This is the Pre-Published Version.
The following publication Y. Zhao, M. Lyu, C. Li and H. Huang, "Bidirectional Thrust Control for Quadrotor Safety," in IEEE Robotics and Automation Letters, vol. 11, no. 3, pp. 2650-2657, March 2026 is available at https://doi.org/10.1109/LRA.2026.3653327.

Bidirectional Thrust Control for Quadrotor Safety

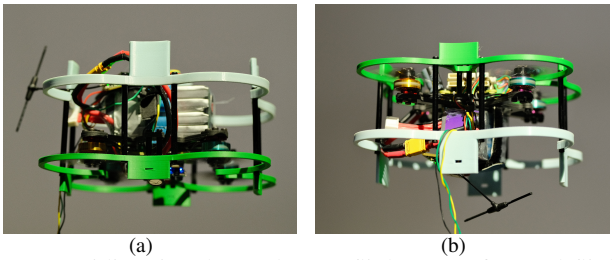
Yibo Zhao, Mingyang Lyu, Chengao Li and Hailong Huang

***Abstract*— Quadrotors performing aerial tasks are vulnerable to sudden external disturbances, which may lead to instability, control loss, or even structural damage such as broken arms or frame failure. These threats are particularly critical during flight, where recovery opportunities are limited. To address this, we propose a bidirectional thrust control framework that improves mid-air impact resilience. A lightweight recurrent neural network (RNN)-attention module detects and evaluates external forces in real time. When the disturbance is mild, a model predictive control (MPC) + active disturbance rejection control (ADRC) controller ensures stability; when it nears a critical level, the system switches to a flipping recovery policy that exploits bidirectional thrust to regain balance. Experiments validate robustness and safety under sudden external disturbances.**

I. INTRODUCTION

In recent years, the rapid development of both hardware and software technologies has significantly accelerated the deployment of quadrotors across a wide range of applications, including aerial surveillance [1], precision agriculture [2], infrastructure inspection [3], and urban delivery [4]. As these platforms become increasingly integrated into complex and unstructured environments, they are also exposed to a variety of external disturbances during flight. Among these, impulsive forces —whether due to environmental conditions [5], collisions [6], or adversarial interference [7]—pose a serious risk to flight stability. Even minor disturbances may render onboard tasks infeasible, while stronger impacts can result in complete loss of control or catastrophic crashes. These risks not only jeopardize the airframe itself but also endanger people and property on the ground. Such scenarios underscore the urgent need for responsive and adaptive control systems capable of handling abrupt, directional disturbances in real time.

To meet these challenges, artificial intelligence (AI) technologies have increasingly been adopted in the field of quadrotors. Deep learning-based perception modules, for instance, have improved obstacle detection [8], terrain following [9], and failure prediction [10]. More recently, the integration of AI into control pipelines has gained attention, offering the potential to enhance resilience through predictive and adaptive capabilities [11]. Specifically, RNN and attention mechanisms have shown excellent performance in modeling time-varying external inputs and estimating system states under partial observability or sudden perturbations [12]. Despite these advances, AI-assisted control frameworks remain underexplored in the context of emergency response and recovery—particularly for counteracting physical impulses from below that may induce quadrotor flipping.



(a) (b)

Fig. 1: Bidirectional quadrotor flight: (a) forward-flight mode; (b) reverse-flight mode with propeller rotation reversed.

Concurrently, progress in model-based control techniques—such as MPC [13] and ADRC [14]—has offered robust tools for handling uncertainties and achieving fast dynamic responses. Meanwhile, various other methods based on adaptive control [15] or observer-based control [16] have also been explored. These approaches, when carefully designed and combined, can not only stabilize the system under moderate disturbances [17] but also serve as building blocks for more agile and aggressive recovery maneuvers [18]. However, most existing frameworks assume unidirectional thrust capabilities, which limit the ability of the vehicle to respond effectively in extreme conditions such as sudden impacts or partial overturning. There remains a lack of unified strategies that leverage both intelligent perception and bidirectional actuation to ensure flight safety.

Existing research on bidirectional quadrotors remains limited. One notable contribution is the attitude representation method proposed by [19], which addresses the singularity issues inherent in traditional Euler angle formulations. In [20], the authors proposed a control allocation strategy for bidirectional thrust quadrotors subject to actuator constraints. Their work focuses on addressing the limitations of bidirectional actuators by formulating an optimal control allocation method that ensures feasibility and efficiency under constrained conditions. [21] also conducted related work, demonstrating agile flight trajectories enabled by bidirectional thrust capabilities. Building upon this, [22] formulated

This work was supported by the Research Centre for Low-Altitude Economy (RCLAE), Hong Kong Polytechnic University.

Yibo Zhao and Chengao Li are with Department of Aeronautical & Aviation Engineering (AAE), Hong Kong Polytechnic University, Hong Kong, China 23039432r@connect.polyu.hk, chen3li@polyu.edu.hk.

Mingyang Lyu is with Department of Industrial & System Engineering (ISE), Hong Kong Polytechnic University, Hong Kong, China robert-jeff.lyu@connect.polyu.hk.

Hailong Huang is with Department of Aeronautical & Aviation Engineering (AAE) and the Research Centre for Low-Altitude Economy (RCLAE), Hong Kong Polytechnic University, Hong Kong, China hailong.huang@polyu.edu.hk.

the control problem of bidirectional quadrotors as a standard model predictive control framework, providing a structured approach for trajectory optimization under bidirectional thrust dynamics. However, the above works are either limited to simulation or demonstrate relatively conservative flight behaviors, without fully exploiting the expanded feasible region enabled by bidirectional thrust.

In this paper, a RNN-Attention-aided Bidirectional Thrust Control (BTC) framework is designed to improve quadrotor resilience under sudden impulsive forces. The bidirectional thrust mechanism allows the quadrotor to generate upward or downward force as needed, enabling agile recovery from precarious flight states. Our contributions are summarized as follows:

1) We propose a novel Robust BTC framework, which can make full use of the forward and backwards thrust of the rotor in different scenarios to achieve the purpose of safe flight. The framework consists of a frontend that generates the desired state and a backend that solves the trajectory optimization and control in one step.
2) We develop and release a compact and extensible simulation platform in MATLAB Simulink tailored for bidirectional quadrotor dynamics, along with a lightweight C++ controller implementation that facilitates rapid testing and deployment[1]. The controller only requires four digital output signals to drive the motors directly, making it theoretically feasible to bypass full flight controllers like PX4. This simplifies system architecture and opens the possibility for deployment on low-cost quadrotor platforms.
3) We conduct, to the best of our knowledge, one of the few real-world flipping experiments on bidirectional thrust quadrotors. This demonstrates the feasibility of aggressive inversion maneuvers using minimal hardware modifications and fills a gap rarely addressed in existing literature.

II. METHODOLOGY

The overall quadrotor system is illustrated in Fig. 2. Based on the data provided by the motion capture system and controller feedback, which estimates the quadrotor's full state and external disturbance, our BTC framework directly generates actuator commands. Specifically, the BTC framework comprises two main components: the frontend and the backend. The frontend function is designed to detect the level of disturbance to determine whether a flip is necessary to maintain balance. When the disturbance is below the threshold, the quadrotor employs the MPC + ADRC scheme to maintain stability. However, when the disturbance exceeds the threshold, the quadrotor activates the flipping mechanism, utilizing the external disturbance as an assistive force to execute a 180° flip for stability maintenance. The backend runs in a loop at a constant frequency (100 Hz) and consists of MPC and ADRC, where the attitude controller incorporates a bidirectional thrust mechanism to address external disturbances that exceed the predetermined threshold.

[1]https://github.com/Y1bo-Zhao/Bidirectional

TABLE I: Overview of the main symbols

Symbol	Definition
$\mathbf{F}, \mathbf{M_x}, \mathbf{M_y}, \mathbf{M_z}$	Vehicle's thrust, moments
W, B	World inertial frame, quadrotor body frame
$\mathbf{r} = [x, y, z]$	Vehicle's position
ϕ, θ, ψ	Vehicle's attitude
m, I_1, I_2, I_3	Vehicle's mass, inertia
F_1, F_2, F_3, F_4	Each rotor's thrust
l, C	Arm length, force to moment scaling factor
g, K_i	Gravity acceleration, drag coefficients
γ	The set of flat outputs
$\mathbf{R}, \mathbf{R_d}$	Rotation matrix, desired rotation matrix
K_p, K_v, K_R, K_w	PD controller's coefficients
$\mathbf{b_3}, \mathbf{z_{bd}}, a, b, c$	Body z-axis, desired body z-axis, components of b_3
$q_{abc}, q_\psi, \overline{q}_{abc}, \overline{q}_\psi$	N mode's quaternion, S mode's quaternion

The notations adopted throughout this paper are summarized in Table I. In the proposed system model, the quadrotor state is denoted by $\mathbf{x}$, while the input vector is given by $\mathbf{u} = [\mathbf{F}, \mathbf{M_x}, \mathbf{M_y}, \mathbf{M_z}]^T$. For readability, we have removed the frame indexes because they are consistent throughout the description.

A. Bidirectional Quadrotor Dynamic

The quadrotor's state space is described from the inertial frame W to the body frame B, as $\mathbf{x} = [x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \dot{\phi}, \dot{\theta}, \dot{\psi}]^T$, where $x, y, z \in R$ are three positions, ϕ, θ, ψ are three Euler angles, which describe the rotation of the quadrotor. The input to the system is provided by the thrust F_1, F_2, F_3, F_4 generated by the four motors. We define u_i as the virtual control inputs as follows:

$$\begin{aligned} u_1 &= \frac{F_1 + F_2 + F_3 + F_4}{m}, \\ u_2 &= \frac{l(-F_1 - F_2 + F_3 + F_4)}{I_1}, \\ u_3 &= \frac{l(-F_1 + F_2 - F_3 - F_4)}{I_2}, \\ u_4 &= \frac{C(F_1 - F_2 + F_3 - F_4)}{I_3}, \end{aligned} \tag{1}$$

where m is the total mass of the quadrotor, l is the distance from the axis of rotation of the rotors to the center of the quadrotor, I_i is the moments of inertia with respect to the axes, and C is the force-to-moment scaling factor. A simplified dynamic model of the quadrotor is given below:

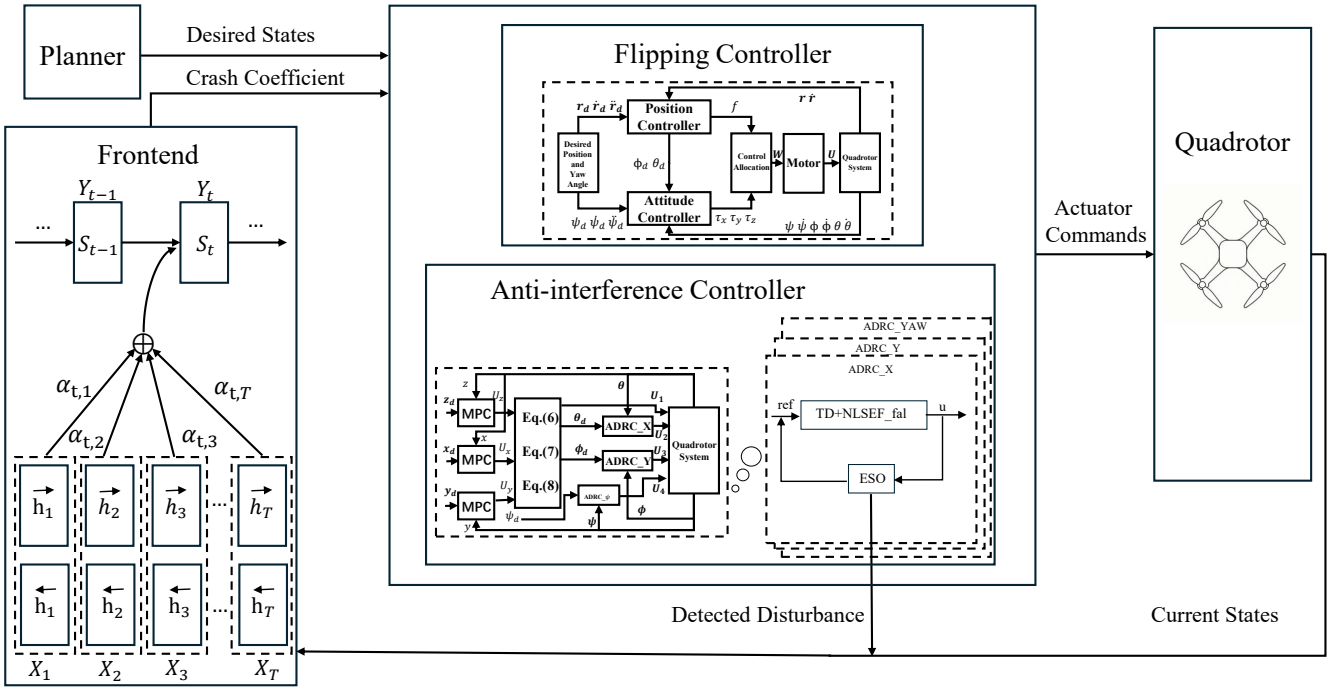


Fig. 2: Overall system architecture. The planner generates the desired position and yaw angle, which is beyond the scope of this paper. The quadrotor dynamics are detailed in Section II-A. The frontend outputs a risk assessment, which is discussed in Section II-B. The two control strategies are described in Sections II-C and II-D, respectively.

$$
\begin{aligned}
\ddot{x} &= u_1(\cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi) - \frac{K_1\dot{x}}{m},\\
\ddot{y} &= u_1(\sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi) - \frac{K_2\dot{y}}{m},\\
\ddot{z} &= u_1(\cos\theta\cos\phi) - g - \frac{K_3\dot{z}}{m},\\
\ddot{\theta} &= u_2 - \frac{lK_4\dot{\theta}}{I_1},\\
\ddot{\phi} &= u_3 - \frac{lK_5\dot{\phi}}{I_2},\\
\ddot{\psi} &= u_4 - \frac{K_6\dot{\psi}}{I_3},
\end{aligned}
\tag{2}
$$

where g is the acceleration of gravity, K_i is the drag coefficients. Due to the well-known differential flat properties of the quadrotor dynamics with four inputs. A common combination of controlled outputs can be x, y, z and ψ, which are utilized to track the desired positions, achieve an arbitrary heading, and stabilize the remaining two angles.

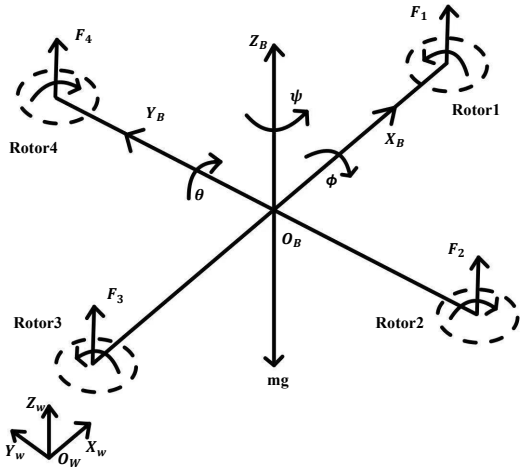


Fig. 3: Schematic of the quadrotor's free-body dynamics: body frame B relative to world frame W.

B. RNN-supported frontend

In quadrotor control, real-time roll-over risk prediction is essential for safety. Because the flight dynamics are highly nonlinear and history-dependent, simple thresholds on attitude or disturbance are unreliable, so we place a learning-based predictor at the controller frontend. It ingests a 1-s window of flight data—ESO disturbance estimates and vehicle states—and an attention-based RNN, implemented as a gated recurrent unit network, maps this sequence to a roll-over probability between 0 and 1. This risk score drives a hysteresis mode switch: at low risk, a disturbance-rejection controller is used; at high risk, a flip-recovery policy exploits strong disturbances to restore attitude. For clarity, the backend controller is conceptually decoupled into these two regimes.

C. Anti-interference Controller

To balance extensibility and robustness against external disturbances, the controller is designed in an MPC + ADRC structure [23]. The upper-layer MPC offers flexibility to incorporate scenario-specific constraints, while the lower-layer ADRC provides disturbance rejection and estimates external forces as feedback for the frontend. Drawing from the dynamic characteristics of quadrotor systems, the control architecture is structured into four decoupled subsystems: vertical motion along the z-axis, a cascaded x–θ loop, a cascaded y–ϕ loop, and yaw stabilization around ψ. The z-axis altitude and yaw angle ψ are directly regulated by the virtual control inputs U_1 and U_4, respectively. For lateral control, a hierarchical cascaded scheme is adopted: the x-axis is modulated via the pitch angle θ, while the y-axis position is controlled through adjustments in the roll angle ϕ. The overall structure of this multi-loop control strategy is shown in Fig. 4.

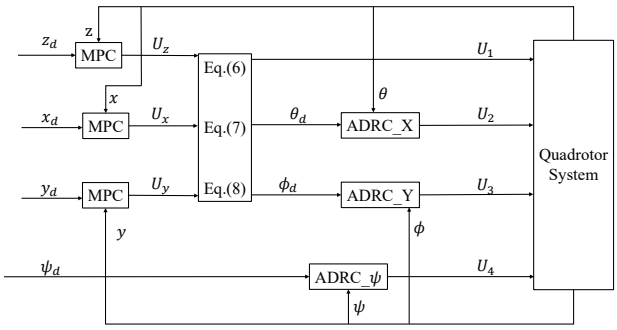


Fig. 4: Block diagram of the anti-interference controller. The upper-level MPC maps desired states and constraints to desired torques; the lower-level attitude loop uses a three-channel ADRC.

1) Controller design for altitude z channel: In the vertical motion control subsystem, the goal is to compute the collective thrust command U_1. First, the controller determines the intermediate signals U_x, U_y, and U_z, which represent the desired linear accelerations along the UAV's body–frame x, y, and z axes, respectively, required to track the reference trajectory. These accelerations are then transformed into U_1 according to Eq. (2), yielding

$$U_1 = m\sqrt{U_x^2 + U_y^2 + \left(U_z - g\right)^2}\,. \tag{3}$$

2) Controller design for $x-\theta$, $y-\phi$ and ψ cascade channel: Taking the x–θ channel as an example, the desired position x_d initiates the outer-loop control, from which the virtual control U_x is calculated. The corresponding pitch reference θ_d is then obtained and passed to the inner loop:

$$\theta_d = \arctan\left(\frac{U_x cos\psi + U_y \sin\psi}{U_z - g}\right). \tag{4}$$

The altitude channel is first processed using a tracking differentiator (TD), defined as:

$$\begin{aligned} \dot{v}_{\theta 1} &= v_{\theta 2}, \\ \dot{v}_{\theta 2} &= f_{han}(v_{\theta 1} - \theta_d, v_{\theta 2}, R, h), \end{aligned} \tag{5}$$

where $v_{\theta 1}$ corresponds to the smoothed tracking signal for the desired roll angle θ_d, and $v_{\theta 2}$ denotes its estimated derivative. The parameter R regulates the rate of convergence, while h controls the smoothing intensity. The nonlinear function $f_{han}(\cdot)$ represents the fast synthesis control law [24]. Then the ESO for this channel is designed as:

$$\begin{aligned} \dot{z}_{11} &= z_{12} - \beta_1(z_{11} - y_\theta), \\ \dot{z}_{12} &= z_{13} - \beta_2(z_{11} - y_\theta) + U_2, \\ \dot{z}_{13} &= -\beta_3(z_{11} - y_\theta). \end{aligned} \tag{6}$$

Finally, using the estimated states from the observer, the control law is given by:

$$\begin{aligned} U_2 = {} & \beta_4 fal(\theta_d - z_{11}, \alpha_1, \delta) \\ & + \beta_5 fal(v_{\theta 2} - z_{12}, \alpha_2, \delta) \\ & - z_{13}, 0 < \alpha_1 < 1 < \alpha_2, \\ fal(x, \alpha, \delta) = {} & \begin{cases} \frac{x}{\delta^{1-\alpha}}, & \text{if } |x| < \delta, \\ |x|^\alpha \cdot \operatorname{sign}(x), & \text{otherwise}, \end{cases} \end{aligned} \tag{7}$$

where $v_{\theta 1}$ tracks the reference θ_d, and $v_{\theta 2}$ is its derivative. The observer states z_{11}, z_{12}, and z_{13} estimate $v_{\theta 1}$, $v_{\theta 2}$, and the total disturbance in the x-θ channel, respectively, with y_θ as the measured output. With the observer in place, the control law is implemented via nonlinear state error feedback (NLSEF). The y–ϕ and ψ channel adopts the similar control structure as the x–θ loop.

D. Flipping Controller

Geometric control for quadrotors leverages the fact that thrust is generated along the body-fixed axis b_3, which imposes constraints on the structure of the rotation matrix R, specifically fixing its third column.

As described in [25], classical geometric control constructs an orthonormal frame in $\mathbb{R}^3$ by applying the Gram–Schmidt process to two linearly independent vectors, resulting in a right-handed frame that forms a valid rotation matrix in $SO(3)$. The vector a_1 defines the yaw direction. A singularity arises when $\ddot{r} + g$ aligns with a_1. To avoid this, a_1 is commonly chosen orthogonal to gravity, such as $a_1 = [\cos(\psi), \sin(\psi), 0]^T$. With this selection, a singular case occurs when the quadrotor undergoes a 90° pitch relative to its hovering orientation,

$$\begin{aligned} R(\gamma) &= \begin{bmatrix} b_1^W & b_2^W & b_3^W \end{bmatrix}, \\ b_3^W &= \frac{\ddot{r} + g}{\|\ddot{r} + g\|}, \\ b_2^W &= \frac{b_3^W \times a_1}{\|b_3^W \times a_1\|}, \\ b_1^W &= b_2^W \times b_3^W. \end{aligned} \tag{8}$$

The rotational control input, consisting of proportional and derivative components, is computed from the desired orientation following the approach in [26],

$$\begin{bmatrix} m_1 \\ m_2 \\ m_3 \end{bmatrix} = -k_R \left(\frac{1}{2} \left(R_d^T R - R^T R_d \right)^\vee \right) - k_\omega (\omega - \omega_d). \quad (9)$$

The force command is derived by applying feedback to the position and velocity errors of the quadrotor:

$$f = (-k_x(r - r_d) - k_v(\dot{r} - \dot{r}_d) + \dot{r}_d) \cdot R_{e3}. \quad (10)$$

It is mentioned that this parameterization encounters a singularity when $e_3 \times b_3 = 0$, making attitude construction problematic. While this issue is typically negligible during nominal flight, it becomes critical for maneuvers involving full inversion. To support such aggressive trajectories, a globally defined attitude representation is required. To overcome this limitation and enable a more robust formulation of $R(\gamma)$, we adopt unit quaternions as the attitude representation, following the method in [27]. To achieve global coverage of $SO(3)$, two parameterizations are required. The first is referred to as the N mode:

$$q_{abc} = \frac{1}{\sqrt{2(1+c)}} \begin{bmatrix} 1+c \\ -b \\ a \\ 0 \end{bmatrix}, \quad (11)$$

$$R(q_{abc})e_3 = [a, b, c]^T, \quad (12)$$

$$q_\psi = \left[\cos\frac{\psi}{2}, 0, 0, \sin\frac{\psi}{2} \right], \quad (13)$$

$$q = q_{abc} \otimes q_\psi, \quad (14)$$

where $[a, b, c]^T$ is the unit vector components of b_3^W. $R(q)$ represents the rotation matrix obtained from the quaternion q via the standard mapping to $SO(3)$.

To compute the control input, a transformation from flat outputs to state variables is required. By parameterizing the rotation matrix R with the quaternion q, it becomes necessary to utilize a standard quaternion property that relates the time derivative of q to the angular velocity expressed in the body frame:

$$\omega = 2q^{-1} \otimes \dot{q}, \quad (15)$$

$$\begin{aligned} \omega_1 &= \sin(\psi)\dot{a} - \cos(\psi)\dot{b} - (a\sin(\psi) + b\cos(\psi)) \left(\frac{-\dot{c}}{c+1} \right), \\ \omega_2 &= \cos(\psi)\dot{a} + \sin(\psi)\dot{b} - (a\cos(\psi) - b\sin(\psi)) \left(\frac{-\dot{c}}{c+1} \right), \\ \omega_3 &= \frac{b\dot{a} - a\dot{b}}{1+c} + \dot{\psi}. \end{aligned} \quad (16)$$

Since the time derivatives of a, b, and c are needed, the normalization constraint is differentiated accordingly. This results in full expressions for both angular velocity and orientation in terms of the flat outputs,

$$\begin{bmatrix} \dot{a} & \dot{b} & \dot{c} \end{bmatrix}^T = \frac{\xi^T \xi \cdot I - \xi \xi^T}{\|\xi\|^3} \cdot \dot{\xi}, \quad (17)$$

$$\xi = f \cdot b_3 = \ddot{r} + g. \quad (18)$$

A failure case of N mode arises during maneuvers involving full inversion, such as flips, which is singular at $c = -1$. To handle arbitrary orientations, including inverted configurations, an additional attitude mapping is required. This is accomplished by introducing a second quaternion parameterization that becomes singular at $c = 1$, in contrast to the original map.

$$\overline{q}_{abc} = \frac{1}{\sqrt{2(1-c)}} \begin{bmatrix} -b \\ 1-c \\ 0 \\ a \end{bmatrix}, \quad (19)$$

$$q_{\overline{\psi}} = \left[\cos\frac{\overline{\psi}}{2}, 0, 0, \sin\frac{\overline{\psi}}{2} \right], \quad (20)$$

$$\overline{\psi} = 2atan2(a, b) + \psi, \quad (21)$$

$$\overline{q} = \overline{q}_{abc} \otimes q_{\overline{\psi}}, \quad (22)$$

$$\begin{aligned} \omega_1 &= \sin(\overline{\psi})\dot{a} + \cos(\overline{\psi})\dot{b} - (a\sin(\overline{\psi}) + b\cos(\overline{\psi})) \left(\frac{\dot{c}}{c-1} \right), \\ \omega_2 &= \cos(\overline{\psi})\dot{a} - \sin(\overline{\psi})\dot{b} - (a\cos(\overline{\psi}) - b\sin(\overline{\psi})) \left(\frac{\dot{c}}{c-1} \right), \\ \omega_3 &= \frac{b\dot{a} - a\dot{b}}{1+c} + \dot{\overline{\psi}}. \end{aligned} \quad (23)$$

As a result, the system operates across these two modes. With the above formulation, the control inputs can now be defined. We begin by specifying the position and velocity errors as:

$$e_p = r - r_d, e_v = \dot{r} - \dot{r}_d. \quad (24)$$

Then, the desired control input **u** is computed as follows:

$$u_1 = (-K_p e_p - K_v e_v + mgz_W + m\ddot{r}_d) \cdot z_B, \quad (25)$$

$$\begin{aligned} [u_2, u_3, u_4]^T &= -K_R e_R - K_w e_w, \\ e_R &= \frac{1}{2}(R_{des}^T R_B^W - R_B^{WT} R_{des})^\vee, \\ e_w &= w^B - w_{des}^B. \end{aligned} \quad (26)$$

The detailed implementation is shown in Algorithm 1.

Algorithm 1: Flipping Controller

Require: Current state, desired state, model parameters, gains $(K_p, K_v, K_R, K_\omega)$, mode flag η
Ensure: Control inputs $[u_1, u_2, u_3, u_4]$

Compute position and velocity errors: $\mathbf{e}_p = \mathbf{r} - \mathbf{r}_d$, $\mathbf{e}_v = \dot{\mathbf{r}} - \dot{\mathbf{r}}_d$
Compute desired acceleration: $\mathbf{a}_{des} = -K_p\mathbf{e}_p - K_v\mathbf{e}_v + g\mathbf{e}_3 + \ddot{\mathbf{r}}_d$
Compute desired thrust: $\mathbf{F}_d = m\mathbf{a}_{des}$
Get current orientation R and body z-axis: $\mathbf{z}_b = R(:, 3)$
if $\eta = +1$ **then**
    $\mathbf{z}_{bd} \leftarrow \mathbf{a}_{des}/\|\mathbf{a}_{des}\|$
    $u_1 \leftarrow \mathbf{F}_d^\top \mathbf{z}_b$
else if $\eta = -1$ **then**
    $\mathbf{z}_{bd} \leftarrow -\mathbf{a}_{des}/\|\mathbf{a}_{des}\|$
    $u_1 \leftarrow -\mathbf{F}_d^\top \mathbf{z}_b$
end if
Compute actual acceleration and jerk error
Convert $\mathbf{z}_{bd}$ to (a, b, c) and compute derivatives
if $M = \text{N}$ **and** $c \le -0.6$ **and** $c_{\text{prev}} > -0.6$ **then**
    $M \leftarrow \text{S}$ ▷ Switch to S mode when crossing below -0.6
else if $M = \text{S}$ **and** $c \ge 0.6$ **and** $c_{\text{prev}} < 0.6$ **then**
    $M \leftarrow \text{N}$ ▷ Return to N mode when crossing above 0.6
end if
if $M = \text{N}$ **then**
    $q \leftarrow q_{abc}(a, b, c) \otimes q_\psi$
else
    $q \leftarrow \overline{q}_{abc}(a, b, c) \otimes q_{\overline{\psi}}$
end if
Compute desired rotation $R_d = R(q)$
Compute attitude error $\mathbf{e}_R$ and angular velocity error $\mathbf{e}_\omega$
Compute moments: $[u_2, u_3, u_4] = -K_R\mathbf{e}_R - K_\omega\mathbf{e}_\omega$

III. EXPERIMENT

A. Frontend Analysis and Training

Compared with the CNN–LSTM–attention model in [11]—which stacks a 1-D CNN+pooling front-end (compressing 53 flight-state channels and mining spatial correlations) before an LSTM and a temporal-attention layer—our predictor uses a lightweight recurrent encoder with a single temporal-attention head and no CNN. This choice reflects our setting: a 1-s (20-step) sequence of only nine physically meaningful variables $(x, y, z, \phi, \theta, \psi, d_x, d_y, d_z)$, where dynamic already captures inter-relations and convolutional local-feature extraction is unnecessary. In terms of training strategy, [11] selects the top-5 pitch-related features (via Pearson correlation) from 53 and applies min–max normalization under an 80/20 split, whereas we feed nine disturbance-aware channels (state + ESO disturbances) standardized by train-set mean/standard deviation because exogenous perturbations are pivotal to risk formation but are not explicitly modeled in [11]. Fig. 5 shows a head-to-head comparison with [11] under identical settings.

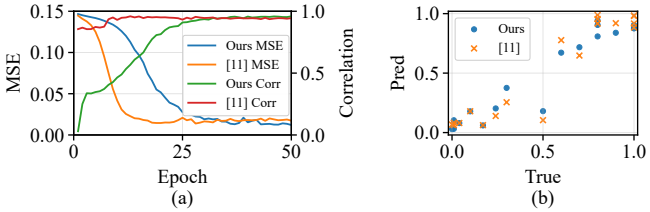


Fig. 5: Comparison between our frontend and Ref. [11]. (a) Validation MSE (left y-axis) and correlation (right y-axis) versus training epochs. (b) Predicted vs. ground-truth values on the held-out test set.

To generate a dataset and validate our algorithm, we built a MATLAB/Simulink simulation environment comprising a high-fidelity 6-DOF bidirectional quadrotor, a planner, and a complete ADRC-based controller. During runs, exogenous forces of varying magnitudes and application points on the airframe were injected to stress the system. In each trial, the ADRC's ESO estimated external disturbances while the plant provided states. These streams were sampled as 100 time steps at 10 ms (a 1 s window), forming the temporal input to the neural network and capturing short-horizon dynamics indicative of instability.

For labeling, we assigned a risk score in $[0, 1]$ based on whether the vehicle maintained flight or approached failure. In total, 100 labeled sequences generated via Monte Carlo simulations were collected. Correlating the risk score with the flipping controller's trigger across sweeps of force magnitude and location, we observed loss of control near a risk score of 0.81; to conservatively avoid structural damage (e.g., arm fracture), we set the switching threshold at a risk score of 0.8 for our airframe. The RNN-attention network was implemented in Python 3.12 using PyTorch 2.7.0 and trained with a cross-entropy loss and the Adam optimizer. All input features were normalized prior to training. This model serves as the decision-making front end for adaptive control switching in subsequent experiments.

B. Simulation Validation

To validate the proposed approach, a quadrotor model and its corresponding controller were first implemented in Simulink, following the design outlined earlier. The drone parameters were based on hardware routinely used in our laboratory, as summarized in TABLE II. The controller ran at 100 Hz, and experiments were conducted under two different external force conditions.

1) Experiment 1 Anti-disturbance: In the first experiment, the quadrotor is commanded to hover via continuous inputs from the planner. At $t = 5s$, a 10N impulsive force is applied to the center of one arm. Fig. 7(a) shows the angle between the body-frame z-axis and the reference vector $(0, 0, 1)$, indicating only minor deviation and demonstrating the ADRC

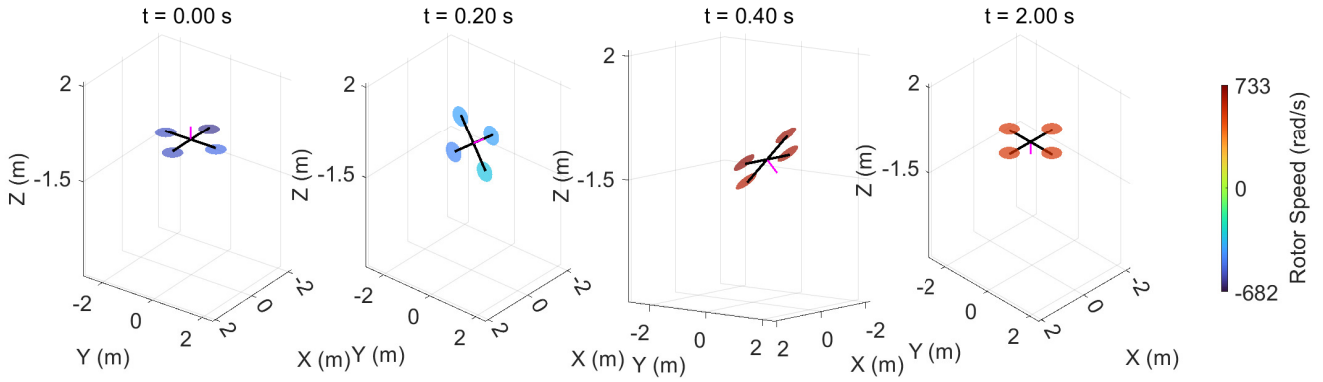


Fig. 6: Here is demonstrated the entire process of the bidirectional flipping flight in the simulation. The color of each rotor indicates its rotational speed, while the pink axis represents the body-fixed z-axis, serving as a visual cue for orientation.

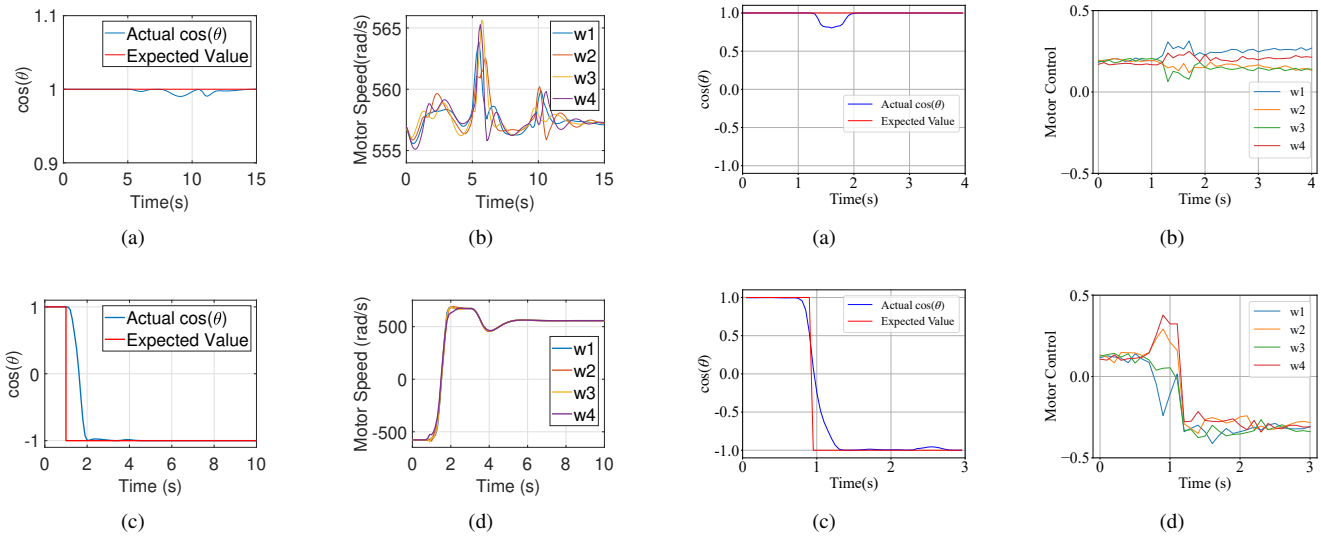


Fig. 7: Simulation result. During hovering, the four rotors spin at approximately 550 rad/s. As shown in (a), the applied external force is insufficient to trigger the flipping mechanism, and the controller maintains a nominal upright hovering attitude. In (b), the rotor speeds fluctuate slightly around 550 rad/s before stabilizing. In contrast, (c) and (d) show the response when a flip is triggered: $\cos(\theta)$ rapidly drops to -1, indicating a full inversion, and the rotor speeds reverse from approximately -550 rad/s to 550 rad/s.

Fig. 8: Real-world result. During hovering, the quadrotor remains stable before the external disturbance is applied. As shown in (a) and (b), when a disturbance occurs at $t = 1.2$ s, the controller restores the attitude with a maximum deviation of $41°$, demonstrating its disturbance rejection capability. In another case, as illustrated in (c), a stronger impact at $t = 0.65$ s causes the frontend to predict a crash and trigger a flip, forcing $\cos(\theta)$ = -1. Correspondingly, (d) indicates that the motor commands reverse sign and become marginally larger in magnitude, consistent with the propellers' non-symmetric thrust–torque characteristics under reverse rotation.

controller's effectiveness in rejecting the disturbance. Fig. 7(b) presents the rotational speeds of the four motors during the maneuver.

2) Experiment 2 Reverse: Under the same setup, the external force was increased to 50N, which triggered the quadrotor's flipping mechanism at $t = 0s$. Fig. 7(c) shows the desired and actual angles between the body-frame z-axis and the vector $(0, 0, 1)$, confirming that a full inversion was achieved. Fig. 7(d) illustrates the motor speeds, where the reversal of thrust direction required for the flip is clearly observable. The whole process is shown in Fig. 6.

C. Real-world Validation

To verify the transferability of our control strategy from simulation to real-world, we conducted physical experiments using the similar disturbance scenarios as in Experiments 1 and 2. In the anti-disturbance test, a moderate lateral impulse was delivered to one arm with a steel rod while hovering; the system successfully rejected the disturbance, as shown in Fig. 8(a) and Fig. 8(b), which respectively show the attitude deviation and motor responses. In the inversion test,

TABLE II: Quadrotor Model Parameters

Parameter	Value	Unit
Distance from body center to each motor	0.225	m
Quadrotor mass	1.4	kg
Moment of inertia about the x-axis	0.0211	$kg{\cdot}m^2$
Moment of inertia about the y-axis	0.0219	$kg{\cdot}m^2$
Moment of inertia about the z-axis	0.0366	$kg{\cdot}m^2$
Total rotational inertia of the motor and propeller about the rotation axis	0.0001287	$kg{\cdot}m^2$
Propeller thrust coefficient	3.5611	–
Propeller moment coefficient	1.1366	–
Propeller diameter	0.1	m
Damping coefficients $[K_1..K_6]$	[0.62 0.65 0.7 0.048 0.05 0.045]	–

a stronger impulsive disturbance was applied by hand to the arm, triggering the flipping controller. The resulting roll-over maneuver is depicted in Fig. 8(c) and Fig. 8(d), confirming that the bidirectional-thrust mechanism and control logic remained effective under real-world conditions.

IV. CONCLUSIONS

In this paper, we introduced an AI-aided BTC framework that enhances quadrotor resilience to sudden, potentially destabilizing disturbances. A lightweight RNN-attention frontend predicts roll-over risk online, while a dual-mode MPC+ADRC backend selects between disturbance-rejection and flip-recovery accordingly. Unlike conventional designs, BTC deliberately exploits bidirectional thrust to recover from near-overturn impacts. High-fidelity simulations and real flight tests confirm superior robustness and control adaptability.

Future work will focus on designing a unified optimization framework that simultaneously performs risk prediction, control mode selection, and trajectory generation, enabling more responsive and intelligent bidirectional control. Additionally, extending the framework to accommodate more complex aerial tasks and multi-vehicle coordination remains a promising direction.

REFERENCES

[1] Payam Shafiei Gohari, Hossein Mohammadi, and Sajjad Taghvaei. Using chaotic maps for 3D boundary surveillance by quadrotor robot. *Applied Soft Computing*, 76:68–77, 2019.

[2] Taha Elmokadem. Distributed coverage control of quadrotor multi-UAV systems for precision agriculture. *IFAC-PapersOnLine*, 52(30):251–256, 2019.

[3] Alex Junho Lee, Wonho Song, Byeongho Yu, Duckyu Choi, Christian Tirtawardhana, and Hyun Myung. Survey of robotics technologies for civil infrastructure inspection. *Journal of Infrastructure Intelligence and Resilience*, 2(1):100018, 2023.

[4] Gino Brunner, Bence Szebedy, Simon Tanner, and Roger Wattenhofer. The urban last mile problem: Autonomous drone delivery to your balcony. In *2019 International Conference on Unmanned Aircraft Aystems (ICUAS)*.

[5] Yuwei Wu, Ziming Ding, Chao Xu, and Fei Gao. External forces resilient safe motion planning for quadrotor. *IEEE Robotics and Automation Letters*, 6(4):8506–8513, 2021.

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

[7] Jianchuan Ye, Jiang Wang, Tao Song, Zeliang Wu, and Pan Tang. Nonlinear modeling the quadcopter considering the aerodynamic interaction. *IEEE Access*, 9:134716–134732, 2021.

[8] Xi Dai, Yuxin Mao, Tianpeng Huang, Na Qin, Deqing Huang, and Yanan Li. Automatic obstacle avoidance of quadrotor UAV via cnn-based learning. *Neurocomputing*, 402:346–358, 2020.

[9] Amirreza Kosari, Hossein Maghsoudi, Abolfazl Lavaei, and Rohollah Ahmadi. Optimal online trajectory generation for a flying robot for terrain following purposes using neural network. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 229(6):1124–1141, 2015.

[10] Xiaoxu Liu, Zike Yuan, Zhiwei Gao, and Wenwei Zhang. Reinforcement learning-based fault-tolerant control for quadrotor UAVs under actuator fault. *IEEE Transactions on Industrial Informatics*, 2024.

[11] Xuanlu Chen, Shenghan Zhou, Wenbing Chang, Fajie Wei, and Linchao Yang. Research on flight attitude prediction method for multi-rotor UAV based on cnn-lstm-attention model. In *2024 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pages 1139–1143. IEEE, 2024.

[12] Hang Zhou, Yibo Zhao, Xiaogang Xiong, Yunjiang Lou, and Shyam Kamal. IMU dead-reckoning localization with RNN-IEKF algorithm. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11382–11387. IEEE, 2022.

[13] Guillem Torrente, Elia Kaufmann, Philipp Föhn, and Davide Scaramuzza. Data-driven MPC for quadrotors. *IEEE Robotics and Automation Letters*, 6(2):3769–3776, 2021.

[14] Mauricio Alejandro Lotufo, Luigi Colangelo, Carlos Perez-Montenegro, Enrico Canuto, and Carlo Novara. UAV quadrotor attitude control: An ADRC-EMC combined approach. *Control Engineering Practice*, 84:13–22, 2019.

[15] Yuhao Zhou, Biao Luo, Xiaodong Xu, and Chunhua Yang. Adaptive robust attitude control for multiple quadrotor systems via integral reinforcement learning. *IEEE Transactions on Aerospace and Electronic Systems*, 2025.

[16] Kang Liu, Wenyu Yang, Lin Jiao, Zhipeng Yuan, and Chih-Yung Wen. Fast fixed-time distributed neural formation control-based disturbance observer for multiple quadrotor uavs under unknown disturbances. *IEEE Transactions on Aerospace and Electronic Systems*, 2025.

[17] Taki Eddine Lechekhab, Stojadin Manojlovic, Momir Stankovic, Rafal Madonski, and Slobodan Simic. Robust error-based active disturbance rejection control of a quadrotor. *Aircraft Engineering and Aerospace Technology*, 93(1):89–104, 2021.

[18] Fang Nan, Sihao Sun, Philipp Foehn, and Davide Scaramuzza. Nonlinear MPC for quadrotor fault-tolerant control. *IEEE Robotics and Automation Letters*, 7(2):5047–5054, 2022.

[19] Michael Watterson, Ahmed Zahra, and Vijay Kumar. Geometric control and trajectory optimization for bidirectional thrust quadrotors. In *International Symposium on Experimental Robotics*, pages 165–176. Springer, 2018.

[20] Walter Jothiraj, Inna Sharf, and Meyer Nahon. Control allocation of bidirectional thrust quadrotor subject to actuator constraints. In *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 932–938. IEEE, 2020.

[21] Moritz 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. IEEE, 2018.

[22] Jad Wehbeh and Inna 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.

[23] Huiying Liu, Yongming Yao, Jie Wang, Yutong Qin, and Tianyu Li. A control architecture to coordinate energy management with trajectory tracking control for fuel cell/battery hybrid unmanned aerial vehicles. *International Journal of Hydrogen Energy*, 47(34):15236–15253, 2022.

[24] Jingqing Han. From pid to active disturbance rejection control. *IEEE transactions on Industrial Electronics*, 56(3):900–906, 2009.

[25] Jur Van Den Berg, Dave Ferguson, and James Kuffner. Anytime path planning and replanning in dynamic environments. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 2366–2371. IEEE, 2006.

[26] Daniel Mellinger and Vijay Kumar. Minimum snap trajectory generation and control for quadrotors. In *2011 IEEE International Conference on Robotics and Automation*, pages 2520–2525. IEEE, 2011.

[27] Michael Watterson and Vijay Kumar. Control of quadrotors using the hopf fibration on so (3). In *Robotics Research: The 18th International Symposium ISRR*, pages 199–215. Springer, 2019.