Nonlinear MPC for Quadrotor Fault-Tolerant Control

Fang Nan, Sihao Sun, Philipp Foehn, Davide Scaramuzza

***Abstract*—The mechanical simplicity, hover capabilities, and high agility of quadrotors lead to a fast adaption in the industry for inspection, exploration, and urban aerial mobility. On the other hand, the unstable and underactuated dynamics of quadrotors render them highly susceptible to system faults, especially rotor failures. In this work, we propose a fault-tolerant controller using nonlinear model predictive control (NMPC) to stabilize and control a quadrotor subjected to the complete failure of a single rotor. Differently from existing works, which either rely on linear assumptions or resort to cascaded structures neglecting input constraints in the outer-loop, our method leverages full nonlinear dynamics of the damaged quadrotor and considers the thrust constraint of each rotor. Hence, this method could effectively perform upset recovery from extreme initial conditions. Extensive simulations and real-world experiments are conducted for validation, which demonstrates that the proposed NMPC method can effectively recover the damaged quadrotor even if the failure occurs during aggressive maneuvers, such as flipping and tracking agile trajectories.**

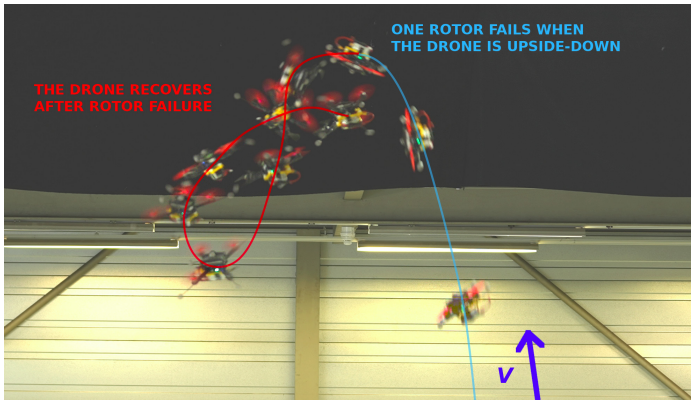


Fig. 1: Our quadrotor recovering from a rotor failure occurred during a flipping maneuver. **One rotor completely failed** when the quadrotor was on the apex and upside-down, where our controller recovered the damaged quadrotor despite the rotor failure.

VIDEO

Video of the experiments: https://youtu.be/Cn_836XGEnU

I. INTRODUCTION

A. Motivation

Multirotor aerial vehicles are rapidly reshaping the industry of logistics, inspection, agriculture, and even flying cars [1], [2]. Until 2021, there have been more than 300,000 commercial drones registered in the United States alone, according to the Federal Aviation Administration [3], and the global drone market is projected to 63.6 billion USD by 2025 [4]. However, flying accidents owing to system malfunctions, such as rotor failures, are major stumbling blocks to the development and the public acceptance of the drone industry. For instance, two crashes in 2019 caused the suspension of the autonomous drone delivery service by the Swiss Post for a year in Switzerland, after more than 3,000 successful autonomous flights [5].

Fault-tolerant flight control (FTFC) is a promising technique to improve flying safety, which only requires algorithmic adaptations rather than mechanical changes (e.g., parachutes). It is particularly valuable for quadrotors which are most susceptible to rotor failures, but are also the most popular type of all multirotors, due to their structural simplicity and efficiency.

Manuscript received: September 09, 2021; Revised December, 19, 2021; accepted January 23, 2022.

The authors are with the Robotics and Perception Group, Department of Informatics, University of Zurich, and Department of Neuroinformatics, University of Zurich and ETH Zurich, Switzerland (http://rpg.ifi.uzh.ch). This work was supported by the National Centre of Competence in Research (NCCR) Robotics, through the Swiss National Science Foundation (SNSF), and the European Union's Horizon 2020 Research and Innovation Programme under grant agreement No. 871479 (AERIAL-CORE) and the European Research Council (ERC) under grant agreement No. 864042 (AGILEFLIGHT).
Digital Object Identifier (DOI): see top of this page.

B. Related Work

FTFC is a long-standing challenge within the robotics and aerospace industries. Initial research in this field focused on partial effectiveness loss of rotors, such as blade damage, where a quadrotor is mainly regarded as a research platform for testing novel robust/adaptive control methods. These methods include but are not limited to sliding mode control [6], [7], $\mathcal{L}_1$ adaptive control [8], and active disturbance rejection control [9]. Extensive research on fault detection and diagnosis is also carried out and tested on quadrotor platforms (e.g., see [10], [11]).

While partial motor failure can be effectively handled by these adaptive methods, failure of a complete rotor is a more challenging problem and has a higher practical value. This is because switching off a severely damaged rotor in practice is more favorable as it prevents vibrations caused by rotor imbalance. A seminal work to address this problem is proposed by [12], which reveals that relinquishing the stability in yaw direction is inevitable after the failure of a motor. But even then, the altitude and position of a post-failure quadrotor remain controllable. Based on this seminal work [12] a wide variety of methods have been proposed and tested, including linear controllers such as proportional-integral-derivative (PID) controlller [13], linear quadratic regulator (LQR) [14], and linear parameter varying (LPV) [15] methods that rely on a

relaxed hovering equilibrium to perform linearization [16]. The Sequential Linear Quadratic (SLQ) control [17], on the other hand, performs the linearization around a predicted trajectory instead of a single equilibrium. Nonlinear methods that do not require the linearization have also been proposed, including robust feedback linearization [18], nonlinear dynamic inversion (NDI) [19], backstepping [20], and incremental nonlinear dynamic inversion (INDI) [21], [22]. Recently, vision-based approaches proposed by [19] have also jointly addressed the control and state estimation problem using only onboard sensors.

Most of these methods, however, rely on different assumptions for controller design, such as small-angle assumptions for model linearization or the principle of time-scale separations for separately designing position and attitude controllers. Moreover, all the aforementioned controllers do not consider actuator limits, which are crucial for flight performance, especially after rotor failures. For this reason, the nonlinear model predictive controller (NMPC), which exploits the full dynamics of a quadrotor, is a perfect candidate to tackle the fault-tolerant control problem.

NMPC is a control method that uses a nonlinear model to predict the system dynamics from a current time over a receding horizon and optimizes the behavior of the system based on certain performance measurements. It was shown in previous research that NMPC on quadrotors has a great advantage in agile trajectory tracking and dealing with constraints [23], [24], [25]. Due to the ability of the NMPC to fully exploit the input space without constraint violation, it was tested as the FTFC to stabilize a hexacopter with a rotor failure in [25], although the loss of a single rotor is not a challenging situation for hexacopter control. A more difficult case was tested, in which three rotors failed on a hexacopter [26]. It was shown with simulations that the nonlinear NMPC controls the horizontal position and maintains the altitude of the drone. The authors of [27] also proposed a method using MPC as the inner loop controller for a quadrotor subject to rotor failures. However, in [26] and [27], the method is only validated in the simulation, and only tested for FTFC starting from near-hovering conditions.

NMPC has several disadvantages that make it difficult to be implemented in the real world [28], [29]. To obtain good predictions it is necessary to accurately model the system dynamics, which is particularly challenging for aerial robots whose dynamics are severely influenced by aerodynamic effects. The complexity of the optimization problem also makes NMPC difficult to satisfy the requirements for real-time control. As a result, to the best of our knowledge, NMPC has never been implemented in real-world experiments to control a quadrotor in the case of a complete failure of a rotor.

C. Contribution

We present a nonlinear model predictive control (NMPC) framework for quadrotors that is tolerant to the complete failure of a single rotor. Using a real-time iteration scheme, the NMPC can be solved onboard at an adequate speed that satisfies the requirements for control. The proposed controller considers the full nonlinear dynamics of quadrotors, without resorting to a cascaded structure or any assumptions for linearization. The NMPC uses single rotor thrusts as inputs and considers their correct limits, which allows exploiting the full potential of any intact and damaged quadrotor without violating the real dynamics. An INDI low-level controller is applied to regulate the original NMPC thrust command to compensate for model uncertainties and disturbances on the rotational dynamics.

Experimental validations are performed in both simulation and real-world environments. We demonstrate the recovery performance of the proposed controller from several extreme conditions, such as flipping maneuvers and agile flights. Experiments also validate the ability of our controller in tracking agile trajectories despite the complete failure of a single rotor. Such recovery performance can significantly expand the flight envelope of damaged quadrotors, enabling controlled descent and landing maneuvers and ultimately allowing even safe flight under rotor failures.

II. Methodology

This section will introduce the control algorithm, and the model used for controller design. We use bold lowercase letters for vectors (e.g., $\boldsymbol{v}$) and bold uppercase letters for matrices (e.g., $\boldsymbol{M}$); otherwise they are scalars. Two right-handed coordinate systems are used to represent the body frame $\{\boldsymbol{x}_B,\ \boldsymbol{y}_B,\ \boldsymbol{z}_B\}$ and the inertial frame $\{\boldsymbol{x}_I,\ \boldsymbol{y}_I,\ \boldsymbol{z}_I\}$. The body frame is defined as is shown in Fig. 2. The inertial frame is defined with $\boldsymbol{z}_I$ pointing upwards opposite to gravity.

A. Quadrotor Modeling

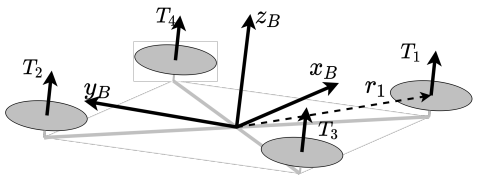


Fig. 2: Definition of quadrotor body frame and rotor indices.

To model the dynamics of a quadrotor, we define the quadrotor body frame as shown in Fig. 2. We use T_1 to T_4 to represent the thrusts generated by the four rotors, with the indices shown in Fig. 2, where $\boldsymbol{r}_1$ to $\boldsymbol{r}_4$ denote the position vectors of these rotors. Using quaternion representation of the quadrotor orientation, we could write the dynamics of the quadrotor as

$$\begin{aligned} \dot{\boldsymbol{p}} &= \boldsymbol{v}, & \dot{\boldsymbol{q}} &= \frac{1}{2}\boldsymbol{q} \circ \begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix}, \\ \dot{\boldsymbol{v}} &= \boldsymbol{q} \odot \begin{bmatrix} 0 \\ 0 \\ T \end{bmatrix} - \boldsymbol{g}, & \dot{\boldsymbol{\omega}} &= \boldsymbol{I}_v^{-1} \left(\boldsymbol{\tau} - \boldsymbol{\omega} \times (\boldsymbol{I}_v \boldsymbol{\omega}) + \boldsymbol{\tau}_{\text{ext}} \right), \end{aligned} \tag{1}$$

where $\boldsymbol{p}$ and $\boldsymbol{v}$ are the position and velocity of the center of gravity expressed in the inertial frame. $\boldsymbol{q} = \begin{bmatrix} q_w & q_x & q_y & q_z \end{bmatrix}^T$ is the quaternion representation of the quadrotor orientation, and $\boldsymbol{\omega} = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^T$ is the angular

velocity of the quadrotor expressed in the body frame. $\boldsymbol{I}_v$ is the inertia matrix of the entire quadrotor. The $\circ$ symbol represents quaternion multiplication, while $\odot$ represents the rotation of a vector using a quaternion. The unmodeled external torques due to aerodynamic disturbances and model mismatch are denoted as $\boldsymbol{\tau}_{\text{ext}}$. T and $\boldsymbol{\tau}$ are collective thrust and torque generated by the actuators, expressed by

$$\begin{bmatrix} T \\ \boldsymbol{\tau} \end{bmatrix} = \boldsymbol{G}\boldsymbol{t}, \tag{2}$$

where $\boldsymbol{t} = \begin{bmatrix} T_1 & T_2 & T_3 & T_4 \end{bmatrix}^T$ is the thrust vector, and $\boldsymbol{G}$ is the control effectiveness matrix:

$$\boldsymbol{G} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ r_{y,1} & r_{y,2} & r_{y,3} & r_{y,4} \\ -r_{x,1} & -r_{x,2} & -r_{x,3} & -r_{x,4} \\ -\kappa_t & -\kappa_t & \kappa_t & \kappa_t \end{bmatrix}, \tag{3}$$

where $r_{x,i}$ and $r_{y,i}$ indicate the x and y component of $\boldsymbol{r}_i$ in the body frame, and κ_t represents the torque coefficient, a factor between thrust and drag torque generated by a single rotor.

The model in (1)-(2) takes the single rotor thrusts as input. However, when performing real-time control, the motors cannot track the commanded thrust without delay. Thus we also included motor dynamics into the model. We used the first-order model on the rotor thrust to approximate the motor dynamics, yielding

$$\dot{T}_i = \frac{1}{\sigma}(u_i - T_i), \qquad i = 1, 2, 3, 4, \tag{4}$$

where σ is the time constant identified from data, and u_i is the command for the i-th rotor.

B. Fault-Tolerant Model Predictive Control

We define the state of the quadrotor $\boldsymbol{x} = [\boldsymbol{p}^T\ \boldsymbol{q}^T\ \boldsymbol{v}^T\ \boldsymbol{\omega}^T\ \boldsymbol{t}^T]^T$ and the input $\boldsymbol{u} = [u_1\ u_2\ u_3\ u_4]^T$. Then the discrete dynamics of the quadrotor $\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k)$ could be obtained by discretizing (1) and (4).

NMPC optimizes the following cost function with constraints on system dynamics and input bounds to solve for the optimal control input sequence $\boldsymbol{u}_{k:k+N-1}$:

$$\begin{aligned} \min_{\boldsymbol{u}_{k:k+N-1}} \ & \boldsymbol{y}_N^T \boldsymbol{Q}_N \boldsymbol{y}_N + \sum_{i=k}^{k+N-1} \boldsymbol{y}_i^T \boldsymbol{Q} \boldsymbol{y}_i + \boldsymbol{u}_i^T \boldsymbol{R} \boldsymbol{u}_i \\ \text{s.t.} \ & \boldsymbol{x}_{i+1} = f(\boldsymbol{x}_i, \boldsymbol{u}_i) \quad i = k, k+1, \ldots, k+N-1 \\ & \underline{\boldsymbol{u}} \le \boldsymbol{u} \le \bar{\boldsymbol{u}}. \end{aligned} \tag{5}$$

where k is the current time step, N is the number of sampled time steps in the prediction horizon, and $\boldsymbol{x}_{k:k+N}$ is the predicted state trajectory. $\boldsymbol{Q}$, $\boldsymbol{Q}_N$ and $\boldsymbol{R}$ are positive-definite weight matrices. $\underline{\boldsymbol{u}} = \boldsymbol{0}_{4\times 1}$ is the lower bound of the thrust command, while $\bar{\boldsymbol{u}}$ is the upper bound of the thrust command. During normal flight $\bar{\boldsymbol{u}} = T_{max}\boldsymbol{1}_{4\times 1}$, and if the i-th rotor fails, we set the i-th element of $\bar{\boldsymbol{u}}$ to $\bar{\boldsymbol{u}}_i = \underline{\boldsymbol{u}}_i = 0$. Hence NMPC is aware of the fact that the i-th motor is unable to generate thrust.

The cost function consists of two components: a running cost that accumulates at every time step as a function of $\boldsymbol{x}$ and $\boldsymbol{u}$, and a terminal cost that only depends on the terminal state.

For a quadrotor with a failed rotor, it is impossible to fully control the attitude. In order to define the cost on attitude, we first compute the attitude error $\boldsymbol{q}_w = \boldsymbol{q}_{ref} \circ \boldsymbol{q}^{-1}$ from the reference ($\boldsymbol{q}_{ref}$) and actual ($\boldsymbol{q}$) attitude. then we split the attitude error into z and xy rotation according to $\boldsymbol{q}_e = \boldsymbol{q}_z \circ \boldsymbol{q}_{xy}$ where $\boldsymbol{q}_z = \begin{bmatrix} q_{z,w} & 0 & 0 & q_{z,z} \end{bmatrix}^\intercal$ and $\boldsymbol{q}_{xy} = \begin{bmatrix} q_{xy,w} & q_{xy,x} & q_{xy,y} & 0 \end{bmatrix}^\intercal$. We could then form the cost vector $\boldsymbol{y}_i$ and the block-diagonal weight matrix $\boldsymbol{Q}$ for the running costs as

$$\boldsymbol{y}_i = \begin{bmatrix} \boldsymbol{p} - \boldsymbol{p}_{ref} \\ q_{xy,x}^2 + q_{xy,y}^2 \\ q_{z,z} \\ \boldsymbol{v} - \boldsymbol{v}_{ref} \\ \boldsymbol{\omega} - \boldsymbol{\omega}_{ref} \\ \boldsymbol{t} - \boldsymbol{t}_{ref} \\ \boldsymbol{u} - \boldsymbol{u}_{ref} \end{bmatrix}, \tag{6}$$

$$\boldsymbol{Q} = \operatorname{diag}\left(\begin{bmatrix} \boldsymbol{Q}_p & Q_{xy} & Q_z & \boldsymbol{Q}_v & \boldsymbol{Q}_\omega & \boldsymbol{Q}_t & \boldsymbol{Q}_u \end{bmatrix}\right), \tag{7}$$

with all values taken at their time-step i. Additionally, the terminal cost $\boldsymbol{y}_N^T \boldsymbol{Q}_N \boldsymbol{y}_N$ is structured the same way, but without the input $\boldsymbol{u}$. The reference values in the cost functions are either hover references used for all time points in the prediction horizon, or sampled points from trajectories.

We set Q_z in (7) to zero in the rotor-failure condition, and only keep the cost on $\boldsymbol{q}_{xy}$. Thus we enforce the quadrotor to align its thrust direction with the reference while discarding the control of yaw motion.

C. Incremental Nonlinear Dynamic Inversion (INDI)

For quadrotors fault-tolerant control, the aerodynamic effects arising from the high body rates cannot be accurately described by a model that is simple enough to be inserted into the NMPC. In addition, the center of gravity position and other mechanical, electrical, or environmental effects can all result in further model mismatches.

In order to mitigate these effects, similar to previous research [30], INDI has been used to refine the thrust command from NMPC to be adaptive against model uncertainties. Specifically, INDI approximates $\boldsymbol{\tau}_{\text{ext}}$ in (1) by instantaneous sensor measurements instead of finding an accurate model that can be difficult to obtain. There are different derivations of INDI for quadrotors (see, e.g., [31], [32]). Here we adopt the version from [32]. According to the last equation of (1), we can estimate the external torques as

$$\boldsymbol{\tau}_{\text{ext}} = \boldsymbol{I}_v \dot{\boldsymbol{\omega}}_f - \boldsymbol{\tau}_f + \boldsymbol{\omega}_f \times \boldsymbol{I}_v \boldsymbol{\omega}_f \tag{8}$$

where the subscript f indicates that the variable is measured and low-pass filtered for noise reduction. $\boldsymbol{\tau}_f$ is the estimated torque obtained from (2), with the force of each rotor $\boldsymbol{t}$ obtained from the low-pass filtered rotor speed measurements. Note that all variables have a similar amount of delay as they are filtered with the same LPF. We also assume that the bandwidth of $\boldsymbol{\tau}_{\text{ext}}$ is significantly lower than the bandwidth

of the LPF. Thus the delay of $\boldsymbol{\tau}_{\text{ext}}$ is negligible. Then we substitute (8) into the last equation of (1), yielding

$$\begin{aligned} \boldsymbol{I}_v\dot{\boldsymbol{\omega}} &= \boldsymbol{I}_v\dot{\boldsymbol{\omega}}_f + \boldsymbol{\tau} - \boldsymbol{\tau}_f + (\boldsymbol{\omega}_f \times \boldsymbol{I}_v\boldsymbol{\omega}_f - \boldsymbol{\omega} \times \boldsymbol{I}_v\boldsymbol{\omega}) \\ &\approx \boldsymbol{I}_v\dot{\boldsymbol{\omega}}_f + \boldsymbol{\tau} - \boldsymbol{\tau}_f \end{aligned} \tag{9}$$

The approximation in (9) is reasonable since the actuator dynamic is significantly faster than the quadrotor body rate change.

With the thrust command $\boldsymbol{u}$ from NMPC, we can obtain the desired collective thrust T_d and angular acceleration $\boldsymbol{\alpha}_d$ from (1) and (2), yielding

$$\begin{bmatrix} T_d \\ \boldsymbol{I}_v\boldsymbol{\alpha}_d + \boldsymbol{\omega} \times \boldsymbol{I}_v\boldsymbol{\omega} \end{bmatrix} = \boldsymbol{G}\boldsymbol{u} \tag{10}$$

Replacing the left-hand side of (9) by the desired angular acceleration $\boldsymbol{\alpha}_d$ obtained from (10), we can calculate the desired torque:

$$\boldsymbol{\tau}_d = \boldsymbol{\tau}_f + \boldsymbol{I}_v\left(\boldsymbol{\alpha}_d - \dot{\boldsymbol{\omega}}_f\right), \tag{11}$$

Finally, the thrust command of each rotor can be obtained by

$$\boldsymbol{u}_{indi} = \hat{\boldsymbol{G}}^{+}\begin{bmatrix} T_d \\ \boldsymbol{\tau}_d \end{bmatrix} \tag{12}$$

where $\hat{\boldsymbol{G}}$ is the reduced control effectiveness matrix by setting the i-th column of $\boldsymbol{G}$ to zero. As such, the final thrust command $\boldsymbol{u}_{indi}$ of the damaged (i-th) rotor is always zero. Since $\hat{\boldsymbol{G}}$ is not full rank, we use Moore-Penrose inversion in (12).

III. Simulations

To validate the effectiveness of the proposed fault-tolerant controller, we first conduct experiments in a simulation environment. We also compare the proposed NMPC against a benchmark method [21] in the same simulation environment. The benchmark is a cascaded nonlinear controller that has been validated in real-world experiments withstanding strong external wind disturbances. Afterward, real-world experiments will be performed and described in Sec. IV.

Method	Avg. iter time (ms)	Max. iter time (ms)
Benchmark	1.122e-2	0.5459
NMPC	6.431	25.75

TABLE I: Average and maximum iteration computing of proposed NMPC controller and the benchmark controller. Measured during simulations on a Jetson TX2.

Weight	Value	Weight	Value
$\boldsymbol{Q}_{p,xy}$	80	$\boldsymbol{Q}_{p,z}$	800
$\boldsymbol{Q}_{xy}$	60	$\boldsymbol{Q}_{v}$	1
$\boldsymbol{Q}_{\omega,xy}$	0.5	$\boldsymbol{Q}_{\omega,z}$	0.1
$\boldsymbol{Q}_{t}$	3.0	$\boldsymbol{Q}_{u}$	1.0

TABLE II: Weight selection of the NMPC cost function

A. Implementation Details

For both simulations and real-world experiments, the control algorithm is implemented in Agilicious [33], a software and hardware co-designed platform for agile autonomous quadrotor flight. The NMPC controller is solved by a sequential quadratic programming (SQP) algorithm executed in a real-time iteration scheme [34]. Specifically, we implement the algorithm using ACADO [35] with qpOASES [36] as the solver. In both the simulations and the real-world experiments, we set the weight of the NMPC cost function to be the values in Table II. A prediction horizon of $1\,\text{s}$ is used, and discretized into 20 steps. The position tracking error is clamped to prevent divergence of the NMPC solver.

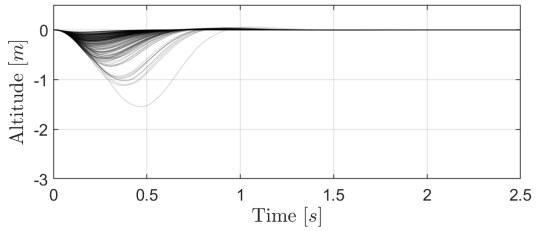
A mode switch mechanism is added to the software to simulate the rotor failure. Once the switch is triggered externally, the controller command for one of the motors is clamped to zero immediately. Since there are many effective and rapid quadrotor failure detection methods (see, e.g., [11], [10]), the fault-detection problem is out of the scope of this paper. Instead, the index of the failed rotor is directly sent to the flight controller when the failure is triggered through a mock fault detector.

In order to evaluate the computation time of the proposed and the benchmark method, the simulations have been performed on the NVIDIA Jetson TX2, the same onboard computing unit we use in the real-world experiments. The computational time for the controller is recorded and listed in Table I. It is clear from the data that the NMPC has a much longer iteration time than the nonlinear controller. However, on average the frequency of the NMPC reaches more than $150\,\text{Hz}$, which we found enough for the control of quadrotor in the tests. The maximum iteration time of the NMPC reaches $25\,\text{ms}$, but it only appears at the first iteration after the rotor failure is triggered. As the real-time iteration scheme is used, the internal states of the solver have to be reinitialized to ensure no violation of the updated constraint. Thus this particular iteration takes a longer time than average.

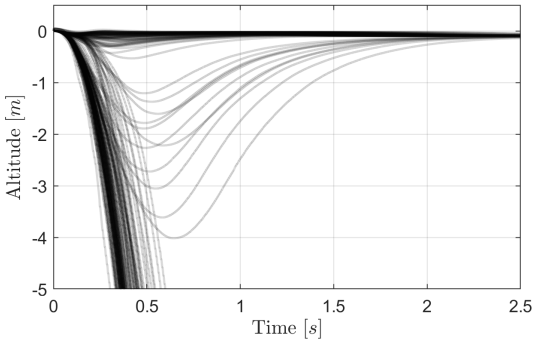
B. Recovery from Randomized Orientations

Extensive randomized simulations are performed to compare the ability to recover a damaged quadrotor from different initial orientations. In the simulations, the quadrotor is initialized at random orientations with a rotor failure and commanded to hover at the same position. The initial orientations of the quadrotor are sampled from a uniform distribution [37], and the velocity and angular rates are initialized with zeros.

The recovery simulation is performed 200 times with both the proposed NMPC and the benchmark method. The altitude profiles in these flights are shown in 3. In all 200 randomized tests, the quadrotor is able to recover from the initial orientation when controlled by the NMPC. With the benchmark method, however, it only succeeded in 85 of the tests. The recovery time and the loss of height in the recovery process are also significantly lower using the proposed NMPC. We sorted the 200 tested initial orientations based on the cosine of the tilt angle, and computed the success rate of the recoveries in different intervals of initial tile angles. The success rate is



(a) NMPC



(b) Benchmark

Fig. 3: Time history of the relative altitude with respect to the initial point in the randomized recovery tests using (a) proposed NMPC, and (b) benchmark controller.

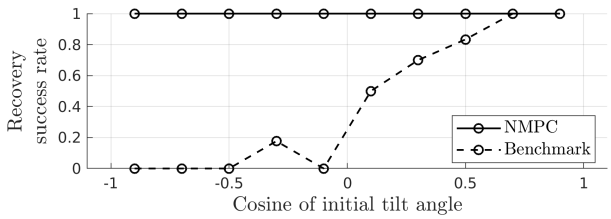


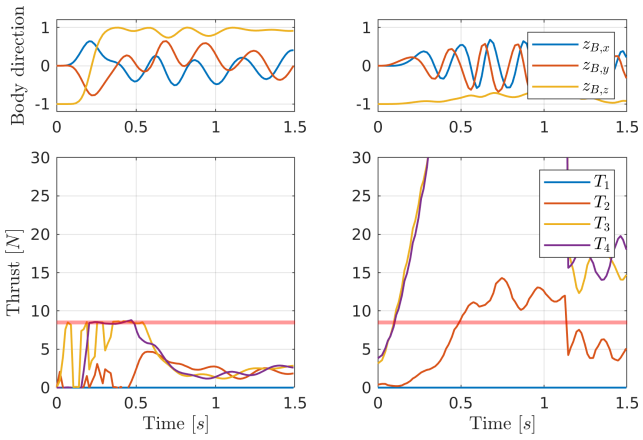
Fig. 4: The success rate of recovery using the proposed NMPC and the benchmark method. Each point shows the success rate of recovery when the cosine of the initial tile angle falls in the corresponding interval.

plotted in Fig. 4. The data shows that the benchmark method is only able to recover the quadrotor if the initial orientation is close to the hovering orientation, while the success rate is almost zero if the quadrotor's initial direction is more than 90° tilted. Figure 5 shows the attitude of the quadrotor when trying to recover it from an upside-down direction with either controller. It is observed that the benchmark controller, which does not take into account the actuator limits, fails to recover the quadrotor after experiencing actuator saturation. NMPC, however, recovers the quadrotor without violating the actuator constraints.

IV. Real-World Experiments

A. Experiment Setup

For real-world experiments, we validate the proposed algorithm in an instrumented flight arena [38], which is equipped with a VICON motion tracking system providing the position and orientation of the quadrotor at 400 Hz. The quadrotor we used in the experiments is shown in Fig. 6. It is actuated by four motors equipped with 5-inch propellers, each providing a maximum of 8.5 N lift force. The state estimation and control



(a) NMPC (b) Benchmark

Fig. 5: Quadrotor attitude and thrust commands during a recovery test in simulation. Initial attitude is 180° flipped. **(a)** controlled by NMPC, **(b)** controlled by the benchmark controller. The bold red line in the thrust plots shows the actuator limit. The attitude of the quadrotor is represented by the x, y, and z components (in the inertial frame) of z_B, a unit vector in the thrust direction.

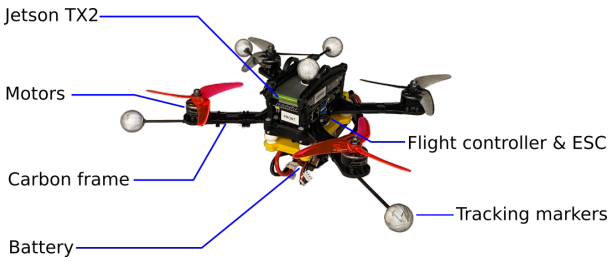


Fig. 6: Photo of the tested quadrotor platform.

runs on an NVIDIA Jetson TX2 system-on-a-chip with a frequency of 150 Hz. The low-level control is done with a Radix FC[1] flight control, which also sends the IMU and motor speed measurements to the Jetson at 500 Hz. The quadrotor is powered with a 4-cell LiPo battery and has a total mass of 0.75 kg. Fig. 7 presents the system diagram for real-world experiments.

B. Rotor Failure during Hovering Flight

In the first experiment, we trigger the rotor failure while the quadrotor is hovering. After the failure, the reference is set to hover at the same position.

Fig. 8 shows the quadrotor states in this test. The failure is triggered at $t = 0.5$ s, and the quadrotor subsequently starts yaw spinning. Even so, the position in x and y directions are effectively controlled. Within 2 s after the failure, the quadrotor enters the stable rotating equilibrium. We observe that the gain selection of the NMPC controller can influence the behavior in the controlled flights. In general, with higher weights on the attitude error and pitch and roll rates, the controller leads to a dynamic equilibrium with a smaller inclination angle and a higher yaw rate.

[1]https://www.brainfpv.com/product/radix-fc/

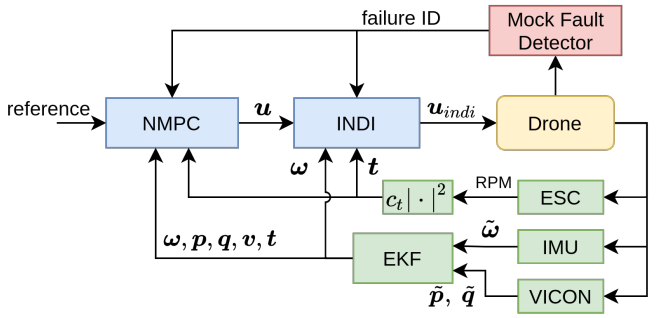


Fig. 7: System diagram in real-world experiments. An NMPC block generates the initial thrust command of rotors, followed by an INDI block to modify these commands for robustification. The full states are obtained by an extended Kalman filter (EKF) fusing measurements from an inertia measurement unit (IMU) and a motion capture system (VICON). The thrust of each rotor is calculated by a quadratic model with the coefficient c_t, and rotor speeds (RPM) measured by the electronic speed controller (ESC). While the failures are triggered manually for our experiments, a fault detection algorithm (e.g., [11], [10]) could replace the mock fault detector.

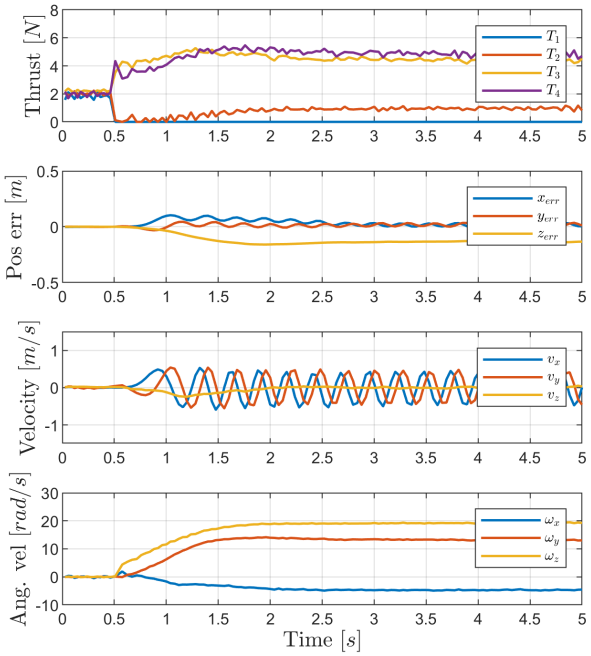


Fig. 8: Time history of quadrotor states controlled by the proposed fault-tolerant controller. Failure of the 1st rotor (T_1) is triggered at $t = 0.5\,\mathrm{s}$. The damaged quadrotor is then stabilized at its original position.

C. Trajectory Tracking under Rotor Failure

Instead of hovering at a single waypoint, we test the performance of our controller in tracking agile trajectories with the top speed at $5\,\mathrm{m\,s^{-1}}$. A "lemniscate" reference trajectory is selected, which is defined by the following parametric equation:

$$x(t) = 4\sin(\omega t), \quad y(t) = 2\sin(2\omega t), \tag{13}$$

where ω is a constant to control the reference velocity. We tested with $\omega_1 = 0.3535$ and $\omega_2 = 0.8838$, such that the maximum velocity of the reference trajectory is $2\,\mathrm{m\,s^{-1}}$ and $5\,\mathrm{m\,s^{-1}}$ respectively.

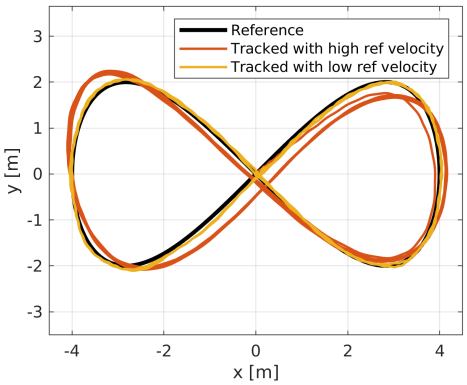


Fig. 9: Reference trajectory and tracked trajectory with maximum reference velocity $5.0\,\mathrm{m\,s^{-1}}$ and $2.0\,\mathrm{m\,s^{-1}}$ respectively

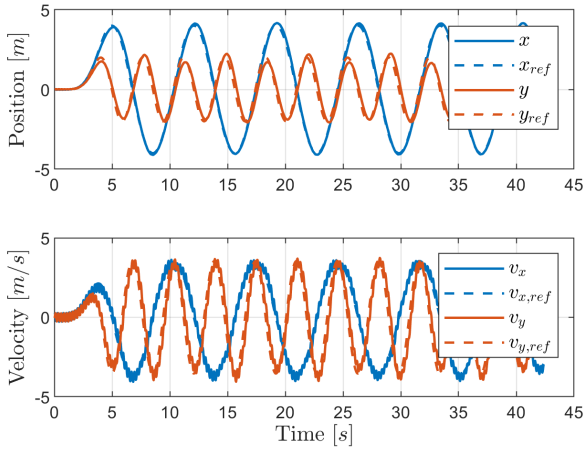


Fig. 10: Quadrotor position and velocity during tracking the trajectory with a max velocity at $5\,\mathrm{m\,s^{-1}}$

The tracking performance of these two trajectories is shown in Fig. 9. The quadrotor is able to track the slower trajectory accurately, despite the complete loss of a single rotor. As is expected, tracking the faster trajectory leads to a higher tracking error. Be that as it may, this experiment validates the performance of tracking a pre-designed trajectory at up to 5 m/s, showing the ability of a quadrotor to navigate home or finish the delivery task even if a rotor has failed.

Fig. 10 shows the position and velocity of the center of gravity. According to Fig. 10, the desired position of the quadrotor is followed with a small delay. The high-frequency oscillations in the velocity plots are caused by the spinning motion of the quadrotor. We hypothesize that the inadequate tracking performance of the quadrotor is caused by the model uncertainties from aerodynamic effects [39]. Even though an INDI controller is implemented, it does not compensate for the aerodynamic forces on the damaged quadrotor. The effect of these aerodynamics on the trajectory tracking performance still needs to be further justified.

D. Rotor Failure during Agile Flights

As mentioned before, a major advantage of the NMPC is the ability to exploit the full nonlinear dynamics of the quadrotor. Thus it can recover the damaged quadrotor from extreme conditions largely deviated from the hovering condition. Similarly to the simulations performed in Sec. III, we conducted two real-world experiments:

- i) The rotor failure occurs when the drone is flying upside-down on the apex of a vertical flipping maneuver.
- ii) The rotor failure occurs when the drone is tracking an agile racing trajectory.

Fig. 11 shows the quadrotor trajectory in the first scenario, namely recovery during a flipping maneuver. A snapshot of the experiment is shown in Fig. 1. The first two plots in Fig. 11 present the position of the quadrotor during the post-failure recovery process; the third plot shows the quadrotor orientation with curves showing the x, y, and z components of the thrust direction $\boldsymbol{z}_B$ in the inertial frame; and the last plot shows the thrust command of the controller. In this experiment, the quadrotor reaches the upside-down orientation at $t \approx 0.5\,\mathrm{s}$. At this moment, failure of the 1st rotor is triggered, and the quadrotor is immediately commanded to recover to the hovering state. Right after the moment when the failure happens, T_3 is set as 5 N while T_2 and T_4 are commanded as zero, such that the quadrotor can quickly flip to recover the orientation. Then the controller increases T_4 to stop the drone from flipping. Finally, the high thrusts on both T_3 and T_4 lead to a high yaw rate that further stabilizes the quadrotor leveraging the gyroscopic effect, and the position is consequently controlled. The above process only lasts for $2\,\mathrm{s}$, and the height of the damaged quadrotor is only dropped by $0.9\,\mathrm{m}$.

Finally, we tested triggering the rotor failure while tracking an agile racing trajectory at speed up to $16\,\mathrm{m\,s^{-1}}$. Fig. 12 shows the trajectory of the quadrotor during the experiment. Failure of the 1st rotor is triggered when the quadrotor reaches a designated position. At this moment, the quadrotor is almost $90°$ inclined, and is flying at over $7.5\,\mathrm{m\,s^{-1}}$. Then the fault-tolerant controller successfully recovers the pose of the drone in spite of the large translational speed and the resultant aerodynamic effects. The entire recovery procedure lasts for $3\,\mathrm{s}$. In the end, the damaged quadrotor is commanded to return and land.

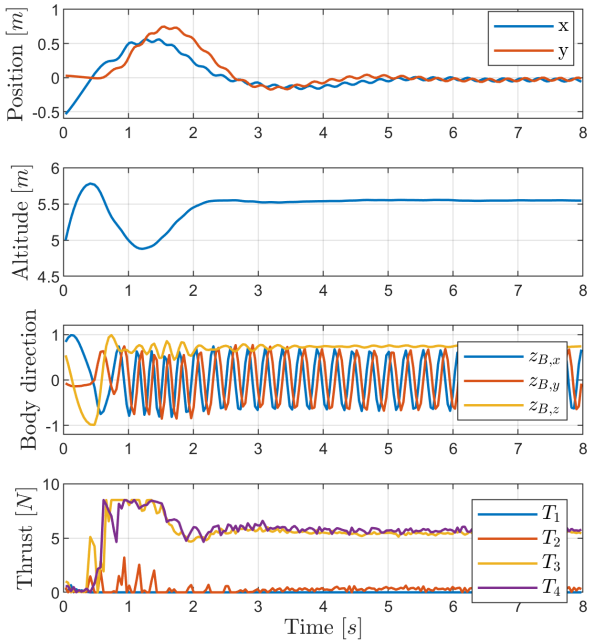


Fig. 11: Time history of quadrotor states during the post-failure recovery process from an upside-down condition. Failure is triggered at $t = 0.5\,\mathrm{s}$. Positional origin is shifted in x and y directions to the point where the rotor failure happened.

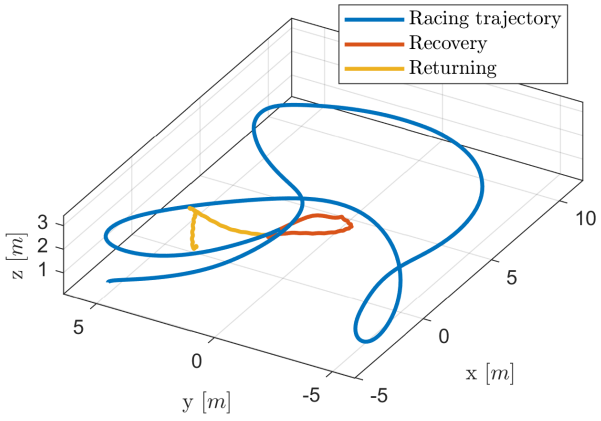


Fig. 12: The 3D trajectory of the quadrotor recovering when experiencing a rotor failure in a racing trajectory

V. Conclusions

In this work, we proposed a nonlinear model predictive control framework for quadrotor fault-tolerant flight. The NMPC method can effectively stabilize the quadrotor subjected to the complete failure of a single rotor. The proposed method does not use a cascaded structure, nor rely on linear assumptions. Instead, it takes the full nonlinear model of the quadrotor, and takes into account the constraints of each rotor. Extensive tests against traditional nonlinear control method have showed that the NMPC is more robust to the initial condition where the rotor failure happens. The proposed method can recover the damaged quadrotor when a single rotor failure happens at arbitrary orientation or during an agile flight. The method is also able to control the position of the damaged quadrotor and even follow a trajectory.

Acknowledgment

The authors would like to thank Thomas Längle, Leonard Bauersfeld, and Jiaxu Xing for their help in experiments, photo rendering, and data collection.

References

[1] G. Loianno and D. Scaramuzza, "Special issue on future challenges and opportunities in vision-based drone navigation," *Journal of Field Robotics*, vol. 37, no. 4, pp. 495–496, 2020.

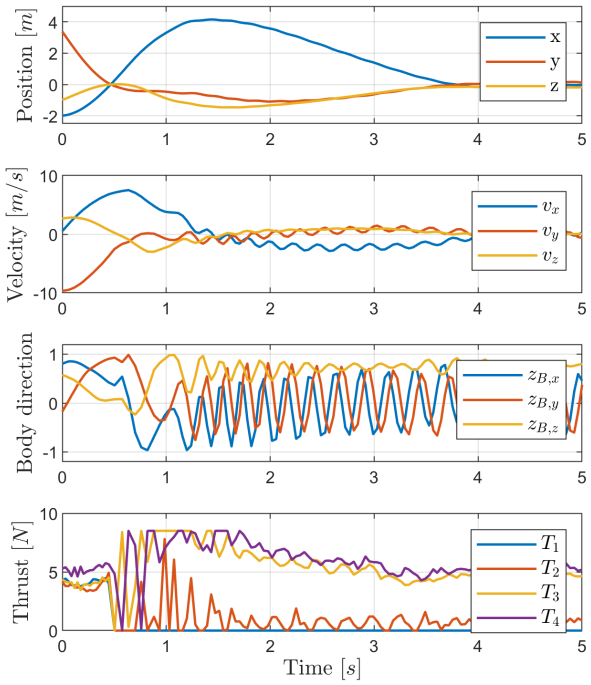


Fig. 13: Time history of the quadrotor states during the recovery process when experiencing a rotor failure in a racing trajectory. Failure is triggered at $t = 0.5\,\mathrm{s}$. Positional origin is shifted to the point where the rotor failure happened.

[2] S. Rajendran and S. Srinivas, "Air taxi service for urban mobility: A critical review of recent developments, future challenges, and opportunities," *Transportation Research Part E: Logistics and Transportation Review*, vol. 143, p. 102090, Nov 2020. arXiv: 2103.01768.

[3] "UAS by numbers." https://www.faa.gov/uas/resources/by_the_numbers/, Aug 2021.

[4] Newswire, "Global drone service market report 2019: Market is expected to grow from usd 4.4 billion in 2018 to usd 63.6 billion by 2025, at a cagr of 55.9%." Available at: https://bit.ly/3zUewze, Apr 2019.

[5] E. Ackerman, "Swiss post suspends drone delivery service after second crash," *IEEE Spectrum*, July 2019.

[6] X. Wang, S. Sun, E.-J. van Kampen, and Q. Chu, "Quadrotor fault tolerant incremental sliding mode control driven by sliding mode disturbance observers," *Aerospace Science and Technology*, vol. 87, pp. 417–430, 2019.

[7] L. Besnard, Y. B. Shtessel, and B. Landrum, "Quadrotor vehicle control via sliding mode controller driven by sliding mode disturbance observer," *Journal of the Franklin Institute*, vol. 349, no. 2, pp. 658–684, 2012.

[8] D. Xu, J. F. Whidborne, and A. Cooke, "Fault tolerant control of a quadrotor using l1 adaptive control," *International Journal of Intelligent Unmanned Systems*, 2016.

[9] Y. Guo, B. Jiang, and Y. Zhang, "A novel robust attitude control for quadrotor aircraft subject to actuator faults and wind gusts," *IEEE/CAA Journal of Automatica sinica*, vol. 5, no. 1, pp. 292–300, 2017.

[10] M. H. Amoozgar, A. Chamseddine, and Y. Zhang, "Experimental test of a two-stage kalman filter for actuator fault detection and diagnosis of an unmanned quadrotor helicopter," *Journal of Intelligent & Robotic Systems*, vol. 70, no. 1, pp. 107–117, 2013.

[11] A. Freddi, S. Longhi, and A. Monteriu, "Actuator fault detection system for a mini-quadrotor," in *2010 IEEE International Symposium on Industrial Electronics*, pp. 2055–2060, IEEE, 2010.

[12] A. Freddi, A. Lanzon, and S. Longhi, "A Feedback Linearization Approach to Fault Tolerance in Quadrotor Vehicles," *IFAC Proceedings Volumes*, vol. 44, pp. 5413–5418, Jan. 2011.

[13] V. Lippiello, F. Ruggiero, and D. Serra, "Emergency landing for a quadrotor in case of a propeller failure: A PID based approach," in *2014 IEEE International Symposium on Safety, Security, and Rescue Robotics (2014)*, pp. 1–7, Oct. 2014.

[14] M. W. Mueller and R. D'Andrea, "Stability and control of a quadrocopter despite the complete loss of one, two, or three propellers," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 45–52, May 2014.

[15] J. Stephan, L. Schmitt, and W. Fichter, "Linear parameter-varying control for quadrotors in case of complete actuator loss," *Journal of Guidance, Control, and Dynamics*, vol. 41, no. 10, pp. 2232–2246, 2018.

[16] M. W. Mueller and R. D'Andrea, "Relaxed hover solutions for multicopters: Application to algorithmic redundancy and novel vehicles," *The International Journal of Robotics Research*, vol. 35, no. 8, pp. 873–889, 2016.

[17] C. de Crousaz, F. Farshidian, M. Neunert, and J. Buchli, "Unified motion control for dynamic quadrotor maneuvers demonstrated on slung load and rotor failure tasks," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2223–2229, May 2015.

[18] A. Lanzon, A. Freddi, and S. Longhi, "Flight control of a quadrotor vehicle subsequent to a rotor failure," *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 2, pp. 580–591, 2014.

[19] S. Sun, G. Cioffi, C. De Visser, and D. Scaramuzza, "Autonomous quadrotor flight despite rotor failure with onboard vision sensors: Frames vs. events," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 580–587, 2021.

[20] V. Lippiello, F. Ruggiero, and D. Serra, "Emergency landing for a quadrotor in case of a propeller failure: A backstepping approach," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4782–4788, Sept. 2014.

[21] S. Sun, X. Wang, Q. Chu, and C. de Visser, "Incremental nonlinear fault-tolerant control of a quadrotor with complete loss of two opposing rotors," *IEEE Transactions on Robotics*, vol. 37, no. 1, pp. 116–130, 2020.

[22] S. Sun, L. Sijbers, X. Wang, and C. De Visser, "High-Speed Flight of Quadrotor Despite Loss of Single Rotor," *IEEE Robotics and Automation Letters*, vol. PP, pp. 1–1, June 2018.

[23] P. Foehn, A. Romero, and D. Scaramuzza, "Time-optimal planning for quadrotor waypoint flight," *Science Robotics*, vol. 6, no. 56, p. eabh1221, 2021.

[24] M. Bangura and R. Mahony, "Real-time Model Predictive Control for Quadrotors," *IFAC Proceedings Volumes*, vol. 47, pp. 11773–11780, Jan. 2014.

[25] M. Kamel, K. Alexis, M. Achtelik, and R. Siegwart, "Fast nonlinear model predictive control for multicopter attitude tracking on SO(3)," in *2015 IEEE Conference on Control Applications (CCA)*, pp. 1160–1166, Sept. 2015.

[26] Y. Aoki, Y. Asano, A. Honda, N. Motooka, and T. Ohtsuka, "Nonlinear Model Predictive Control of Position and Attitude in a Hexacopter with Three Failed Rotors," *IFAC-PapersOnLine*, vol. 51, pp. 228–233, Jan. 2018.

[27] W. Jung and H. Bang, "Fault and Failure Tolerant Model Predictive Control of Quadrotor UAV," *International Journal of Aeronautical and Space Sciences*, vol. 22, pp. 663–675, June 2021.

[28] M. L. Darby and M. Nikolaou, "MPC: Current practice and challenges," *Control Engineering Practice*, vol. 20, pp. 328–342, Apr. 2012.

[29] U. Eren, A. Prach, B. B. Koçer, S. V. Raković, E. Kayacan, and B. Açıkmeşe, "Model predictive control in aerospace systems: Current state and opportunities," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 7, pp. 1541–1566, 2017.

[30] S. Sun, A. Romero, P. Foehn, E. Kaufmann, and D. Scaramuzza, "A comparative study of nonlinear mpc and differential-flatness-based control for quadrotor agile flight," *arXiv preprint arXiv:2109.01365*, 2021.

[31] E. J. Smeur, Q. Chu, and G. C. de Croon, "Adaptive incremental nonlinear dynamic inversion for attitude control of micro air vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 39, no. 3, pp. 450–461, 2016.

[32] E. Tal and S. Karaman, "Accurate tracking of aggressive quadrotor trajectories using incremental nonlinear dynamic inversion and differential flatness," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 3, pp. 1203–1218, 2020.

[33] Robotics and Perception Group, "Agilicious." https://agilicious.dev/.

[34] M. Diehl, H. G. Bock, H. Diedam, and P.-B. Wieber, "Fast direct multiple shooting algorithms for optimal robot control," in *Fast motions in biomechanics and robotics*, pp. 65–93, Springer, 2006.

[35] B. Houska, H. Ferreau, and M. Diehl, "ACADO Toolkit – An Open Source Framework for Automatic Control and Dynamic Optimization,"

Optimal Control Applications and Methods, vol. 32, no. 3, pp. 298–312, 2011.

[36] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpoases: A parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, no. 4, pp. 327–363, 2014.

[37] J. Arvo, "Iii.4 - fast random rotation matrices," in *Graphics Gems III (IBM Version)* (D. KIRK, ed.), pp. 117–120, San Francisco: Morgan Kaufmann, 1992.

[38] UZH Robotics and Perception Group, "The World's Largest Indoor Drone-Testing Arena." https://www.youtube.com/watch?v=EV4ACi5ZO2k, June 2021.

[39] S. Sun and C. de Visser, "Aerodynamic Model Identification of a Quadrotor Subjected to Rotor Failures in the High-Speed Flight Regime," *IEEE Robotics and Automation Letters*, vol. 4, pp. 3868–3875, Oct. 2019. Conference Name: IEEE Robotics and Automation Letters.