap trajectory baseline follows the method outlined in [30]. Inversion trajectories are defined using three waypoints: the start position, an increase in z by a heuristically defined and tuned height, and a return to the start position. Equal time is allocated between each waypoint and a change in thrust posture is applied at the second waypoint to satisfy free-fall constraints.

Simulation: The objective of the simulation experiment is to characterize inversions from both nominal hover $\boldsymbol{R}_{\text{inv}} = \boldsymbol{I}$, and inverted hover $\boldsymbol{R}_{\text{inv}} = \boldsymbol{R}_x(\pi)$ with an initial state near $\boldsymbol{x}_{\text{init}} = [\mathbf{0}, \mathbf{0}, \boldsymbol{R}_{\text{inv}}, \mathbf{0}]$, as would be experienced during deployment. The experiment uses n=20 parallel simulation environments initialized from random initial conditions according to

$$\boldsymbol{r}_{\text{init}} \sim \mathcal{U}(-\Delta\boldsymbol{r}, \Delta\boldsymbol{r}), \quad \dot{\boldsymbol{r}}_{\text{init}} \sim \mathcal{N}(\mathbf{0}, \sigma_{\dot{r}}^2 \boldsymbol{I}), \quad \boldsymbol{R}_{\text{init}} \sim \boldsymbol{R}(\psi, \theta, \phi)\boldsymbol{R}_{\text{inv}}, \quad \boldsymbol{\omega}_{\text{init}} \sim \mathcal{N}(\mathbf{0}, \sigma_\omega^2 \boldsymbol{I}), \tag{14}$$

where $\Delta\boldsymbol{r} = [0.1, 0.1, 0.1]^\top \text{m}$ is the maximum position perturbation, while $\sigma_{\dot{r}} = 0.1\,\text{m}\,\text{s}^{-1}$ and $\sigma_\omega = 0.1\,\text{rad}\,\text{s}^{-1}$ are standard deviations in velocity and body angular rate distributions, respectively. The yaw angle is sampled according $\psi \sim \mathcal{U}(-\pi, \pi)$ rad, while roll and pitch angles are sampled according to $\phi, \theta \sim \mathcal{U}(-0.1, 0.1)$ rad.

Single-rollout still frames are illustrated in Fig. 2 while associated trajectories are shown in Fig. 3. All quantitative simulation experiment results are reported in Table 1. We report settling times



(a) Nominal to inverted (NTI) transitions, single rollout (left), parallel rollouts (right)

(b) Inverted to nominal (ITN) transitions, single rollouts (left), parallel rollouts (right)

Figure 3: Method comparison in simulation. Our approach consistently outperforms classical baseline inversion strategies for both NTI and ITN transitions in our JAX-based simulator `acrorl`. Results show single (n=1) and parallel (n=20) rollout performance across methods. For the sake of clarity, single rollout inversions (left) are shown from ideal hover conditions.

inversion type	**method**	$\bar{e}_{\boldsymbol{r}}$	t_s	$\max(\delta r_x)$	$\max(\delta r_y)$	$\max(\delta r_z)$
nominal ↓ inverted	$S(t)$+HFCA	0.9349	2.1720	1.0036	0.8768	2.2526
	$S(t)$+HFCA+OCA	0.6896	1.4280	0.9408	0.8589	1.7733
	min $\boldsymbol{r}^{(4)}$+HFCA+OCA	0.2195	2.0760	0.2980	0.3443	0.5637
	$\boldsymbol{\pi}$+HFCA+OCA (ours)	0.1492	0.8840	0.1877	0.2601	0.4955
inverted ↓ nominal	$S(t)$+HFCA	0.3216	0.8690	0.5486	0.6000	0.8650
	$S(t)$+HFCA+OCA	0.3196	0.8640	0.5599	0.5999	0.8642
	min $\boldsymbol{r}^{(4)}$+HFCA+OCA	0.1920	2.0400	0.2791	0.2891	0.4437
	$\boldsymbol{\pi}$+HFCA+OCA (ours)	0.1164	0.8480	0.1739	0.2802	0.4339

Table 1: Parallelized method comparison in JAX-based simulation, where $\bar{e}_{\boldsymbol{r}}$ is the position RMSE in m, δr_x, δr_y, and δr_z are deviations in x-, y-, and z- positions also in m, respectively, and t_s is the $\boldsymbol{g}_b$ settling time within a 10° cone of $\boldsymbol{g}_{b,d}$, measured in s. Results shown for n=20 parallel rollouts. Best results per transition type are highlighted in green, second best in orange.

t_s, position root mean square error (RMSE) $\bar{e}_{\boldsymbol{r}}$, and maximum position deviations $\max(\delta r_\xi)$ for $\xi \in x, y, z$. The settling time is defined as the time required for the projected gravity vector to enter and remain within a 10° cone of the target $\boldsymbol{g}_{b,d}$ (see Fig. 4), and define position error metrics relative to the world origin $\boldsymbol{r} = \mathbf{0} \in \mathbb{R}^3$ for all methods, including the minimum-snap trajectory baseline. Although counterintuitive for a trajectory tracking method, this choice is deliberate: the objective is not pure tracking, but minimal position deviation from the starting hover position.

The learned policy consistently outperforms both baselines in every performance metric across all test cases. We attribute this to learning and optimizing over the true closed-loop system dynamics, specifically the non-linearities and stochasticity of our transient thrust model. In contrast, the step command relies on heuristically tuned gains, while the minimum snap-planner re-parameterizes heuristic tuning to waypoints for a trajectory that minimizes snap as a proxy for true dynamic feasibility, without considering actuator constraints.

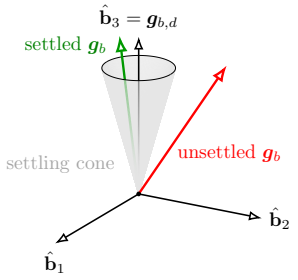
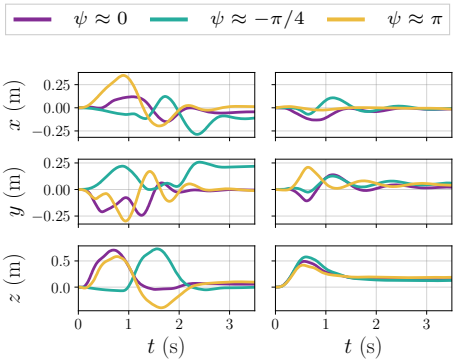
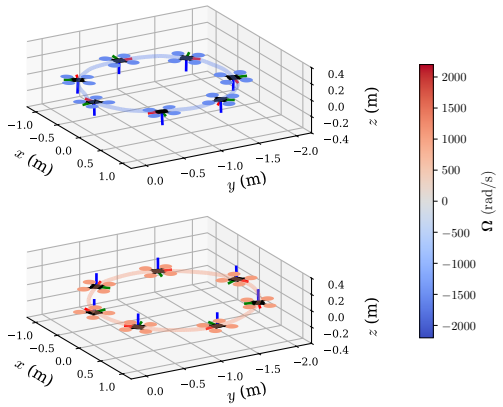


Figure 4: A visualization of the settling cone for an NTI transition.

Although the step command can produce competitive settling times, it often results in motor saturation and poor position tracking, as seen in Fig. 3a. Increased performance of the step command aided by OCA highlights its effectiveness under motor saturation. It is noted that the policy exhibits small steady-state errors (3.52 cm and 4.81 cm) for ITN and NTI transitions, respectively, which we attribute to competing terms in the reward function, a noted limitation of RL-based quadrotor control [42].



(a) Hardware inversions at varying initial yaw angles for NTI (left) and ITN (right) transitions.



(b) Circular trajectories after a NTI transition (top) and an ITN transition (bottom). Colors encode rotor angular rates along the trajectory.

Figure 5: Hardware experiments demonstrating the proposed method. (a) Inversion trajectories at varying yaw angles demonstrate that the proposed method can reliably invert at arbitrary yaw angles, as required for following general trajectories before and after transition. (b) Circular trajectories flown after each transition type demonstrate that the proposed method is compatible with the HFCA.

inversion type	**initial yaw**	$\bar{e}_{\boldsymbol{r}}$	t_s	$\max(\delta r_x)$	$\max(\delta r_y)$	$\max(\delta r_z)$
nominal	$\psi \approx 0$	0.2557	2.2040	0.1501	0.2430	0.7074
$\downarrow$	$\psi \approx -\pi/4$	0.3233	2.8566	0.2872	0.2577	0.7309
inverted	$\psi \approx \pi$	0.2737	2.3222	0.3498	0.2975	0.5834
inverted	$\psi \approx 0$	0.2236	1.3681	0.1320	0.1389	0.4924
$\downarrow$	$\psi \approx -\pi/4$	0.2431	1.2754	0.1080	0.1267	0.5766
nominal	$\psi \approx \pi$	0.2563	1.2353	0.0823	0.2090	0.4200

Table 2: Hardware inversion results for the proposed method across three initial yaw configurations, where $\bar{e}_{\boldsymbol{r}}$ is the position RMSE in m, δr_x, δr_y, and δr_z are deviations in x-, y-, and z-positions also in m, respectively, and t_s is the $\boldsymbol{g}_b$ settling time within a 10° cone of $\boldsymbol{g}_{b,d}$, measured in s. Best results per transition type are highlighted in green, second best in orange.

Hardware: Two hardware experiments were performed to validate the proposed method. The first tested the NTI and ITN transitions at varying yaw angles, validating inversion position deviation while maintaining yaw using the HFCA. The second validated the full pipeline by performing yaw-tracking circular trajectories before and after inversions. Results from both experiments are visualized in Fig. 5, and quantitatively summarized in Table 2. Since hardware flips are not initialized at the true world origin, position error metrics are computed relative to the inversion starting point $\boldsymbol{r}_{\text{init}}$ over the same duration as the simulation rollouts, enabling reasonable comparison.

The proposed method transfers favorably from simulation to hardware, achieving successful zero-shot quadrotor inversions. System performance trends observed in simulation are preserved, suggesting the simulator successfully captures the dominant closed-loop system behavior.

More specifically, the average maximum position deviations increase by at most 40% relative to the simulation for NTI transitions, while ITN deviations compare favorably on hardware, with $\max(\delta r_{x,y})$ decreasing by roughly 41%. This asymmetry, along with the larger increase in mean settling time for NTI (178%) relative to ITN (52%), is consistent with adverse propeller geometry in the inverted regime inducing rotor-body-rotor aerodynamic interactions that degrade performance and are absent from the simulation environments. The $\sim$100% increases in position RMSE are primarily attributable to steady-state offsets amplified on hardware, as the reference controller lacks an integrator term and is subject to discrepancies between thrust model coefficients identified with the test stand and those on the vehicle. This is particularly evident in the ITN z-position traces of Fig. 5a.

For the circular trajectory experiment, the proposed HFCA chart-switching logic enables world-frame yaw tracking throughout the trajectory, both before and after inversion transitions, demonstrating that the full pipeline is suitable for agile trajectory following across flight regimes.

5 Conclusion

We presented AcroRL, a learning-based method to perform *compact* quadrotor inversions using bidirectional thrust. By training separate reference modulation policies using PPO, we achieve in-place inversions while remaining compatible with traditional geometric control techniques for executing arbitrary trajectories before and after inversion. Careful modeling of asymmetric steady-state thrust characteristics and stochastic reversal delay during training yields favorable sim2real transfer without requiring complex stochastic modeling or genetic optimization.

In simulation, our method consistently outperforms baseline classical methods in position RMSE, maximum position deviation, and settling times to inversion. More specifically, our method reduces RMSE by at least 32% and settling time by 57% relative to a minimum snap optimization-based baseline. On hardware, we perform successful inversions at varying yaw angles with position RMSE below $0.35\,\mathrm{m}$, and demonstrate our framework's ability to perform a yaw-tracking circular trajectory in both flight regimes, before and after inversion.

Limitations: A key limitation of the approach is the observed sim2real gap, specifically concerning unmodeled aerodynamic effects during inverted-regime asymmetric propeller operation. Additionally, the absence of an integrator in the reference controller produces steady-state position offsets on hardware. Finally, the current reward structure is tailored to inversion maneuvers, and extending to more general acrobatic behaviors would require reward reshaping and retuning.

Acknowledgments

This work was supported in part by the Army Research Laboratory and the Army Research Office under contract/grant number W911NF-25-2-0153, AI2C Seed grant, and gift funding from Shield AI.

References

[1] P. Yu, G. Chamitoff, and K. Wong. Perching upside down with bi-directional thrust quadrotor. In *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1697–1703, 2020. doi:10.1109/ICUAS48674.2020.9213946.

[2] J. Bass, I. Tunney, and A. L. Desbiens. Adaptative friction shock absorbers and reverse thrust for fast multirotor landing on inclined surfaces. *IEEE Robotics and Automation Letters*, 7(3): 6701–6708, 2022. doi:10.1109/LRA.2022.3176102.

[3] M. Watterson, A. Zahra, and V. Kumar. Geometric control and trajectory optimization for bidirectional thrust quadrotors. In J. Xiao, T. Kröger, and O. Khatib, editors, *Proceedings of the 2018 International Symposium on Experimental Robotics*, pages 165–176, Cham, 2020. Springer International Publishing. ISBN 978-3-030-33950-0.

[4] W. Jothiraj, I. Sharf, and M. Nahon. Control allocation of bidirectional thrust quadrotor subject to actuator constraints. In *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 932–938, 2020. doi:10.1109/ICUAS48674.2020.9214036.

[5] M. Maier. Bidirectional thrust for multirotor mavs with fixed-pitch propellers. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8, 2018. doi:10.1109/IROS.2018.8593836.

[6] J. Bass and A. L. Desbiens. Improving multirotor landing performance on inclined surfaces using reverse thrust. *IEEE Robotics and Automation Letters*, 5(4):5850–5857, 2020. doi:10.1109/LRA.2020.3010208.

[7] P. Yu and K. Wong. An implementation framework for vision-based bat-like inverted perching with bi-directional thrust quadrotor. *International Journal of Micro Air Vehicles*, 14: 1–12, 2022. doi:10.1177/17568293211073672. URL https://doi.org/10.1177/17568293211073672.

[8] M. Gong, Z. Shao, B. Li, J. Wang, and Y. Wang. Aggressive perching trajectory planning and control for quadrotor. *IFAC-PapersOnLine*, 59(20):1243–1248, 2025.

[9] B. Habas, A. Brown, D. Lee, M. Goldman, and B. Cheng. From ceilings to walls: Universal dynamic perching of quadrotors on surfaces with variable orientations. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 288–294, 2025. doi:10.1109/ICRA55743.2025.11128577.

[10] B. Habas and B. Cheng. From flies to robots: Inverted landing in small quadcopters with dynamic perching. *IEEE Transactions on Robotics*, 41:1773–1790, 2025. doi:10.1109/TRO.2025.3543263.

[11] B. Habas, J. W. Langelaan, and B. Cheng. Inverted landing in a small aerial robot via deep reinforcement learning for triggering and control of rotational maneuvers. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3368–3375, 2023. doi:10.1109/ICRA48891.2023.10160376.

[12] J. Mao, G. Li, S. Nogar, C. Kroninger, and G. Loianno. Aggressive visual perching with quadrotors on inclined surfaces. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5242–5248, 2021. doi:10.1109/IROS51168.2021.9636690.

[13] Y. Zou, H. Li, Y. Ren, W. Xu, Y. Li, Y. Cai, S. Zhou, and F. Zhang. Perch a quadrotor on planes by the ceiling effect. In *2023 IEEE 19th International Conference on Automation Science and Engineering (CASE)*, pages 1–7, 2023. doi:10.1109/CASE56687.2023.10260542.

[14] Y. Li, D. Liu, Y. Zhu, J. Zhang, Y. Luo, Z. Wang, C. Liu, and J. Zhao. Design and control of a perching drone inspired by the prey-capturing mechanism of venus flytrap. *arXiv preprint arXiv:2509.13249*, 2025.

[15] Y. Li, D. Li, Y. Zhang, L. Huang, S. Wang, and B. Cai. Tail-mav: Design and control of a perching micro aerial vehicle inspired by the tail-suspended behavior of primates. In *2025 7th International Symposium on Robotics and Intelligent Manufacturing Technology (ISRIMT)*, pages 372–376, 2025. doi:10.1109/ISRIMT67769.2025.11413171.

[16] J. L. Paneque, J. R. M.-d. Dios, A. Ollero, D. Hanover, S. Sun, A. Romero, and D. Scaramuzza. Perception-aware perching on powerlines with multirotors. *IEEE Robotics and Automation Letters*, 7(2):3077–3084, 2022. doi:10.1109/LRA.2022.3145514.

[17] A. Battiston, I. Sharf, and M. Nahon. Attitude estimation for collision recovery of a quadcopter unmanned aerial vehicle. *The International Journal of Robotics Research*, 38(10-11):1286–1306, 2019.

[18] M. W. Mueller. Multicopter attitude control for recovery from large disturbances, 2018. URL https://arxiv.org/abs/1802.09143.

[19] M. Faessler, F. Fontana, C. Forster, and D. Scaramuzza. Automatic re-initialization and failure recovery for aggressive flight with a monocular vision-based quadrotor. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1722–1729, 2015. doi:10.1109/ICRA.2015.7139420.

[20] F. Liao, D. Neo, K. Peng, D. Jia, A. Yash, and W. Liu. Reversible thrust-based fault tolerant control for quadrotor uavs against motor failure. In *2025 European Control Conference (ECC)*, pages 1531–1536, 2025. doi:10.23919/ECC65951.2025.11187187.

[21] Z. Chen, S. Mo, B. Zhang, J. Li, and H. Cheng. Robust control for bidirectional thrust quadrotors under instantaneously drastic disturbances. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6186–6192, 2024. doi:10.1109/ICRA57147.2024.10611241.

[22] Y. Zhao, M. Lyu, and H. Huang. A novel anti-disturbance control framework for bidirectional quadrotors. In *2025 IEEE 19th International Conference on Control & Automation (ICCA)*, pages 310–315, 2025. doi:10.1109/ICCA65672.2025.11129814.

[23] Y. Zhao, M. Lyu, C. Li, and H. Huang. Bidirectional thrust control for quadrotor safety. *IEEE Robotics and Automation Letters*, 11(3):2650–2657, 2026. doi:10.1109/LRA.2026.3653327.

[24] J. Wehbeh and I. Sharf. An mpc formulation on $so(3)$ for a quadrotor with bidirectional thrust and nonlinear thrust constraints. *IEEE Robotics and Automation Letters*, 7(2):4945–4952, 2022. doi:10.1109/LRA.2022.3154021.

[25] J. Wehbeh and I. Sharf. Geometric mpc techniques for reduced attitude control on quadrotors with bidirectional thrust. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 12330–12335, 2022. doi:10.1109/IROS47612.2022.9982250.

[26] J. Wehbeh and I. Sharf. Nonlinear scenario-based model predictive control for quadrotors with bidirectional thrust. *International Journal of Robust and Nonlinear Control*, 34(18):12450–12475, 2024. doi:https://doi.org/10.1002/rnc.7627. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rnc.7627.

[27] L. Xu, Z. Cai, Y. Wang, and Z. Shen. The control method of a quadrotor driven by bidirectional electronic speed controllers. *Scientific Reports*, 14(1):19532, 2024. doi:10.1038/s41598-024-70681-3. URL https://doi.org/10.1038/s41598-024-70681-3.

[28] L. Xu, Z. Cai, Y. Wang, R. Cai, and Y. Liu. The motion planning, learning and control of a bidirectional thrust quadrotor with special tasks. *Journal of Electrical Engineering & Technology*, 21(3):3043–3059, 2026. doi:10.1007/s42835-026-02591-5. URL https://doi.org/10.1007/s42835-026-02591-5.

[29] W. Jothiraj, C. Miles, E. Bulka, I. Sharf, and M. Nahon. Enabling bidirectional thrust for aggressive and inverted quadrotor flight. In *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 534–541, 2019. doi:10.1109/ICUAS.2019.8798234.

[30] K. Mao, J. Welde, M. A. Hsieh, and V. Kumar. Trajectory planning for the bidirectional quadrotor as a differentially flat hybrid system. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1242–1248, 2023. doi:10.1109/ICRA48891.2023.10160320.

[31] K. Mao, I. Spasojevic, M. A. Hsieh, and V. Kumar. Toppquad: Dynamically-feasible time-optimal path parametrization for quadrotors. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 13136–13143. IEEE, 2024.

[32] S. Ji, Y. Wang, and X. He. Flipping an upright-inverted bimodal bicopter uav: Attitude control and optimization. *IEEE/ASME Transactions on Mechatronics*, 30(6):5063–5073, 2025. doi:10.1109/TMECH.2025.3549030.

[33] R. Mahony, V. Kumar, and P. Corke. Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. *IEEE Robotics & Automation Magazine*, 19(3):20–32, 2012. doi:10.1109/MRA.2012.2206474.

[34] P. J. Huber. Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics*, 35(1):73 – 101, 1964. doi:10.1214/aoms/1177703732. URL https://doi.org/10.1214/aoms/1177703732.

[35] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

[36] J. Heeg, Y. Song, and D. Scaramuzza. Learning quadrotor control from visual features using differentiable simulation, 2025. URL https://arxiv.org/abs/2410.15979.

[37] M. Watterson and V. Kumar. Control of quadrotors using the hopf fibration on so(3). In N. M. Amato, G. Hager, S. Thomas, and M. Torres-Torriti, editors, *Robotics Research*, pages 199–215, Cham, 2020. Springer International Publishing. ISBN 978-3-030-28619-4.

[38] A. Spitzer and N. Michael. Rotational error metrics for quadrotor control. *CoRR*, abs/2011.11909, 2020. URL https://arxiv.org/abs/2011.11909.

[39] Z. Nurlanov. $so(3)$ transformations and jacobian of extended log map. Technical report, Legged Robotics (Kindr), 2024. URL https://github.com/nurlanov-zh/so3_log_map.

[40] D. Bertsekas. *Nonlinear Programming*. Athena scientific optimization and computation series. Athena Scientific, 2016. ISBN 9781886529052. URL https://books.google.com/books?id=rC1EEAAAQBAJ.

[41] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, Y. Katariya, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/jax-ml/jax.

[42] G. Cano Lopes, M. Ferreira, A. da Silva Simões, and E. Luna Colombini. Intelligent control of a quadrotor with proximal policy optimization reinforcement learning. In *2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE)*, pages 503–508, 2018. doi:10.1109/LARS/SBR/WRE.2018.00094.

[43] J. Milnor. Analytic proofs of the "hairy ball theorem" and the brouwer fixed point theorem. *The American Mathematical Monthly*, 85(7):521–524, 1978. ISSN 00029890, 19300972. URL http://www.jstor.org/stable/2320860.

[44] D. W. Lyons. An elementary introduction to the hopf fibration, 2022. URL https://arxiv.org/abs/2212.01642.

[45] V. Ganesh, S. Khidkikar, D. Velarde, J. Cox, C. Barngrover, and N. Michael. EdgeOS: A high-performance middleware framework for autonomous robotics. https://shield.ai/hivemind-edgeos-a-game-changer-for-autonomous-robotics/, 2025.

[46] O. R. developers. Onnx runtime. https://onnxruntime.ai/, 2021. Version: 1.20.1.

[47] D. Mellinger. *Trajectory generation and control for quadrotors*. PhD thesis, 2012. URL https://www.proquest.com/dissertations-theses/trajectory-generation-control-quadrotors/docview/1018692309/se-2. Copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2023-03-03.

A Topological Foundations of the Hopf Fibration-Based Control Algorithm

Modeling a quadrotor as a differentially flat hybrid system with flat variables $\eta \in \{\pm 1\}, (\boldsymbol{r}, \psi)$, is possible because thrust is aligned with $\eta\hat{\mathbf{b}}_3$, the product of thrust posture and the body z-axis. This alignment allows the attitude to be decomposed into a thrust direction $\boldsymbol{s} = \eta\hat{\mathbf{b}}_3 \in \mathbb{S}^2$ and a residual rotation about this axis, $\psi \in \mathbb{S}^1$, which represents yaw. However, this decoupling attempts to assign a globally consistent yaw direction in $\mathbb{S}^1$ to every thrust direction in $\mathbb{S}^2$. Since $\mathbb{S}^2$ cannot be covered by a single smooth coordinate chart, requiring at least two charts with non-affine coordinate transformations, this global assignment breaks down, yielding a singularity where chart definitions conflict (cf. hairy ball theorem [43]).

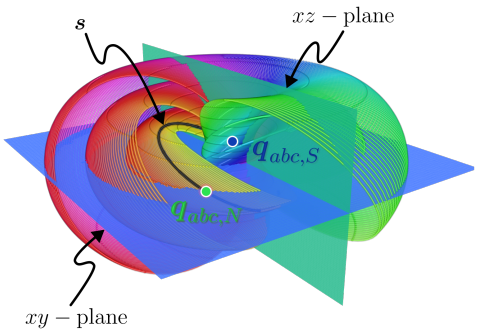


Figure 6: Visualization of the HFCA coordinate charts through stereographic projection.

This topology is described by the Hopf fibration $\mathbb{S}^1 \hookrightarrow \mathbb{S}^3 \xrightarrow{\boldsymbol{p}} \mathbb{S}^2$, where $\mathbb{S}^3$, viewed as the space of unit quaternions representing attitude, is projected to thrust directions by the Hopf map [44]

$$\boldsymbol{p} : \mathbb{S}^3 \to \mathbb{S}^2, \qquad \boldsymbol{q} \mapsto \mathrm{Im}(\boldsymbol{q} \otimes \mathbf{k} \otimes \overline{\boldsymbol{q}}). \tag{15}$$

For a given thrust direction $\boldsymbol{s} \in \mathbb{S}^2$, the fiber $\boldsymbol{p}^{-1}(\boldsymbol{s})$ represents all attitudes that have body z-axis equal to the thrust direction $\boldsymbol{s}$. The fiber bundle structure informs us that this fiber is circular. Specifically, we can observe the fiber is invariant to yaw rotation about the body z-axis, since for all $\boldsymbol{q} \in \boldsymbol{p}^{-1}(\boldsymbol{s}), \psi \in \mathbb{S}^1$ we have

$$\boldsymbol{p}(\boldsymbol{q} \otimes e^{\psi\mathbf{k}}) = \mathrm{Im}(\boldsymbol{q} \otimes e^{\psi\mathbf{k}} \otimes \mathbf{k} \otimes e^{-\psi\mathbf{k}} \otimes \overline{\boldsymbol{q}}) = \mathrm{Im}(\boldsymbol{q} \otimes \mathbf{k} \otimes \overline{\boldsymbol{q}}) = \boldsymbol{p}(\boldsymbol{q}), \tag{16}$$

hence right multiplication by $e^{\psi\mathbf{k}}$ parameterizes the fiber, thus $\boldsymbol{q} \otimes e^{\psi\mathbf{k}} \in \boldsymbol{p}^{-1}(\boldsymbol{s})$.

In this context the Hopf Fibration-Based Control Algorithm (HFCA), first presented by Watterson and Kumar [37], expanded to the bidirectional thrust case by Watterson et al. [3], and re-introduced with modifications for better inversion performance in Section 3.3, maps a desired yaw $\psi \in \mathbb{S}^1$, and thrust direction $\boldsymbol{s} \in \mathbb{S}^2$, to desired attitude in two steps. First, a point along the fiber $\boldsymbol{p}^{-1}(\boldsymbol{s})$ is selected to map $\mathbf{k}$ to $\eta\boldsymbol{s}$. Second, the yaw angle is incorporated as a rotation about the z-axis, represented as $e^{\frac{\psi}{2}\mathbf{k}}$.

As noted, two charts are necessary to define this selection globally, defined in (11). The primary chart $\boldsymbol{q}_{abc,N}$ is drawn from the intersection of the xy-plane with the fiber, hence the rotation is minimal and a tilt without twist. The secondary chart $\boldsymbol{q}_{abc,S}$ is constructed from the intersection of the xz-plane with the fiber. Since the rotation is generally not minimal, additional twist in yaw can occur. Such is minimized by shifting the yaw reference to smoothen the transition from the primary to secondary chart. The desired attitude is then the composition of the chart derived quaternion with the yaw rotation.

The charts $\boldsymbol{q}_{abc,N}$ and $\boldsymbol{q}_{abc,S}$ are for $\mathbb{S}^2$. Thus, which is selected only depends on the desired body z-axis, $\hat{\mathbf{b}}_{3,d} = [a, b, c] \in \mathbb{S}^2$. Because the North chart $\boldsymbol{q}_{abc,N}$ and South chart $\boldsymbol{q}_{abc,S}$ are defined on $\mathbb{S}^2 \setminus \{(0, 0, -1)\}$ and $\mathbb{S}^2 \setminus \{(0, 0, 1)\}$ respectively, we avoid their singularities by choosing to switch at the equator ($c = 0$) with a hysteresis to ensure the active chart is defined in the currently occupied hemisphere, as in [3].

As a measure against the twist introduced by the secondary chart $\boldsymbol{q}_{abc,S}$ we consider

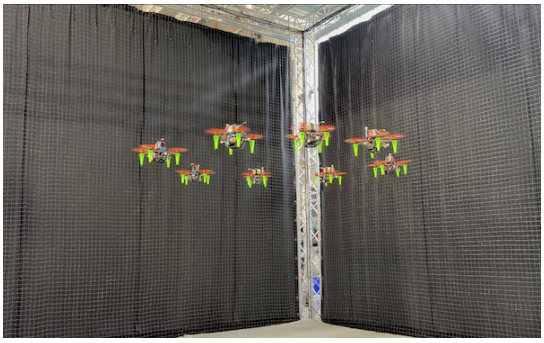
$$\boldsymbol{q}_{abc,N} \otimes \boldsymbol{q}_{\text{yaw}}(\psi_N) = \boldsymbol{q}_{abc,S} \otimes \boldsymbol{q}_{\text{yaw}}(\psi_S) \tag{17}$$

Left-multiplying by $\boldsymbol{q}_{abc,S}$ gives

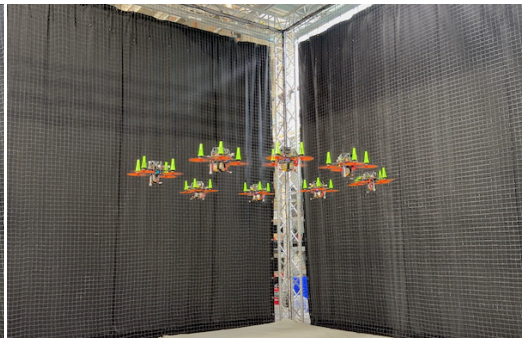
$$\overline{\boldsymbol{q}}_{abc,S} \otimes \boldsymbol{q}_{abc,N} = \boldsymbol{q}_{\text{yaw}}(\psi_S - \psi_N), \tag{18}$$

(a) Top-down view of the aerial platform

(b) Upside-down view of the aerial platform



(c) Nominal circle trajectory



(d) Inverted circle trajectory

Figure 7: Custom quadrotor platform developed for the hardware experiments. (a) illustrates the top view of the drone with 3D printed landing legs in neon green and (b) illustrates the underside of the drone with the black 3D printed cage to protect the NVIDIA Orin NX onboard computer and carrier board. (c) illustrates one hardware experiment where the platform follows a circle trajectory and (d) illustrates the same hardware experiment after the platform has inverted and executes another circle trajectory.

substituting the chart definitions in Eq. (11) yields

$$\boldsymbol{q}_{\text{yaw}}(\psi_S - \psi_N) = \frac{1}{\sqrt{1-c^2}}[-b, 0, 0, -a]^\top. \tag{19}$$

Comparing with yaw quaternion definition in Section 3.3, $\psi_S - \psi_N = 2\,\text{atan2}(a, b)$. Hence, continuity between charts is maintained by setting ψ_S equal to the nominal yaw reference ψ_N plus a saved offset $2\,\text{atan2}(a, b)$, provided that $\text{atan2}(a, b)$ remains unchanged in the South chart. If $\text{atan2}(a, b)$ does change, then ψ_S may gradually diverge from ψ_N. Updating $\text{atan2}(a, b)$ while in the South chart would improve ψ_S tracking of the desired ψ_N, but would become singular at inverted hover $[a, b, c] = [0, 0, -1]$, proving impractical and reflecting the absence of a globally nonsingular coordinate chart on $\mathbb{S}^2$.

B Hardware Deployment

B.1 Experimental Platform

The quadrotor is a custom-built 6" platform (shown in Fig. 7) created with the Lumenier QAV-R 2 Freestyle Quadcopter frame, TBS Lucid H7 flight controller, TBS Lucid 60A AM32 4-in-1 ESC, and an NVIDIA Orin NX (16 GB) with Seeedstudio A603 carrier board. The motors are Brotherhobby Avenger V2 2507-1850KV with T-Motor T6143 propellers powered using a 6S Tattu 1700 mAh

battery. The system ran Ubuntu 22.04 using Jetpack 6.2 with communications between the flight controller and carrier board running over UART. A cage was 3D printed to protect the Orin NX and carrier board using black TPU for AMS filament. The cage was printed with 4 wall loops and 50% infill. Neon green legs were also 3D printed from the same material (100% infill). The TPU for AMS was chosen for its ability to absorb an impact and the strength to withstand breaking. 6 reflective markers ($19\,\mathrm{mm}$) were mounted to the frame so that a minimum of four are viewable by the motion tracking system for any given orientation of the platform.

B.2 Implementation Details

The system ran Betaflight firmware. The middleware used was ROS2 Humble and Shield AI's EdgeOS [45]. All control algorithms ran onboard the Orin NX and DShot commands were sent over UART to the flight controller at a rate of $1\,\mathrm{kHz}$. Policy inference operated at a rate of $50\,\mathrm{Hz}$ using ONNX Runtime's TensorRT Execution Provider [46], position control at $100\,\mathrm{Hz}$, and orientation control at $1\,\mathrm{kHz}$.

B.3 System Identification

The steady-state thrust and moment scale models were fit via least squares from data collected using the Tyto Robotics Series 1585 Drone Thrust Stand and RC control board (to send DShot commands and measure the response). Results for steady-state model fitting are shown in Figs. 8a and 8b. The transient model parameters were identified from step-response experiments: rise times were extracted within each operating regime, and during transitions between them. The $(+)\rightarrow(-)$ and $(-)\rightarrow(+)$ transition points were identified manually from the motor rate response, with results shown in Figs. 8c to 8f with a corresponding simulation of our transient thrust model (Section 3.1) overlaying the experimental data for the same desired motor rates.

The mass of the quadrotor and its motors was measured directly on the hardware, while the location of the center of mass (CM) relative to the geometric center (GC), and the inertial matrix were estimated in a computer-aided design (CAD) program. Estimating the center of mass offset was crucial for hardware deployment. Following the approach of Mellinger [47], the center of mass in $\mathcal{W}$ can be defined relative to the geometric center by the relation $\boldsymbol{r}_{CM} = \boldsymbol{r}_{GC} + \boldsymbol{R}\boldsymbol{r}_{\text{off}}$, where $\boldsymbol{r}_{\text{off}}$ is a vector in $\mathcal{B}$ that defines the offset from the center of mass to the geometric center and $\boldsymbol{R}$ defines the rotation matrix from $\mathcal{B}$ to $\mathcal{W}$. Using this, the equations of motion are modified to [47]:

$$m\left(\ddot{\boldsymbol{r}}_{\text{GC}} + [\dot{\boldsymbol{\omega}}]_\times \boldsymbol{r}_{\text{off}} + [\boldsymbol{\omega}]_\times^2 \boldsymbol{r}_{\text{off}}\right) = -mg\hat{\mathbf{e}}_3 + f_c\hat{\mathbf{b}}_3, \tag{20}$$

$$\mathcal{I}_{CM}\dot{\boldsymbol{\omega}} = -[\boldsymbol{\omega}]_\times \mathcal{I}_{CM}\boldsymbol{\omega} + \boldsymbol{\tau} - \boldsymbol{r}_{\text{off}} \times [0, 0, f_c]^\top, \tag{21}$$

where we use $[\cdot]_\times$ to denote skew-symmetric matrices. Although the simulator does not account for CM offsets, the controller onboard the robot must reject its effect; thus, we modify the attitude control law to:

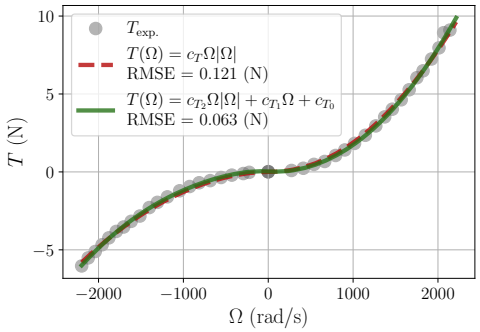
$$\boldsymbol{\tau} = -\boldsymbol{J}_R^{-\top}(\boldsymbol{e}_R)\boldsymbol{K}_{\boldsymbol{R}}\boldsymbol{e}_{\boldsymbol{R}} - \boldsymbol{K}_{\boldsymbol{\omega}}\boldsymbol{e}_{\boldsymbol{\omega}} + \boldsymbol{r}_{\text{off}} \times [0, 0, f_c]^\top. \tag{22}$$

Lastly, the policy inference and communication delay was characterized using a throttled ROS2 logger to print system time differences between observation concatenation and application to HFCA implementation.

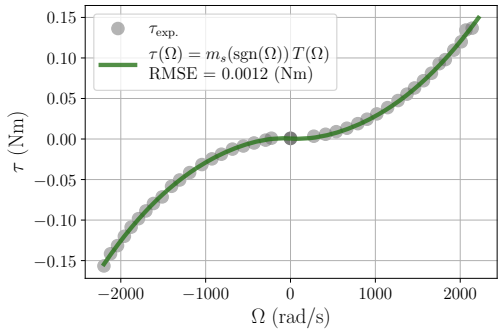
C Training & Simulation

We train all policies in our JAX-based simulator, `acrorl`, using an RTX 4090. For evaluation and deployment, we use the mean of the learned stochastic policy, and introduce an inversion flag μ to switch from trajectory tracking geometric control to inversion on the hardware platform. During training, environments are initialized from randomized orientations, angular rates, positions, and velocities to improve robustness.

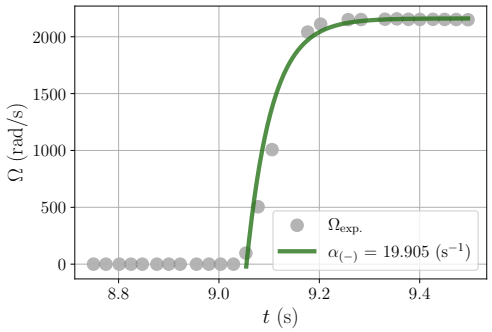
The measured inference and communication delay from the hardware deployment framework is modeled in simulation to aid the sim2real transfer. Training is completed within 9 minutes for



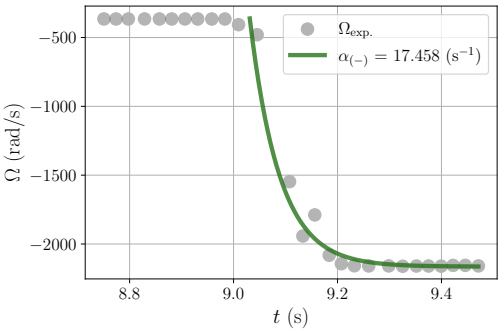
(a) Steady-state thrust model



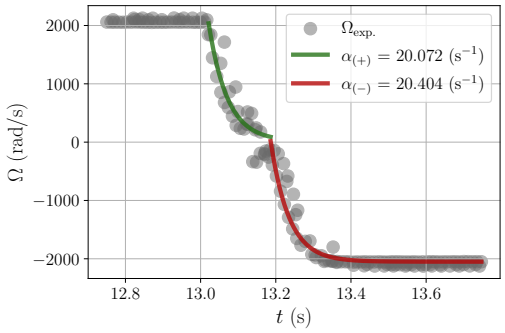
(b) Steady-state torque model



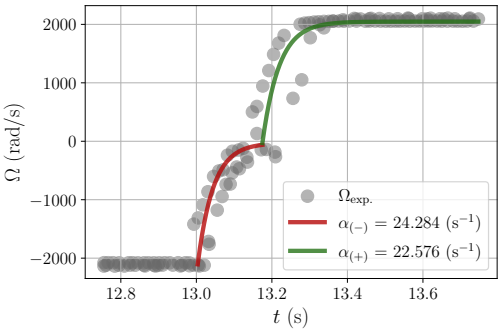
(c) $(+)$ transient model



(d) $(-)$ transient model



(e) $(+)\rightarrow(-)$ transient model



(f) $(-)\rightarrow(+)$ transient model

Figure 8: Thrust and torque model fit to experimental data collected using the Tyto Robotics Series 1585 drone thrust stand and RC control board. (a) illustrates how asymmetric propellers generate significantly more thrust in the $(+)$ operating regime (about $2\times$) compared to the $(-)$ operating regime. (b) illustrates the torque varying across operating regimes due to propeller efficiency asymmetry (i.e., the moment scale is larger in the (-) regime because it is less efficient). (c) illustrates the first-order model fit to the $(+)$ operating regime of the propeller. (d) illustrates the first-order model fit to the $(-)$ operating regime of the propeller. (e) illustrates the piecewise first-order model fit to the $(+)\rightarrow(-)$ operating regime transition, and the apparent stochasticity and asymmetry across regimes. (f) illustrates the piecewise first-order model fit to the $(-)\rightarrow(+)$ operating regime transition, and the apparent stochasticity and asymmetry across regimes.

both policies with 2048 parallel environments and 750 training epochs. Table 3 reports the model architectures, hyperparameters, and key environmental setup choices for both NTI and ITN training, while Table 4 reports the individual weights applied to Eq. (8). We also report the best performing minimum snap trajectory parameters in Table 5.

Parameter	**Description**	**Value**
Network Architecture		
Hidden layer dims	MLP layer widths	$[512, 512]$
Initial $\log \sigma$	Initial log std of action distribution	$\log(0.25)$
Action history	Number of past actions in observation	3
PPO Hyperparameters		
Learning rate	Adam optimizer learning rate	3×10^{-4}
γ	Discount factor	0.99
λ	GAE decay parameter	0.95
Clip ϵ	PPO clipping parameter	0.2
Entropy coefficient	Entropy regularization weight	0.01
Value function coef.	Value loss weight	0.5
Update epochs	Gradient epochs per rollout	4
Minibatches	Number of minibatches per update	20
Training Setup		
Num. environments	Parallel rollout environments	2048
Num. epochs	Total training epochs	750
Episode duration	Training episode length	3 s
dt_{env}	Simulation environment timestep	0.02 s
dt_{dyn}	Simulation dynamics timestep	0.001 s
Delay	Simulated policy inference delay	0.006 s

Table 3: PPO training hyperparameters and environment configuration.

Reward term	**ITN**	**NTI**
Position, $w_{\boldsymbol{r}}$	5.000	5.000
Orientation, $w_{\boldsymbol{g}_b}$	3.000	3.000
Thrust posture, w_η	0.100	0.100
Velocity, $w_{\dot{\boldsymbol{r}}}$	0.000	0.005
Angular velocity, $w_{\boldsymbol{\omega}}$	0.750	0.200
Action rate $w_{\dot{\boldsymbol{r}}_{\text{d}}}$	0.250	0.200

Table 4: Reward weights used for inverted-to-nominal (ITN) and nominal-to-inverted (NTI) training.

Parameter	**Value**
Second waypoint δz	0.45 m
η schedule	(1.0, -1.0)
Trajectory duration	2.0 s
Segment durations	(1.0, 1.0) s

Table 5: Tuned minimum snap trajectory parameters for best inversion performance.