drones

*Article*

# Fractional-Order Control Algorithm for Tello EDU Quadrotor Drone Safe Landing during Disturbance on Propeller

**Nurfarah Hanim Binti Rosmadi [1], Kishore Bingi [1,*], P. Arun Mozhi Devan [1], Reeba Korah [2], Gaurav Kumar [2], B Rajanarayan Prusty [3] and Madiah Omar [4]**

1 Department of Electrical and Electronics Engineering, Universiti Teknologi PETRONAS, Seri Iskandar 31750, Malaysia; nurfarah_20001736@utp.edu.my (N.H.B.R.); arun.selvam@utp.edu.my (P.A.M.D.)
2 Department of Electronics and Communication Engineering, Alliance College of Engineering and Design, Alliance University, Bangalore 562106, India; reeba.korah@alliance.edu.in (R.K.); gaurav.kumar@alliance.edu.in (G.K.)
3 Department of Electrical, Electronics and Communication Engineering, School of Engineering, Galgotias University, Greater Noida 203201, India; b.r.prusty@ieee.org
4 Department of Chemical Engineering, Universiti Teknologi PETRONAS, Seri Iskandar 31750, Malaysia; madiah.omar@utp.edu.my
* Correspondence: bingi.kishore@utp.edu.my

**Abstract:** Quadcopter drones have become increasingly popular because of their versatility and usefulness in various applications, such as surveillance, delivery, and search and rescue operations. Weather conditions and obstacles can undoubtedly pose challenges for drone flights, sometimes causing the loss of one or two propellers. This is a significant challenge as the loss of one or more propellers leads to a sudden loss of control, potentially resulting in a crash, which must be addressed through advanced control strategies. Therefore, this article develops and implements a fractional-order control algorithm to enhance quadrotor drones' safety and resilience during propeller failure scenarios. The research encompasses the complexities of quadrotor dynamics, fractional-order control theory, and existing methodologies for ensuring safe drone landings. The study emphasizes case validation on experimental results, where four distinct cases were tested using PID and Fractional-order PID (FOPID) controllers. These cases involve various simulated failure conditions to assess the performance and adaptability of the developed control algorithms. The results show the proposed FOPID control's superior robustness and adaptability compared to traditional PID controllers. These offer significant advancements in navigating dynamic environments and managing disruptive elements introduced during propeller failure simulations in drone control technology.

**Keywords:** quadrotor drones; fractional-order; PID; controller; propeller failure; drone safety; control algorithms



**Citation:** Rosmadi, N.H.B.; Bingi, K.; Devan, P.A.M.; Korah, R.; Kumar, G.; Prusty, B.R.; Omar, M. Fractional-Order Control Algorithm for Tello EDU Quadrotor Drone Safe Landing during Disturbance on Propeller. *Drones* **2024**, *8*, 566. https://doi.org/10.3390/drones8100566

Academic Editors: Zhihong Liu, Shihao Yan, Kehao Wang and Yirui Cong

Received: 17 August 2024
Revised: 4 October 2024
Accepted: 9 October 2024
Published: 10 October 2024



## 1. Introduction

Unmanned Aerial Vehicles (UAVs) have gained widespread adoption across various industries due to their versatility and ability to perform multiple tasks. These tasks include surveillance, search and rescue, agriculture, aerial surveying, logistics services, infrastructure inspections, warfare, aerial photography and recreational activities [1–4]. Quadrotors are popular among the various types of UAVs because of their simple design and comparatively high aerodynamic efficiency. However, quadcopters are susceptible to motor failures due to their limited rotor redundancy, making them more vulnerable than other drones [5]. Safety is a significant concern in the drone industry, and preventing quadcopters from crashing after motor failures is crucial for the industry's growth. Like other vehicles with safety concerns, drones are susceptible to uncertainties during operations. As a result, most flight plans are designed with a high degree of caution. The growing societal acceptance of drones and public perceptions of the associated risks influence this conservative approach [6].

Moreover, the complicated link between human behaviour, operational responsibilities, and the human–machine interface adds to the complexity of ensuring safe drone operations [7]. Consequently, a growing focus is on developing reliable drone response systems capable of real-time health status estimation and predicting the remaining useful life of drones, attracting increasing attention within the industry [8]. The safe operation of quadrotor drones is of utmost importance during propeller failure, which can disrupt flight dynamics, potentially leading to a loss of control and a subsequent crash [9]. This limitation has hindered quadrotor drones' broader adoption and integration across various sectors. Quadrotors are inherently under-actuated systems, which makes their control systems complex to maintain stability. This complexity is heightened during malfunctions such as propeller failures, increasing the risk of drone loss and posing potential hazards to surrounding areas [10]. Environmental disturbances, hardware and software faults, and operator error factors contribute to these failures. Since most quadrotors lack built-in fault-tolerant capabilities, specialized controllers are essential for maintaining stability during propeller failures. Mitigating these risks is critical for expanding the operational capabilities of quadrotor drones [11].

## 2. Related Works

Current research delves into various strategies, such as SafeEYE, an automated emergency landing system designed for larger drones, which utilizes real-time decision-making based on visual and vibration data analysis [12]. Additionally, fault-tolerant control algorithms have been explored to ensure flight stability and achieve controlled landings after propeller loss [13]. These algorithms typically employ control allocation and redistribution of control efforts among remaining functional propellers [14]. A practical approach involves utilizing Safe Landing Zone Detection with onboard sensors such as cameras and LiDAR. The captured data are analysed using image processing techniques to identify suitable landing areas based on surface texture, slope, and absence of obstacles. This information helps guide the drone toward the designated landing zone [15]. Additionally, researchers are exploring vision-based techniques for precision landing. These methods rely on computer vision algorithms to track visual markers or landmarks on the intended landing surface. By comparing the real-time position of these markers with their expected location, the control system can adjust the drone's descent trajectory for an accurate touchdown [16,17]. Additionally, advancements in Simultaneous Localization and Mapping contribute to safe drone landing by enabling drones to construct a real-time map of their surroundings. SLAM algorithms help plan safe landing paths and avoid obstacles during descent, particularly in unknown environments [18].

Researchers are investigating various methods to improve quadrotor safety in motor failure and propeller damage situations. One is fault-tolerant control systems, which redistribute control efforts among the remaining operational propellers to ensure stability or mission continuation. This approach involves creating specialized control algorithms that compensate for lost thrust and maintain flight stability despite propeller failures [8]. Additionally, precise propeller aerodynamic models are being developed, considering factors such as freestream influence and tilting rotors to enhance efficiency and stability, particularly during controlled crash landings [19]. Machine learning is crucial in empowering drones to adapt to changing flight dynamics and detect propeller failure in real-time. Reinforcement learning allows drones to adaptively learn and respond to changing flight dynamics resulting from propeller loss. Meanwhile, deep learning, with its ability to process data rapidly, is utilized for real-time propeller failure detection, eliminating the need for additional hardware [20]. Additionally, fail-safe architectures prioritize safe landing using only three or even two functional propellers by manipulating the drone's orientation and utilizing PID controllers for altitude and attitude control [21].

The fractional-order control shows promise in addressing challenging control problems such as quick response, overshooting, and resonance, which are common issues in classical control approaches. Additionally, it exhibits remarkable capabilities in suppressing chaotic behaviours observed in real-world system models, unlocking a new level of complexity in modelling and controlling dynamic systems. Traditional integer-order control methods may not fully capture the complex dynamics of a failing drone [22]. Several studies have explored the use of fractional-order modelling to understand the complex dynamics of quadrotors better. Saif et al. have proposed a fractional-order model that incorporates motor dynamics and propeller inertia using fractional-order derivatives. This model has improved accuracy compared to integer-order models, particularly in capturing transient responses [23].

Lu et al. have introduced a fractional-order modelling technique for quadrotors with actuator saturation, demonstrating its effectiveness in capturing non-linear behaviour [24]. Timis et al. introduce a fractional-order PID controller designed for quadrotors experiencing motor failure, exhibiting superior performance in settling time and robustness to disturbances compared to traditional PID controllers [25]. A study by Saif et al. proposes a fractional-order sliding mode control strategy for quadrotors facing the partial loss of control authority, achieving faster convergence and better robustness to uncertainties than integer-order SMC controllers [26]. The overall summary of the literature is given in Table 1, and these studies highlight the current potential of fractional-order models in achieving more precise control of quadrotors and are motivated towards this research. These findings motivated the exploration of fractional-order control algorithms, which provide increased flexibility in modelling system behaviour and have the potential to result in improved response times, stability, and robustness compared to traditional approaches.

**Table 1.** Summary of different controllers application on the quadrotor.

| Ref. | Drone | Objective | Control Technique | Tool | Validation Method |
|---|---|---|---|---|---|
| [9] | Custom made drone | To counteract the ground effect after blade damage | Disturbance observer PID control, $H\infty$ and Sliding mode observer | - | Experiment |
| [12] | DJI Matrice 600 | Emergency landing system | if/else and Neutral network | Python | Experiment |
| [19] | Quadrotor model | Safe landing using fixed tilting angle to the rotors | LQR control | - | Simulation |
| [27] | Parrot AR.Drone | Safe landing with minimal physical damage | Linear Quadratic Regulator (LQR) | MATLAB | Hardware-in-the-loop |
| [28] | Multirotor drone | Safe and obstacle avoidance landing using AI | Yolo v3 | OpenCV | Experiment |
| [29] | Intel Aero drone | Precise landing using neural control with ground effects | Feedback linearization controller | PyTorch | Simulation and Experiment |
| [30] | Custom made drone | To maintain drones position even upon losing one or two propellers | Fault-tolerant PID control and Model Predictive Control | MATLAB | Simulation and Experiment |
| [31] | Custom made drone | To perform safety checks and weight measurement on a landing platform | - | Python 3.7 | Simulation and Experiment |

**Table 1.** *Cont.*

| Ref. | Drone | Objective | Control Technique | Tool | Validation Method |
|---|---|---|---|---|---|
| [32] | Quadrotor model | To identify propeller failures in mid-flight | Reinforcement Learning based PD Control | RaisimGym quadcopter environment | Simulation |
| [33] | DJI Phantom 3 model | UAV impact assessment on aircraft engines for safe operation | - | CFD Simulation | Simulation |
| [34] | AR Drone 2 | Emergency controller design for quadrotor to trirotor conversion to avoid total failure | PID | MATLAB | Experiment |
| [35] | Custom made drone | Develop a collision recovery control strategy upon impact with a wall | LQR control | MATLAB | Hardware-in-the-loop |

The main contributions of this research article are listed as follows:

- The study aims to develop a comprehensive plan to ensure the safe landing of quadrotor drones in the event of propeller failure, using the Tello EDU quadrotor drone for testing control techniques in diverse indoor environments.
- Different propeller failure scenarios are created using commonly available materials (masking tape, paper clip, rubber band, and small stone) that affect the drone's performance during the flight.
- The main goal is to develop a fractional-order PID (FOPID) control strategy to adapt the drone's flight trajectory and orientation in case of propeller failure, ensuring a quadrotor drone's stability and safe landing.
- The proposed FOPID is implemented on a real Tello EDU Quadrotor to test its ability to follow a designated line while experiencing propeller failures in guiding the drone along the intended path despite disturbances and instability caused by propeller loss, and it performed more effectively than the conventional PID.

## 3. Methodology

This section initially discusses the hardware and software components and propeller failure scenarios used in this research, followed by developing the proposed control algorithm and its integration techniques.

### *3.1. Quadrotor Drone's Hardware and Software Configurations*

The Tello EDU quadrotor drone, as shown in Figure 1, is the central hardware platform for developing and testing the proposed fractional-order control algorithm to ensure safe landing during propeller loss. Its robust sensor suite and programmable interface make it an ideal drone for developing and testing the fractional-order control algorithm. The Tello EDU drone has gyroscopes, accelerometers, and other essential sensors that offer crucial real-time data for monitoring and feedback during flight experiments [36–38]. With specifications allowing for hovering at approximately 100 m in height and continuous flight for about 15 min, the drone provides ample flexibility for experimentation.

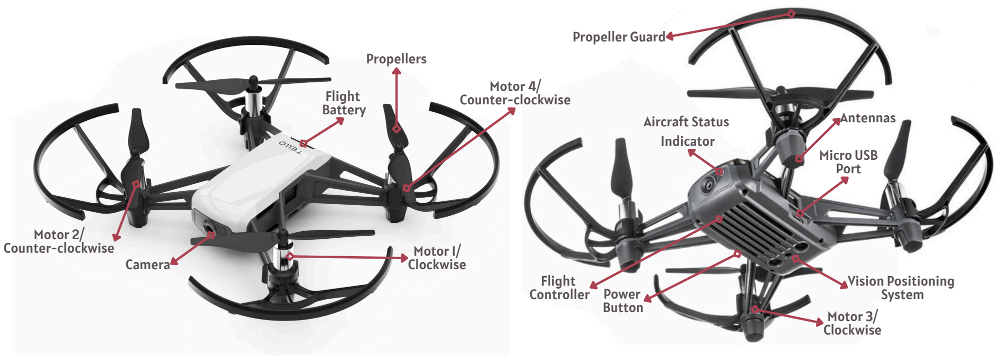


**Figure 1.** Structural description of the Tello EDU quadrotor drone.

The Python programming environment, notably the PyCharm IDE, is chosen for algorithm development and implementation on Tello EDU. PyCharm provides an extensive development environment with advanced debugging capabilities for complex drone control tasks. Additionally, specialized libraries for drone control, such as 'TelloPy', have been integrated into the Python environment to ensure seamless communication with the drone's flight controller, enabling efficient algorithm deployment and testing. Leveraging these capabilities of the Tello EDU drone and Python-based software tools, this research aims to simulate and analyse propeller failure scenarios to gain insights into drone flight dynamics under adverse conditions.

### *3.2. Propeller Failure Scenarios*

This research's methodology for creating propeller failure scenarios is crucial as it aims to introduce instability into the drone's flight dynamics to evaluate its response under simulated failure conditions. Materials such as masking tape, small stones, paper clips, and rubber bands are strategically attached to the propellers to induce different propeller failure scenarios. By carefully adjusting the type and placement of these materials, a comprehensive range of failure scenarios is simulated to thoroughly assess the drone's resilience and adaptability. The initial phase of the methodology focuses on designing propeller failure scenarios and creating testing environments. Visual representations of the normal drone propeller and the materials used in the propeller failure scenarios illustrate the process, as shown in Figure 2 for reference. Given the drone's small, lightweight nature and propellers, even tiny obstacles can cause significant disturbance. Therefore, all materials used in the scenarios are small and lightweight to accurately simulate the impact of propeller failure on drone flight dynamics.

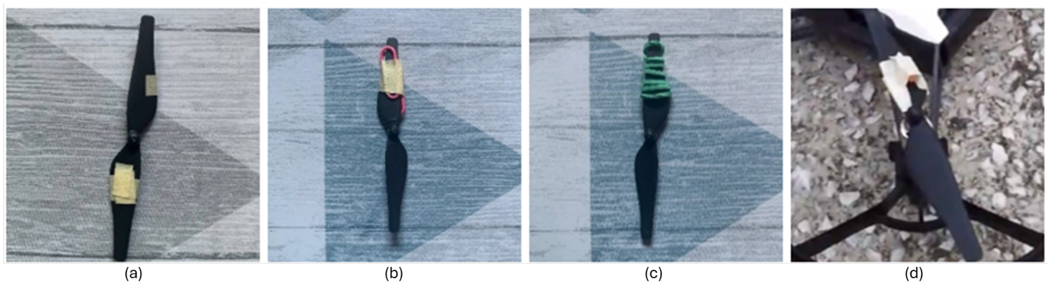


**Figure 2.** Materials used to induce different types of propeller failure: ((**a**) masking tape, (**b**) paper clip, (**c**) rubber band, and (**d**) small stone).

Further, the different indoor testing phase is a critical step in the validation process, providing a real-world evaluation of the drone's capabilities in various environmental conditions. The choice of a controlled open space ensures the testing procedures' safety and allows ample room to perform essential flight manoeuvres for a thorough assessment. This intentional environment selection enables the emulation of realistic scenarios, facilitating researchers to assess the drone's performance under conditions resembling practical settings. Testing considers external factors such as wind and environmental disturbances that significantly impact drone performance. By subjecting the drone to these variables, researchers gain a deeper understanding of the algorithm's robustness and reliability in real-world scenarios.

### *3.3. Controller Development*

After conducting initial tests to analyse the drone's behaviour in the event of propeller failure, the next step is to develop resilient control algorithms to ensure a safe landing. Developing these control algorithms is crucial for improving the drone's stability and responsiveness during propeller failure. This phase focuses on creating and fine-tuning two types of controllers: the conventional PID controller and the advanced fractional-order PID (FOPID) controller. The PID controller, known for its simplicity and effectiveness in numerous control systems, serves as a baseline for performance comparison. However, the unique capabilities of the FOPID controller, which include superior handling of system dynamics through fractional calculus, are being explored to provide enhanced stability and control precision.

#### 3.3.1. PID Controller

The PID control system incorporates three key components: the proportional term (P) with gain $K_p$, which adjusts the output in proportion to the error $E(s)$; the integral term (I) with gain $K_i$, which considers the accumulation of past errors; and the derivative term (D) with gain $K_d$, which anticipates future errors based on the current rate of change. The PID controllers' transfer function is given below.

$$C(s) = \frac{U(s)}{E(s)} = K_p + K_i \frac{1}{s} + K_d s \tag{1}$$

An auto-tune feature has been integrated into the code to ensure the controller optimally adapts to the drone's stability conditions, enabling the drone to calculate the most appropriate PID parameters for maintaining stability [39]. The controller gains have been fine-tuned using a differential evolution algorithm to minimize the cost function, which is the sum of absolute errors over time, and this process guarantees that the controller effectively maintains the drone's stability. The flowchart of the PID control shown in Figure 3 visually represents the process flow, such as reading sensor data, calculating errors, updating the PID controller, and sending control signals to the drone. Further, the block diagram of the PID controller is shown in Figure 4.

#### 3.3.2. Fractional-Order PID Controller

The FOPID control algorithm enhances the traditional PID controller by integrating fractional calculus, enabling more adaptable and precise control [22,39]. The FOPID controllers' transfer function is given below.

$$C(s) = \frac{U(s)}{E(s)} = K_p + K_i \frac{1}{s^{\alpha}} + K_d s^{\beta} \tag{2}$$

As shown in the above equation, the control signal $U(s)$ of FOPID has introduces two additional parameters, denoted as $\alpha$ and $\beta$, representing the integral and derivative fractional-order terms and its flowchart shown in Figure 5. Further, the block diagram of the PID controller is shown in Figure 4. The approximation of FOPID's fractional-order terms $s^{\alpha}$ and $s^{\beta}$ for implementation are obtained using refined Oustaloup approximation, which is computed using the following formula [40].

$$s^{\alpha} \approx \left(\frac{9\omega_h}{10}\right)^{\alpha} \left(\frac{9s^2 + 10\omega_h s}{9(1-\alpha)s^2 + 10\omega_h s + 9\alpha}\right) \prod_{j=1}^{N} \frac{s + \omega'_j}{s + \omega_j} \tag{3}$$

where

$$\omega'_j = \omega_l (\omega_h/\omega_l)^{\frac{2j-1-\alpha}{2N}}, \ \omega_j = \omega_l (\omega_h/\omega_l)^{\frac{2j-1+\alpha}{2N}}.$$

The above approximation is valid for estimating the $N^{th}$ approximation order within the lower and higher frequencies of $\omega_l$ and $\omega_h$, respectively. The parameters of approximation technique considered in this study for $N$, $\omega_l$ and $\omega_h$ are 5, $10^{-5}$, and $10^5$, respectively. The fractional parameters are determined through a rigorous trial-and-error process and determined that values of 0.98 for $\alpha$ and 0.02 for $\beta$ were the most suitable for ensuring the stability and control of the drone. These values enabled the FOPID controller to effectively manage the dynamic behaviour of the drone, providing a more resilient response to disturbances than a traditional PID controller. The FOPID controller is implemented using Python libraries such as 'numpy' for numerical operations and 'scipy' for special functions required in fractional calculus. The controller class, 'FOPID', initializes with specific gains and fractional orders and calculates the control action by combining proportional, fractional integral, and fractional derivative terms.

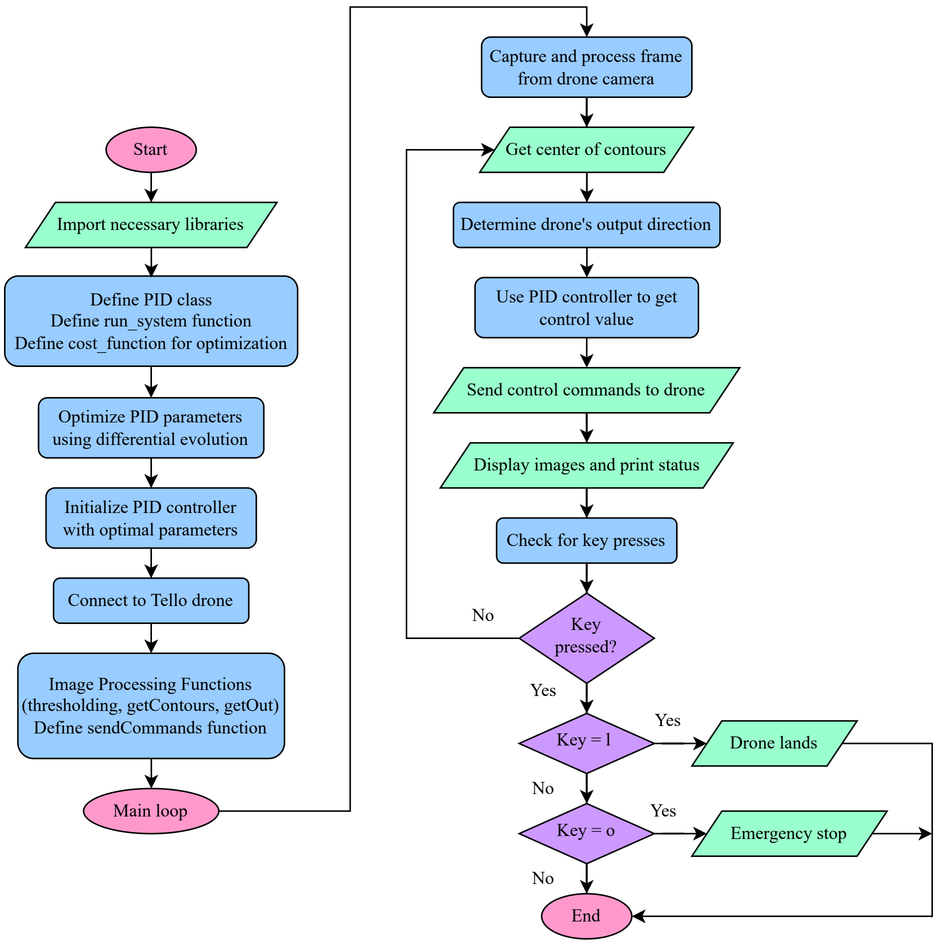


**Figure 3.** Flowchart of the PID controller implementation.

The design and optimization process involved several key steps:

1. Initialization: the FOPID controller is initialized with the PID parameters obtained from the auto-tune process and the chosen fractional orders.
2. Fractional calculus: fractional integral and derivative terms were computed using trail-and-error method and special functions from the 'scipy' library.
3. Control law: the control law combines proportional, fractional integral, and fractional derivative terms to compute the control action.

4. Implementation: the FOPID controller is utilized on the quadrotor using the 'djitellopy' library for drone control. The controller adjusts the drone's flight parameters in real-time to maintain stability.

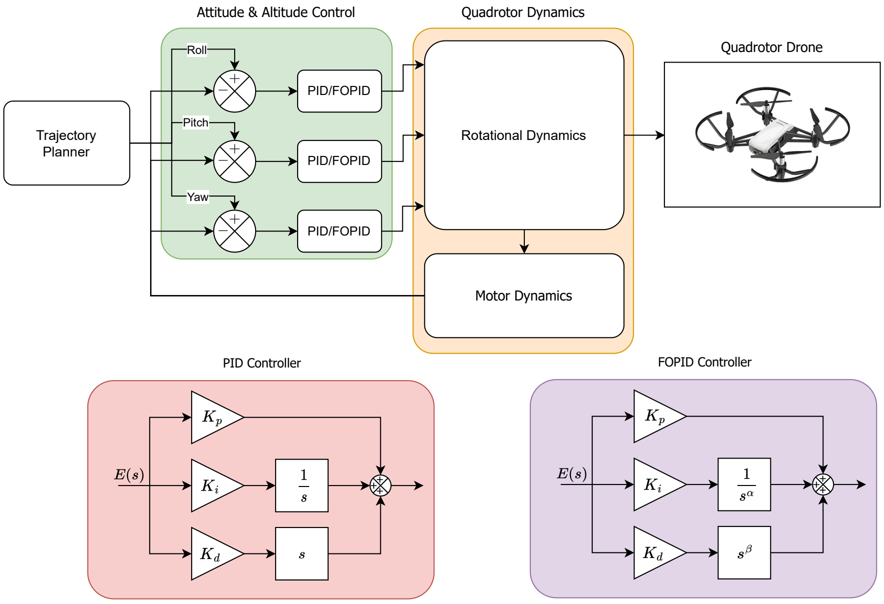


**Figure 4.** Block diagram of the PID and FOPID control implementation on the quadrotor.

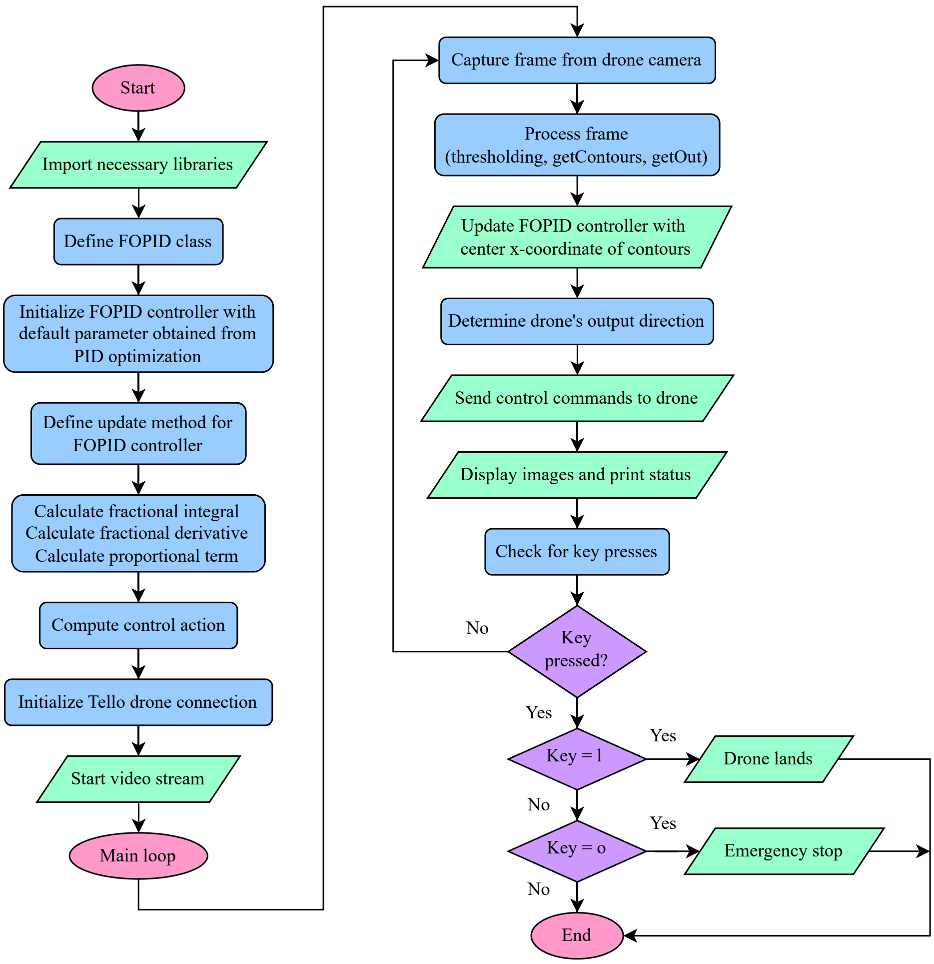


**Figure 5.** Flowchart of the FOPID controller implementation.

### *3.4. Implementation on the Quadrotor*

Implementing control algorithms on the quadrotor requires integrating both hardware and software systems, followed by extensive testing and validation to ensure the effectiveness of the control strategies. The control algorithms were initially embedded into the drone's onboard microcontroller, programmed to execute these algorithms and interface with the drone's motors. The sensors provide real-time data on the drone's orientation, acceleration, and other crucial flight parameters. Communication modules facilitated data transmission between the drone and the ground control station, allowing for remote monitoring and control. The software is developed using Python, libraries such as 'djitellopy' for drone communication and control, and 'scipy' for implementing fractional calculus in the FOPID controller. All of the Python codes developed for this research are available at https://github.com/KishoreBingi/Fractional-Order-Control-Algorithm-for-Tello-EDU-Quadrotor-Drone (accessed on 24 September 2024). The implementation structure of the PID and FOPID controllers on the drone is given as a block diagram in Figure 4. The series of test performances for the control algorithms is as follows:

- Preliminary flight tests: Initial testing is conducted indoors to ensure the algorithms stabilize the drone in a controlled environment. Various propeller failure scenarios shown in Figure 6 were simulated to evaluate the controller's ability to maintain stability and achieve safe landings.
- Line tracking tests: Subsequent testing is conducted under more challenging conditions. These tests were conducted using lightweight masking tape material (see Figure 2a) further evaluated the controllers' robustness and responsiveness to visual environmental disturbances.

Data collected from the sensors and controller during these tests is logged for further analysis. Performance metrics, including stability, response time, and robustness, were collected to compare the effectiveness of the PID and FOPID controllers. The results were then analysed to determine which controller demonstrated superior performance in maintaining drone stability and responding to propeller failures. The following steps outline the implementation of the FOPID controller on the drone:

1. Initialization:
   - Initialize the drone and set up communication using 'djitellopy' [41,42].
   - Initialize the FOPID controller with the optimal parameters.
2. Flight control loop:
   - Continuously capture the drone's flight parameters (e.g., roll, pitch, yaw).
   - Use the FOPID controller to compute the necessary control actions based on the current flight parameters.
   - Adjust the drone's real-time control inputs (e.g., roll, pitch, yaw) to maintain stability.
3. Safety and monitoring:
   - Implement safety checks to ensure the drone remains within operational limits.
   - Monitor the drone's battery level and other critical parameters like all four motor temperatures to prevent potential issues during flight.

## 4. Results and Discussions

The following section outlines the experimental design scenarios for the various test cases conducted during the flight. It then presents the validation of the drone's indoor performance. The propeller response in line tracking under disturbed and undisturbed conditions is analysed, and a comparative analysis follows.

### *4.1. Experimental Design and Cases*

The primary objective of the experimental design is to assess the effectiveness of the PID and FOPID controllers in maintaining drone stability and achieving safe landings during propeller disturbances. Four distinct experimental cases, each representing a different disturbance scenario, were created and maintained until the test completion to comprehensively evaluate the controllers' performance. For effective comparison, the same lightweight masking tape material (see Figure 2a) has been used in all four experiments, as shown in Figure 6. Lastly, all the case performance is comparatively analysed to validate the proposed FOPID's effectiveness in handling the external disturbance with smoother and more robust control actions.

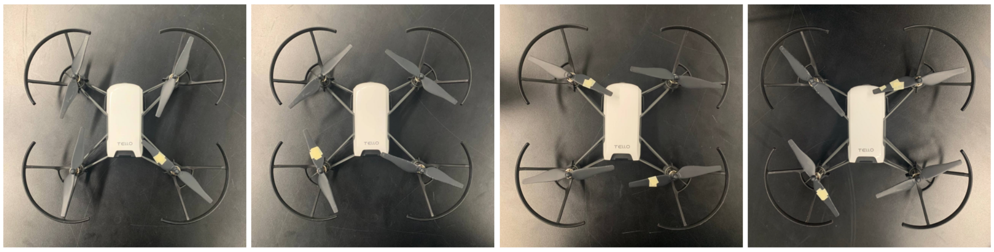

**Figure 6.** Propeller disturbance design for case 1, 2, 3,4.

1. Case 1: disturbance on one counter-clockwise propeller.
   - Details: This case involves creating a disturbance on one of the drone's counter-clockwise (CCW) propellers. The objective is to test the controller's ability to stabilize the drone when only one of the CCW propellers is affected.
   - Expected outcome: the controller should compensate for the disturbance and maintain a stable flight, achieving a safe landing.
2. Case 2: disturbance on one clockwise propeller.
   - Details: This case involves creating a disturbance on one of the drone's clockwise (CW) propellers. Likewise, in case 1, the objective is to assess the controller's ability to handle a single-propeller disturbance on a CW propeller.
   - Expected outcome: the controller should successfully counteract the disturbance, ensuring the drone remains stable and lands safely.
3. Case 3: disturbance on both counter-clockwise propellers.
   - Details: In this scenario, disturbances are introduced to both drones' CCW propellers. This case tests the controller's performance under more severe conditions, as both CCW propellers are affected simultaneously.
   - Expected outcome: the controller needs to show its robustness by stabilizing the drone, even when faced with the disturbance caused by the dual propellers, so that the drone can safely land.
4. Case 4: disturbance on both clockwise propellers.
   - Details: This case involves disturbances on both drones' CW propellers. As with case 3, this scenario presents a challenging condition where the controller must manage dual-propeller disturbances on the CW side.
   - Expected outcome: the controller is expected to mitigate the disturbances and maintain flight stability, culminating in a safe landing.

### *4.2. Line Tracking Under Disturbed and Undisturbed Conditions*

Comparing the drone's ability to maintain line tracking accuracy amidst varying levels of disruption, we can comprehensively understand the algorithm's robustness and adaptability to implement the PID and FOPID controllers on the drone. During indoor testing, case 2 compares its performance with a standard drone in a line-following test. As depicted in Figure 7, the drone maintains a consistent hover at the same altitude for stable and unstable testing once airborne. The assessment of drone stability is carried out during the line-following task. Subsequently, the stable drone successfully follows the line to its end without encountering any issues. In contrast, the unstable drone continues to hover erratically and rotates in an attempt to locate the line due to its instability. For additional details on the testing process, please refer to the video: https://youtu.be/M3OsUNe15DY (accessed on 15 August 2024).

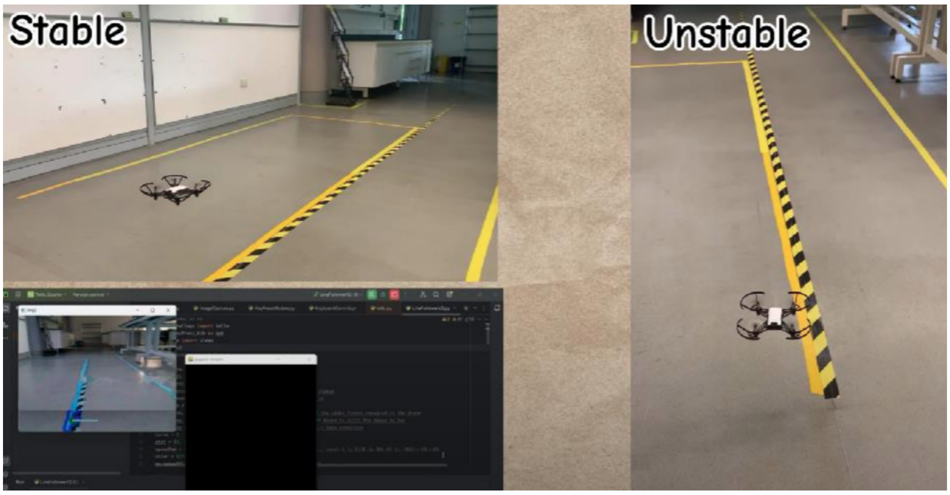


**Figure 7.** Drone's hovering and position a few seconds before beginning to follow the line.

### 4.3. Controller Performance

The performance of the PID and FOPID control algorithms is systematically assessed under four distinct disturbance conditions to determine their ability to uphold stability and control as shown in Figures 8 and 9. This evaluation examines how the controllers respond to disturbances affecting the quadrotor's propellers, shedding light on the algorithms' strengths and limitations. The controller parameters for the different test case conditions are given in Table 2. Also, the video recordings of the four cases can be viewed at the following links:

- Case 1: https://youtu.be/O75SCfhuQqI (accessed on 15 August 2024).
- Case 2: https://youtu.be/_K_Hev007IA (accessed on 15 August 2024).
- Case 3: https://youtu.be/Sxe4myqIjKs (accessed on 15 August 2024).
- Case 4: https://youtu.be/NRb0vmyMmgw (accessed on 15 August 2024).


(**a**)

**Figure 8.** *Cont.*


(**b**)

**Figure 8.** Performance of the PID and FOPID control algorithms in cases 1 and 2 (**a**,**b**).


(**a**)


(**b**)

**Figure 9.** Performance of the PID and FOPID control algorithms in cases 3 and 4 (**a**,**b**).

**Table 2.** PID and FOPID controller parameters for different test cases.

| Case | $K_p$ | $K_i$ | $K_d$ | $\alpha$ | $\beta$ |
|---|---|---|---|---|---|
| Case 1 | 5.852 | 4.268 | 4.462 | 0.98 | 0.02 |
| Case 2 | 1.872 | 9.746 | 2.257 | 0.98 | 0.02 |
| Case 3 | 4.243 | 8.066 | 0.832 | 0.98 | 0.02 |
| Case 4 | 7.326 | 1.287 | 4.198 | 0.98 | 0.02 |

#### 4.3.1. Case 1: Disturbance on One Counter-Clockwise Propeller

In case 1, when an unexpected disturbance affects one of the counter-clockwise propellers, the PID controller demonstrates significant roll, pitch, and yaw fluctuations, as shown in Figure 10.

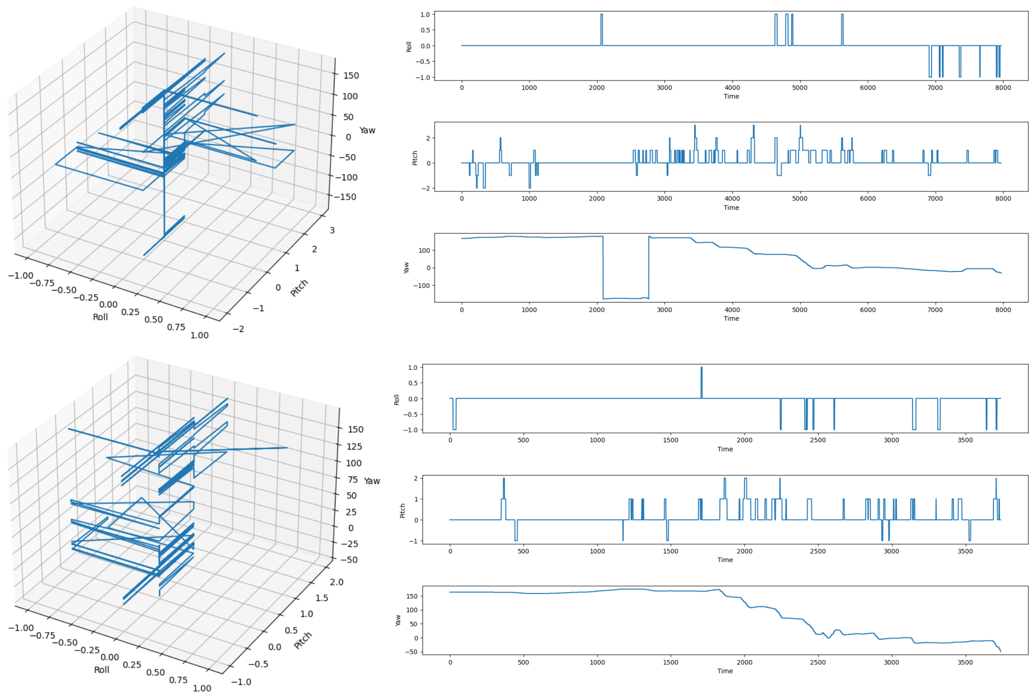


**Figure 10.** Performance plot of roll (deg), pitch (deg) and yaw (deg) vs. time stamp for PID (**top**) and FOPID (**bottom**) controllers with 3D trajectory and tracking performances in case 1.

The trajectory outcomes are inconsistent, indicating the PID controller's difficulty effectively mitigating the disturbance. In contrast, the FOPID controller exhibits remarkable stability and produces much smoother trajectories in roll, pitch, and yaw compared to the PID controller. Due to its capability to adapt control inputs, the FOPID algorithm promptly addresses the disturbance, resulting in minimal deviation from the intended flight path and showcasing superior disturbance rejection capabilities.

#### 4.3.2. Case 2: Disturbance on One Clockwise Propeller

In case 2, when there was a disturbance on one clockwise propeller, the PID controller exhibited significant spikes in control efforts, similar to case 1, resulting in trajectories that deviated from the desired path, as shown in Figure 11. The higher integral gain indicated an increased need for corrective actions due to the disturbance, leading to observed deviations in the flight trajectory. These significant deviations underscored the controller's limited ability to maintain stability and follow the intended trajectory under such disturbances, indicating potential instability in the drone's flight. On the other hand, the FOPID controller demonstrated steady and controlled responses, effectively minimizing deviations from the intended trajectory and maintaining stable flight throughout the testing.

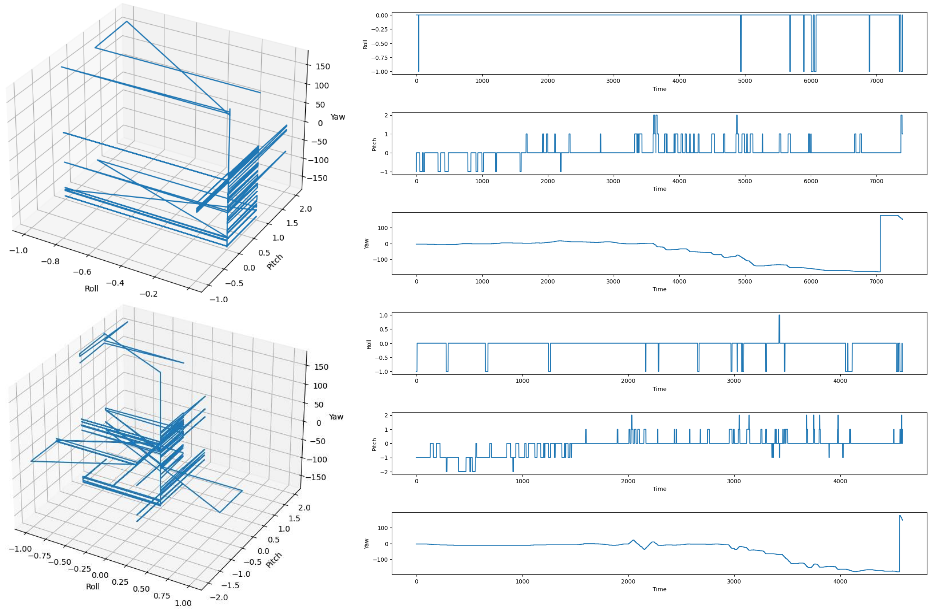


**Figure 11.** Performance plot of roll (deg), pitch (deg) and yaw (deg) vs. time stamp for PID (**top**) and FOPID (**bottom**) controllers with 3D trajectory and tracking performances in case 2.

#### 4.3.3. Case 3: Disturbance on Both Counter-Clockwise Propellers

In case 3, disturbances affect both counter-clockwise propellers simultaneously, and the PID controller is again affected by these disturbances, resulting in increased oscillations and less controlled behaviour under these conditions, as shown in Figure 12. The reduced derivative gain aims to moderate the response of roll, pitch, and yaw control; however, it results in larger oscillations, suggesting challenges in maintaining stability under dual disturbances. This situation reveals the PID controller's limitations in handling complex disturbance patterns, resulting in decreased stability and control. In the same way as the above two cases, the FOPID controller continues to excel, maintaining smooth and controlled flight paths with minimal oscillations. This capability is crucial for scenarios demanding precise manoeuvring and stability under adverse conditions to ensure consistent performance, enhancing reliability in complex disturbance scenarios.

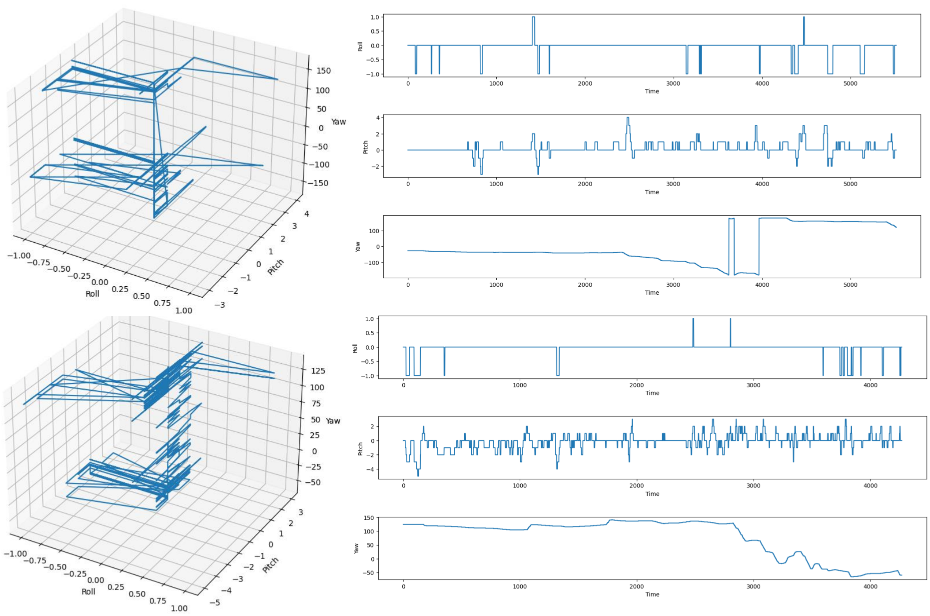


**Figure 12.** Performance plot of roll (deg), pitch (deg) and yaw (deg) vs. time stamp for PID (**top**) and FOPID (**bottom**) controllers with 3D trajectory and tracking performances in case 3.

#### 4.3.4. Case 4: Disturbance on Both Clockwise Propellers

In case 4, when disturbances affected both clockwise propellers, the PID controller showed noticeable roll, pitch, and yaw fluctuations, resulting in decreased stability, as shown in Figure 13. The higher proportional gain emphasized the controller's aggressive response to counteract unstable trajectory movements caused by dual disturbances. However, the observed instability indicated limitations in effectively stabilizing the drone under these conditions, suggesting that the PID controller was not well-suited for managing scenarios involving multiple propeller disturbances. On the other hand, the FOPID controller demonstrated resilience and stability by effectively mitigating the disturbances, leading to controlled trajectories and reduced unstable behaviour compared to conventional PID controllers.

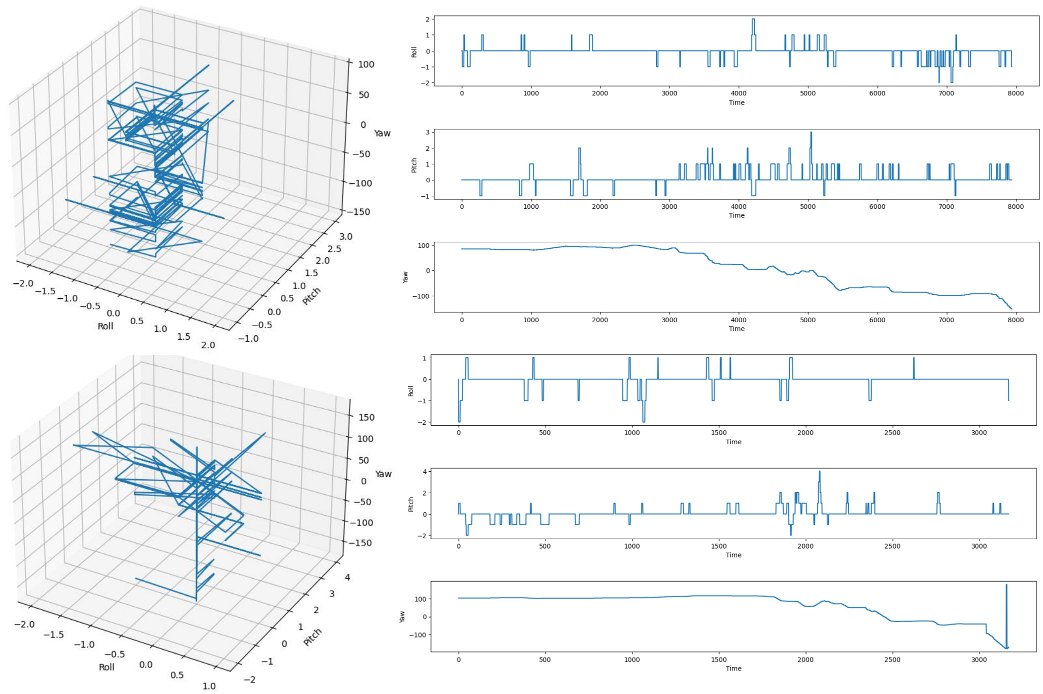


**Figure 13.** Performance plot of roll (deg), pitch (deg) and yaw (deg) vs. time stamp for PID (**top**) and FOPID (**bottom**) controllers with 3D trajectory and tracking performances in case 4.

### *4.4. Comparative Analysis*

In comparing PID and FOPID controllers under various disturbance scenarios throughout the testing, both controllers were assessed based on their capability to maintain stable flight paths and reject disturbances caused by propeller failures. The results demonstrated that the FOPID controller effectively mitigated disturbances and sustained the drone's stability, even under challenging conditions. In contrast, the PID controller consistently exhibited high yaw, pitch, and roll variations, suggesting instability and less effective disturbance rejection. In the first case, the PID controller displayed larger trajectory variations and took approximately 122 s to complete the test, as shown in Table 3. In contrast, the FOPID controller exhibited smoother trajectories and completed the test in 81 s.

In case 2, both controllers encountered challenges with trajectory deviations, while the FOPID controller demonstrated more stable responses over the 127-second test period compared to the PID controller's 130 s. In case 3, both controllers effectively managed the disturbances, but the PID controller displayed more oscillatory behaviour over 95 s, while the FOPID controller maintained smoother trajectories throughout the 92-second test. Lastly, in case 4, significant differences were observed in the PID controller performance, where it exhibited chaotic responses over 123 s, whereas the FOPID controller demonstrated superior control and stability within 73 s.

**Table 3.** Test durations and graph time ranges for PID and FOPID controllers.

| Case | Controller | Graph Time Range | Actual Test Duration (Min) | Actual Test Duration (s) |
|---|---|---|---|---|
| Case 1 | PID | 0–8000 units | 2.03 | 121.8 |
| | FOPID | 0–3500 units | 1.35 | 81 |
| Case 2 | PID | 0–8000 units | 2.17 | 130.2 |
| | FOPID | 0–4000 units | 2.11 | 126.6 |
| Case 3 | PID | 0–5000 units | 1.58 | 94.8 |
| | FOPID | 0–4000 units | 1.52 | 91.2 |
| Case 4 | PID | 0–8000 units | 2.05 | 123 |
| | FOPID | 0–3000 units | 1.22 | 73.2 |

While comparing the graphical results and actual test durations, it is clear that the FOPID controller consistently outperformed the PID controller in terms of stability and trajectory smoothness across all test cases. The discrepancies in time units emphasize the need for additional context when interpreting the real-time data. These findings highlight the advantages of using the FOPID controller over the traditional PID controller to enhance drone stability and response under dynamic conditions. The comparative performance of PID and FOPID controllers is summarized in Table 4 below, quantifying the variations in roll, pitch, and yaw values across different test cases. This comparison emphasizes the FOPID controller's consistent stability and disturbance rejection advantage over the traditional PID controller.

**Table 4.** Comparative performance analysis of PID and FOPID controllers.

| Case | Controller | Roll | Pitch | Yaw | Trajectory Stability |
|---|---|---|---|---|---|
| Case 1 | PID | High | High | High | Unstable |
| | FOPID | Low | Low | Low | Smooth |
| Case 2 | PID | High | High | High | Deviated |
| | FOPID | Low | Low | Low | Steady |
| Case 3 | PID | High | High | High | Oscillatory |
| | FOPID | Low | Low | Low | Controlled |
| Case 4 | PID | High | High | High | Chaotic |
| | FOPID | Low | Low | Low | Controlled |

## 5. Conclusions

This paper's fractional-order control algorithm represents a significant leap forward in quadrotor drone technology, providing improved safety and resilience during propeller failures. Through extensive experimentation and testing in various indoor conditions, including line tracking under disturbed and undisturbed environments, valuable insights were gained into the algorithm's performance under simulated failure scenarios. Despite facing challenges such as tape distortion and excess weight, the algorithm displayed promising adaptability and resilience, laying a solid groundwork for further refinement and optimization. Similarly, the indoor drone test created controlled conditions to evaluate the algorithm's response to simulated propeller failures. However, disruptive factors like masking tape led to instability, highlighting the need for precise calibration to counteract destabilization while maintaining essential flight capabilities. These findings emphasize the complexities of integrating disruptive elements into the algorithm while ensuring optimal performance. In the future, incorporating adaptive control strategies, such as model predictive control or reinforcement learning, can enhance the algorithm's adaptability and robustness in dynamic flight environments. Further, integrating real-time sensor feedback into the control algorithm can bolster situational awareness and enable proactive responses to evolving flight conditions.

**Author Contributions:** Conceptualization, K.B. and R.K.; methodology, N.H.B.R. and P.A.M.D.; software, P.A.M.D. and G.K.; validation, K.B., R.K. and M.O.; formal analysis, N.H.B.R. and G.K.; investigation, N.H.B.R. and K.B.; resources, K.B., R.K. and M.O.; data curation, P.A.M.D. and B.R.P.; writing—original draft preparation, N.H.B.R. and P.A.M.D.; writing—review and editing, K.B., G.K. and M.O.; visualization, B.R.P. and M.O.; supervision, K.B. and R.K.; project administration, R.K., B.R.P. and M.O.; funding acquisition, K.B. and B.R.P. All authors have read and agreed to the published version of the manuscript.

**Funding:** The authors would like to acknowledge Universiti Teknologi PETRONAS, Malaysia, and Alliance University, India, for supporting this work under the International Collaborative Research Fund (ICRF), grant number 015ME0-379.

**Institutional Review Board Statement:** Not applicable.

**Data Availability Statement:** No new data were created or analysed in this study. Data sharing is not applicable to this article.

**Acknowledgments:** The authors would like to acknowledge the support from Universiti Teknologi PETRONAS and Alliance University for providing the research facilities.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Mohsan, S.A.H.; Khan, M.A.; Noor, F.; Ullah, I.; Alsharif, M.H. Towards the unmanned aerial vehicles (UAVs): A comprehensive review. *Drones* **2022**, *6*, 147. [CrossRef]
2. Laghari, A.A.; Jumani, A.K.; Laghari, R.A.; Nawaz, H. Unmanned aerial vehicles: A review. *Cogn. Robot.* **2023**, *3*, 8–22. [CrossRef]
3. Del Cerro, J.; Cruz Ulloa, C.; Barrientos, A.; de León Rivas, J. Unmanned aerial vehicles in agriculture: A survey. *Agronomy* **2021**, *11*, 203. [CrossRef]
4. Wandelt, S.; Wang, S.; Zheng, C.; Sun, X. Aerial: A meta review and discussion of challenges toward unmanned aerial vehicle operations in logistics, mobility, and monitoring. *IEEE Trans. Intell. Transp. Syst.* **2023**. [CrossRef]
5. Lopez-Sanchez, I.; Moreno-Valenzuela, J. PID control of quadrotor UAVs: A survey. *Annu. Rev. Control* **2023**, *56*, 100900. [CrossRef]
6. Zhou, X.; Yu, X.; Guo, K.; Zhou, S.; Guo, L.; Zhang, Y.; Peng, X. Safety flight control design of a quadrotor UAV with capability analysis. *IEEE Trans. Cybern.* **2021**, *53*, 1738–1751. [CrossRef]
7. Carlos, B.B.; Franchi, A.; Oriolo, G. Towards Safe Human-Quadrotor Interaction: Mixed-Initiative Control via Real-Time NMPC. *IEEE Robot. Autom. Lett.* **2021**, *6*, 7611–7618. [CrossRef]
8. Wang, B.; Shen, Y.; Zhang, Y. Active fault-tolerant control for a quadrotor helicopter against actuator faults and model uncertainties. *Aerosp. Sci. Technol.* **2020**, *99*, 105745. [CrossRef]
9. Guo, K.; Zhang, W.; Zhu, Y.; Jia, J.; Yu, X.; Zhang, Y. Safety control for quadrotor UAV against ground effect and blade damage. *IEEE Trans. Ind. Electron.* **2022**, *69*, 13373–13383. [CrossRef]
10. Ullah, S.; Alghamdi, H.; Algethami, A.A.; Alghamdi, B.; Hafeez, G. Robust Control Design of Under-Actuated Nonlinear Systems: Quadcopter Unmanned Aerial Vehicles with Integral Backstepping Integral Terminal Fractional-Order Sliding Mode. *Fractal Fract.* **2024**, *8*, 412. [CrossRef]
11. Liu, H.; Li, B.; Xiao, B.; Ran, D.; Zhang, C. Reinforcement learning-based tracking control for a quadrotor unmanned aerial vehicle under external disturbances. *Int. J. Robust Nonlinear Control* **2023**, *33*, 10360–10377. [CrossRef]
12. Bektash, O.; Pedersen, J.N.; Gomez, A.R.; la Cour-Harbo, A. Automated emergency landing system for drones: SafeEYE project. In Proceedings of the 2020 International Conference on Unmanned Aircraft Systems (ICUAS), Athens, Greece, 1–4 September 2020; pp. 1056–1064.
13. Jung, W.; Bang, H. Fault and failure tolerant model predictive control of quadrotor UAV. *Int. J. Aeronaut. Space Sci.* **2021**, *22*, 663–675. [CrossRef]
14. Lee, S.J.; Jang, I.; Kim, H.J. Fail-safe flight of a fully-actuated quadrotor in a single motor failure. *IEEE Robot. Autom. Lett.* **2020**, *5*, 6403–6410. [CrossRef]
15. Vong, C.H.; Ryan, K.; Chung, H. Trajectory tracking control of quadcopters under tunnel effects. *Mechatronics* **2021**, *78*, 102628. [CrossRef]

16. Lin, J.; Wang, Y.; Miao, Z.; Zhong, H.; Fierro, R. Low-complexity control for vision-based landing of quadrotor UAV on unknown moving platform. *IEEE Trans. Ind. Inform.* **2021**, *18*, 5348–5358. [CrossRef]
17. Kumar, A. Real-time performance comparison of vision-based autonomous landing of quadcopter on a ground moving target. *IETE J. Res.* **2023**, *69*, 5455–5472. [CrossRef]
18. Sonugür, G. A Review of quadrotor UAV: Control and SLAM methodologies ranging from conventional to innovative approaches. *Robot. Auton. Syst.* **2023**, *161*, 104342. [CrossRef]
19. Hedayatpour, M.; Mehrandezh, M.; Janabi-Sharifi, F. Precision modeling and optimally-safe design of quadcopters for controlled crash landing in case of rotor failure. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 5206–5211.
20. Yoo, J.; Jang, D.; Kim, H.J.; Johansson, K.H. Hybrid reinforcement learning control for a micro quadrotor flight. *IEEE Control Syst. Lett.* **2020**, *5*, 505–510. [CrossRef]
21. Zhou, L.; Pljonkin, A.; Singh, P.K. Modeling and PID control of quadrotor UAV based on machine learning. *J. Intell. Syst.* **2022**, *31*, 1112–1122. [CrossRef]
22. Espinoza, J.; Hakim, N.; Tan, D.; Wilson, T.; Bingi, K.; Khan, E.; Masrura, S. Fractional-Order PID Control of Quadrotor Drone. In Proceedings of the 2023 Innovations in Power and Advanced Computing Technologies (i-PACT), Kuala Lumpur, Malaysia, 8–10 December 2023; pp. 1–6.
23. Labbadi, M.; Boukal, Y.; Cherkaoui, M.; Djemai, M. Fractional-order global sliding mode controller for an uncertain quadrotor UAVs subjected to external disturbances. *J. Frankl. Inst.* **2021**, *358*, 4822–4847. [CrossRef]
24. Pouzesh, M.; Mobayen, S. Event-triggered fractional-order sliding mode control technique for stabilization of disturbed quadrotor unmanned aerial vehicles. *Aerosp. Sci. Technol.* **2022**, *121*, 107337. [CrossRef]
25. Timis, D.D.; Muresan, C.I.; Dulf, E.H. Design and experimental results of an adaptive fractional-order controller for a quadrotor. *Fractal Fract.* **2022**, *6*, 204. [CrossRef]
26. Saif, A.W.A.; Gaufan, K.B.; El-Ferik, S.; Al-Dhaifallah, M. Fractional order sliding mode control of quadrotor based on fractional order model. *IEEE Access* **2023**, *11*, 79823–79837. [CrossRef]
27. Setyawan, G.E.; Kurniawan, W.; Gaol, A.C.L. Linear quadratic regulator controller (LQR) for AR. Drone's safe landing. In Proceedings of the 2019 International Conference on Sustainable Information Engineering and Technology (SIET), Lombok, Indonesia, 28–30 September 2019; pp. 228–233.
28. Lee, S.; Kwon, Y. Safe landing of drone using AI-based obstacle avoidance. *Int. J. Mech. Eng. Robot. Res.* **2020**, *9*, 1495–1501. [CrossRef]
29. Shi, G.; Shi, X.; O'Connell, M.; Yu, R.; Azizzadenesheli, K.; Anandkumar, A.; Yue, Y.; Chung, S.J. Neural lander: Stable drone landing control using learned dynamics. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 9784–9790.
30. Altınuç, K.O.; Khan, M.U.; Iqbal, J. Avoiding contingent incidents by quadrotors due to one or two propellers failure. *PLoS ONE* **2023**, *18*, e0282055. [CrossRef] [PubMed]
31. Zhou, Z.; Liu, Y. A smart landing platform with data-driven analytic procedures for UAV preflight safety diagnosis. *IEEE Access* **2021**, *9*, 154876–154891. [CrossRef]
32. Arasanipalai, R.; Agrawal, A.; Ghose, D. Mid-flight propeller failure detection and control of propeller-deficient quadcopter using reinforcement learning. *arXiv* **2020**, arXiv:2002.11564.
33. Liu, H.; Man, M.H.C.; Low, K.H. UAV airborne collision to manned aircraft engine: Damage of fan blades and resultant thrust loss. *Aerosp. Sci. Technol.* **2021**, *113*, 106645. [CrossRef]
34. Merheb, A.R.; Noura, H.; Bateman, F. Emergency control of AR drone quadrotor UAV suffering a total loss of one rotor. *IEEE/ASME Trans. Mechatron.* **2017**, *22*, 961–971. [CrossRef]
35. Toufaili, H.E. *Quadcopter Collision Recovery under Loss of One Propeller*; McGill University (Canada): Montréal, QC, Canada, 2019.
36. Iskandar, M.; Bingi, K.; Prusty, B.R.; Omar, M.; Ibrahim, R. Artificial intelligence-based human gesture tracking control techniques of Tello EDU Quadrotor Drone. In Proceedings of the International Conference on Green Energy, Computing and Intelligent Technology (GEn-CITy 2023), Iskandar Puteri, Malaysia, 10–12 July 2023; Volume 2023, pp. 123–128.
37. Iskandar, M.; Bingi, K.; Ibrahim, R.; Omar, M.; Devan, P.A.M. Hybrid Face and Eye Gesture Tracking Algorithm for Tello EDU RoboMaster TT Quadrotor Drone. In Proceedings of the 2023 Innovations in Power and Advanced Computing Technologies (i-PACT), Kuala Lumpur, Malaysia, 8–10 December 2023; pp. 1–6.
38. Kasraoui, A.; Bingi, K.; Ibrahim, R.; Omar, M.; Devan, P.A.M.; Prusty, B.R. Tracking Control of Tello EDU Quadrotor Drone Using Image Thresholding. In Proceedings of the 2023 Innovations in Power and Advanced Computing Technologies (i-PACT), Kuala Lumpur, Malaysia, 8–10 December 2023; pp. 1–6.
39. Bingi, K.; Ibrahim, R.; Karsiti, M.N.; Hassan, S.M.; Harindran, V.R. *Fractional-Order Systems and PID Controllers*; Springer: Berlin/Heidelberg, Germany, 2020; Volume 264.
40. Bingi, K.; Rajanarayan Prusty, B.; Pal Singh, A. A review on fractional-order modelling and control of robotic manipulators. *Fractal Fract.* **2023**, *7*, 77. [CrossRef]

41. Rodriguez, A.A.; Shekaramiz, M.; Masoum, M.A. Computer Vision-Based Path Planning with Indoor Low-Cost Autonomous Drones: An Educational Surrogate Project for Autonomous Wind Farm Navigation. *Drones* **2024**, *8*, 154. [CrossRef]
42. Lionel, N.; Bingi, K.; Ibrahim, R.; Korah, R.; Kumar, G.; Prusty, B.R. Autonomous Inspection of Solar Panels and Wind Turbines Using YOLOv8 with Quadrotor Drones. In Proceedings of the 2024 9th International Conference on Mechatronics Engineering (ICOM), Kuala Lumpur, Malaysia, 13–14 August 2024; pp. 322–326.