aerospace

Article

# Real-Time Implementation of an Adaptive PID Controller for the Quadrotor MAV Embedded Flight Control System

Aminurrashid Noordin [1,2], Mohd Ariffanan Mohd Basri [1,*] and Zaharuddin Mohamed [1]

1 Faculty of Electrical Engineering, Universiti Teknologi Malaysia (UTM), Johor Bahru 81310, Johor, Malaysia
2 Faculty of Electrical and Electronic Engineering Technology, Universiti Teknikal Malaysia Melaka, Hang Tuah Jaya, Durian Tunggal 76100, Melaka, Malaysia
* Correspondence: ariffanan@fke.utm.my; Tel.: +607-553-5209

**Abstract:** This paper presents the real-time implementation of an altitude-embedded flight controller using proportional, integral, and derivative (PID) control, adaptive PID (APID) control, and adaptive PID control with a fuzzy compensator (APIDFC) for a micro air vehicle (MAV), specifically, for a Parrot Mambo Minidrone. In order to obtain robustness against disturbance, the adaptive mechanism, which was centered on the second-order sliding mode control, was applied to tune the classical parameters of the PID controller of the altitude controller. Additionally, a fuzzy compensator was introduced to diminish the existence of the chattering phenomena triggered by the application of the sliding mode control. Four simulation and experimental scenarios were conducted, which included hovering, as well as sine, square, and trapezium tracking. Moreover, the controller's resilience was tested at 1.1 m above the ground by adding a mass of about 12.5 g, 15 s after the flight launch. The results demonstrated that all controllers were able to follow the reference altitude, with some spike or overshoot. Although there were slight overshoots in the control effort, the fuzzy compensator reduced the chattering phenomenon by about 6%. Moreover, it was found that in the experiment, the APID and APIDFC controllers consumed 2% and 4% less power, respectively, when compared to the PID controller used to hover the MAV.

**Keywords:** adaptive control; sliding mode control; fuzzy compensator; altitude control; micro air vehicle; quadrotor; pid; external disturbance

**Citation:** Noordin, A.; Mohd Basri, M.A.; Mohamed, Z. Real-Time Implementation of an Adaptive PID Controller for the Quadrotor MAV Embedded Flight Control System. *Aerospace* **2023**, *10*, 59. https://doi.org/10.3390/aerospace10010059

Academic Editor: Bosko Rasou

Received: 7 December 2022
Revised: 29 December 2022
Accepted: 4 January 2023
Published: 6 January 2023

## 1. Introduction

Unmanned Aerial Vehicle (UAV) control is a very challenging area of research, particularly for vertical take-off and landing (VTOL) of aircraft such as the quadrotor. A quadrotor is an unmanned aerial VTOL vehicle that belongs to the category of aerial vehicles with rotating wings. Because of its simple structure, quadrotors are a common design for small UAVs. This aerial aircraft features four motors that transmit electricity to propellers to produce propulsion thrust. By adjusting the engine rpm, this vehicle may be controlled and stabilized.

In the last few years, the state-of-the-art for quadrotor control has seen a significant development. All studies begin with the quadrotor's fundamental dynamical model, although more complicated aerodynamic features have also been incorporated [1,2]. As a small-scale UAV, the minidrone, nano, or micro air vehicle (MAV) has attracted the attention of educators and researchers due to its dependability and safety in restricted GPS spaces, such as hallways, schools, and other indoor environments. Thus, various control strategies, such as PID controllers, LQR controllers, sliding mode controllers, intelligent controllers, adaptive controllers, and others, have been studied. For instance, numerous works deal with the modeling and control of the Crazyflie [3–6]. DJI and Parrot are two of the most well-known manufacturers who have established and commercialized quadrotor drones which cover most of the types used for educational purposes. For instance, the company created the Tello EDU, which supports iOS users, and Scratch 2.0, SDK 2.0, and Swift Playgrounds for programming, which was been used in the studies in [7–9]. Parrot recently created the AR Drone 2.0, the Rolling Spider, and the Mambo, all of which allow for the models to be programmed using MATLAB/Simulink. Researchers have used the AR Drone reported in [10–13], whereas other researchers have used the Rolling Spider/Parrot Minidrones, supported by MATLAB's Simulink, as reported in [14–17].

Due to the advancement of technology, quadrotors can be utilized in performing various high risk, high intensity tasks which require a high degree of efficiency, such as military operations, agricultural applications, and express delivery [18]. These applications exhibit the inescapable issue that if the load is unknown or fluctuating, the quadrotor's altitude control will be unstable, making it impossible to maintain stable high-altitude flight or hover in the air. To maintain the stability of the altitude and track the planned trajectory, it is crucial to build a strong altitude control system. The linear control technique is an effort at control, but its effectiveness is limited, and it depends on a number of factors. Additionally, the linear controller is unable to handle the situations of drift and hover.

Due to its straightforward control architecture and effective control performance, PID is the most common flight control algorithm used in multirotor systems. PID is therefore still the favored approach for actual control in a quadrotor UAV [17,19]. However, troubleshooting the PID controller's settings is challenging, mainly relying on the expertise of researchers. In its application to autonomous load carrying and transport, the hovering vehicles must remain stable and balanced in flight as payload mass is added to the vehicle. Therefore, [20] investigates the impact of dynamic load disturbances caused by suddenly increasing the payload mass on a quadrotor operating under PID control. As a result, under PID control, a quadrotor will successfully reject additional load offset to a relative amount within a sizable range and allowable load position. Therefore, the control input that has a wide floating range and a slow convergence rate is deemed to be unstable.

Estimating the mass is another crucial step in altitude control. Altitude control is quite challenging if the quadrotor mass cannot be precisely approximated, since we cannot measure it directly while it is in the air. In reference [21], a mass estimation technique was suggested for estimating the inertial properties of the item being held in order to modify the controller and enhance performance while in flight. The authors of [22] suggested an adaptive robust control (ARC) to account for the parametric uncertainty for unknown payloads. However, the parameter estimate could not get close to its real value when the required output remained constant.

To track the position of a quadrotor, a linear quadratic regulator (LQR) is examined in [23–25]. A sliding mode control (SMC) strategy is created for a class of underactuated systems in [26,27], used to stabilize the position and attitude of a quadrotor UAV. According to simulation and experiment, the second order SMC may significantly reduce chattering when compared to regular SMC. A model free-based single dimension fuzzy SMC technique is created in [28] to enable the finite-time control of a quadrotor's translational and rotational movements, despite perturbations. According to numerical simulation on the altitude, the comprehensive fuzzy adaptive sliding mode control in [29] is able to handle the control problems of the highly coupled nonlinear fixed-wing UAV model to address the chattering problem of classic sliding mode control.

The hybrid backstepping-based control techniques also show good performance to cater to the quadrotor trajectory tracking problem. In order to exhibit the efficacy and practicability of the suggested control approach, simulation results under several scenarios, such as parameter uncertainties under Gaussian random disturbances, constant and time-varying external disturbances are provided, as shown in [30]. In [31], a robust backstepping super-twisting sliding mode control strategy was tested on the X450 quadrotor, emphasize the improved proficiencies of the proposed control approach in terms of tracking accuracy and chattering occurrence reduction.

In a previous study, [32] implemented the control scheme and simulated it on UAV, while [33] successfully used FPGA to apply the control scheme on a DC motor. This paper focuses on the impact of the real-time implementation of PID, adaptive PID (APID), and adaptive PID with a fuzzy compensator (APIDFC) for an altitude controller on the embedded flight system of a quadrotor MAV which has a mass less than 0.1kg. Using the sliding type control as the adaptive mechanism, this technique can suppress the manual controller's re-tuning gains in the classical PID controller. Furthermore, the effect of chattering by the sliding manifolds is removed by a fuzzy compensator. In order to display the benefits of applying the newly suggested flight control system, MATLAB software simulation and real-time hardware implementation were performed by redesigning the Flight Control System in the Simulink Support Package for the Parrot Mambo Minidrone. In conclusion, the APID and APIDFC control scheme can reduce energy usage and robust perturbation throughout, compared to PID controller.

## 2. MAV Quadrotor Modeling

The Parrot Mambo Minidrone is a quadrotor, which falls under the category of a micro aerial vehicle (MAV). This category is actually comprised of four symmetrically arranged rotors and, as displayed in Figure 1, is independently fixed on a rigid fuselage. Rotor 1 and rotor 3 rotate in a clockwise direction, whereas rotor 2 and rotor 4 rotate in a counterclockwise direction.

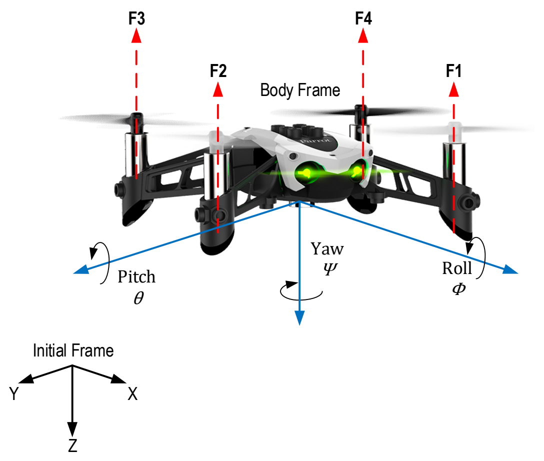

**Figure 1.** Parrot Mambo minidrone MAV.

The mathematical model of the aforementioned MAV is formed based on 6 degrees of freedom (DOF) $[x, y, z, \phi, \theta, \psi]^T$, which is well accepted by the researchers of such systems. Hence, this particular model can be categorized by two coordinate subsystems. Vector $[x, y, z]^T$ defines the absolute positions, while vector $[\phi, \theta, \psi]^T$ designates the orientation.

According to the general Newton–Euler formulation, quadrotor translational dynamics can be defined as:

$$\begin{aligned} m\ddot{x} &= -U_1(Sin(\psi)Sin(\phi) + Cos(\psi)Sin(\theta)Cos(\phi)) \\ m\ddot{y} &= -U_1(-Cos(\psi)Sin(\phi) + Sin(\psi)Sin(\theta)Cos(\phi)) \\ m\ddot{z} &= mg - U_1(Cos(\theta)Cos(\phi)) \end{aligned} \quad (1)$$

where $m$ represent the mass, $g$ denotes the gravitational coefficient, and $U_1$ symbolize the total thrust force. Thus, the rotational dynamics are termed as:

$$\begin{aligned} I_{xx}\ddot{\phi} &= \left(I_{yy} - I_{zz}\right)\dot{\psi}\dot{\theta} - (J_r\Omega_d)\dot{\theta} + lU_2 \\ I_{yy}\ddot{\theta} &= (I_{zz} - I_{xx})\dot{\psi}\dot{\phi} + (J_r\Omega_d)\dot{\phi} + lU_3 \\ I_{zz}\ddot{\psi} &= \left(I_{xx} - I_{yy}\right)\dot{\theta}\dot{\phi} + U_4 \end{aligned} \quad (2)$$

where $U_2$, $U_3$, and $U_4$ represent the total torque for the roll, pitch, and yaw, respectively. $l$, on the other hand, is the lateral arm length. Meanwhile, $I_{xx}$, $I_{yy}$ and $I_{zz}$ are the moments of inertia for the quadrotor. Whereas $J_r$ is the rotor inertia, and $\Omega_\mathrm{d}$ denotes the total rotor speed generated from the torque related to the control inputs as:

$$\begin{aligned} u_1 &= b\left(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2\right) \\ u_2 &= b\left(\Omega_1^2 - \Omega_2^2 - \Omega_3^2 + \Omega_4^2\right) \\ u_3 &= b\left(\Omega_1^2 + \Omega_2^2 - \Omega_3^2 - \Omega_4^2\right) \\ u_4 &= d\left(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2\right) \\ \Omega_d &= -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \end{aligned} \tag{3}$$

where $\Omega_i(i = 1,2,3,4)$ represents the rotor speed. On the other hand, $b$ and $d$ are the thrust and drag coefficients, respectively. To summarize, the Parrot Mambo Minidrone parameters are shown in Table 1.

**Table 1.** Parrot Mambo Model Physical Parameters [17].

| **Specification** | **Parameter** | **Unit** | **Value** |
|---|---|---|---|
| Quadrotor mass | $m$ | kg | 0.0630 |
| Lateral moment arm | $l$ | m | 0.0624 |
| Thrust coefficient | $b$ | Ns$^2$ | 0.0107 |
| Drag coefficient | $d$ | Nms$^2$ | $0.7826400 \times 10^{-3}$ |
| Rolling moment of inertia | $I_{xx}$ | kgm$^2$ | $0.0582857 \times 10^{-3}$ |
| Pitching moment of inertia | $I_{yy}$ | kgm$^2$ | $0.0716914 \times 10^{-3}$ |
| Yawing moment of inertia | $I_{zz}$ | kgm$^2$ | $0.1000000 \times 10^{-3}$ |
| Rotor moment of inertia | $J_r$ | kgm$^2$ | $0.1021 \times 10^{-6}$ |

Translational and rotational dynamics, as shown in (1) and (2), can be expressed as second-order state-space equations:

$$\ddot{x} = f(x) + g(x)u \tag{4}$$

In order to create an orderly control system design method and at the same time, maintain simple notation, the state vector becomes:

$$x = [\, x_1\ x_2\ x_3\ x_4\ x_5\ x_6]^T = [x\ y\ z\ \phi\ \theta\ \psi]^T \tag{5}$$

and the input vector $u = [u_1\ u_2\ u_3\ u_4]^T$. The non-linear function $f(x)$ and $g(x)$ can be rewritten as:

$$f(x) = \begin{bmatrix} f_1(x) \\ f_2(x) \\ f_3(x) \\ f_4(x) \\ f_5(x) \\ f_6(x) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ g \\ a_1\dot{\psi}\dot{\theta} - a_2\Omega_d\dot{\theta} \\ a_3\dot{\psi}\dot{\phi} + a_4\Omega_d\dot{\phi} \\ a_5\dot{\theta}\dot{\phi} \end{bmatrix} \tag{6}$$

$$g(x) = \begin{bmatrix} g_1(x) & 0 & 0 & 0 \\ g_2(x) & 0 & 0 & 0 \\ g_3(x) & 0 & 0 & 0 \\ 0 & g_4(x) & 0 & 0 \\ 0 & 0 & g_5(x) & 0 \\ 0 & 0 & 0 & g_6(x) \end{bmatrix} = \begin{bmatrix} -\frac{1}{m}u_x & 0 & 0 & 0 \\ -\frac{1}{m}u_y & 0 & 0 & 0 \\ -\frac{1}{m}u_z & 0 & 0 & 0 \\ 0 & b_1 & 0 & 0 \\ 0 & 0 & b_2 & 0 \\ 0 & 0 & 0 & b_3 \end{bmatrix} \tag{7}$$

where

$$\begin{aligned} a_1 &= \frac{I_{yy}-I_{zz}}{I_{xx}},\ a_2 = \frac{J_r}{I_{xx}},\ a_3 = \frac{I_{zz}-I_{xx}}{I_{yy}} \\ a_4 &= \frac{J_r}{I_{yy}},\ a_5 = \frac{I_{xx}-I_{yy}}{I_{zz}},\ b_1 = \frac{l}{I_{xx}} \\ b_2 &= \frac{l}{I_{yy}},\ b_3 = \frac{1}{I_{zz}} \end{aligned} \tag{8}$$

and

$$\begin{aligned} u_x &= S\psi S\phi + C\psi S\theta C\phi \\ u_y &= -C\psi S\phi + S\psi S\theta C\phi \\ u_z &= C\theta C\phi \end{aligned} \tag{9}$$

## 3. Flight Controller Design

Four controllers were designed to control the minidrone, which consist of the attitude controller, yaw controller, position controller, and altitude controller, as shown in Figure 2. The control mixer is an inverse of (3) to convert the rotation speed to force. Thereafter, a brief description of the controller's design is mentioned, but in this work, we concentrate on the altitude controller.

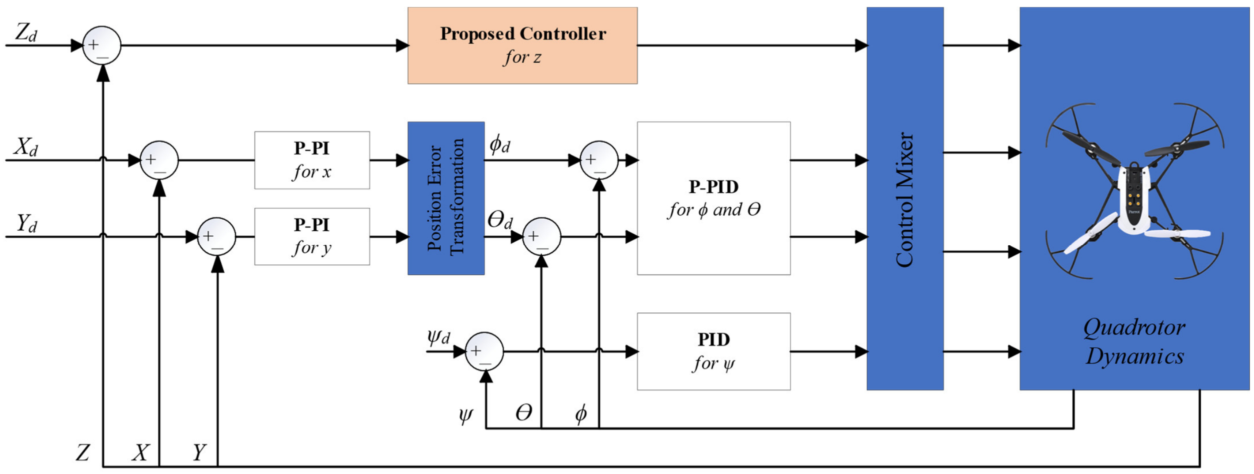

**Figure 2.** Block diagram of the embedded system flight controller.

### 3.1. Attitude and Yaw Controller

The attitude controller is designed using cascaded PID, where the proportional loop controls the pitch and roll angle, while the PID controls the angular velocity as:

$$U_i = k_P^i e_i + k_I^i \int e_i + k_D^i \dot{e}_i \ (i = \phi, \theta, \psi) \tag{10}$$

where $e_i = i_d - i$ and $\dot{e}_i = \dot{i}_d - \dot{i}$ are the error and derivate error between the desired signal and the actual signal, and $k_P^i$, $k_I^i$, and $k_D^i$ are the PID gains parameters $(i = \phi, \theta, \psi)$. The yaw controller, on the other hand, used classical PID (10) as well.

### 3.2. Position Controller

The position controller is designed using cascade PID, where the proportional loop controls the $x$-$y$, while the PI controls the angular velocity as:

$$U_i = k_P^i e_i + k_I^i \int e_i dt \ (i = x, y) \tag{11}$$

In addition, a position error transformation is introduced for error body ($e^b$) to error earth ($e^e$) conversion as:

$$\begin{bmatrix} e_x^b \\ e_y^b \end{bmatrix} = \begin{bmatrix} \cos(\psi) & \sin(\psi) \\ -\sin(\psi) & \cos(\psi) \end{bmatrix} \begin{bmatrix} e_x^e \\ e_y^e \end{bmatrix}$$

### *3.3. Altitude Control Design*

Figure 3 illustrates the block diagram of an adaptive PID scheme applied to the altitude control systems. In this control system, a PID controller and a fuzzy logic compensator are combined in the suggested method as:

$$u_1 = u_{pid} + u_{fc} \tag{12}$$

where $u_{pid}$ is the PID controller implemented to approach the optimum controller, $u^*$, and $u_{fc}$ is the designated fuzzy logic compensator applied to decrease the approximation error left between both controllers.

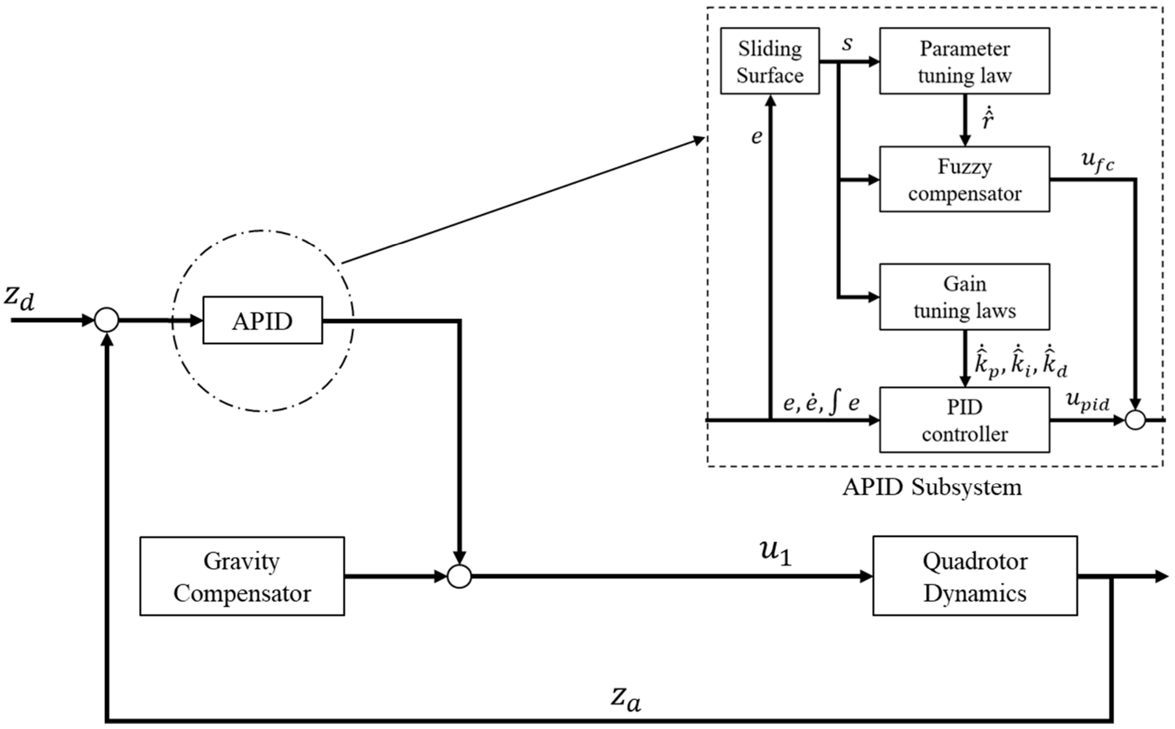

**Figure 3.** Adaptive PID controller for an altitude controller block diagram.

The aim of the system is to determine a certain control law so that $x$ can track the coveted $x_d$ as close as possible. Hence, the tracking error can be defined as:

$$e = x_d - x \tag{13}$$

Presuming that the optimal controller is created when all the parameters in (4) are identified, this can be expressed as:

$$u^* = g^{-1}\left(-f + \ddot{x}_d + k_1\dot{e} + k_2 e\right) \tag{14}$$

where $k_1$ and $k_2$ are chosen as the non-zero positive constants to adhere to the Hurwitz criterion, indicating $\lim\limits_{t\to\infty} e = 0$ for any initial starting conditions. Substituting (14) into (4) then yields:

$$\ddot{e} + k_1\dot{e} + k_2 e = 0 \tag{15}$$

Due to the system dynamics that are normally unidentified in real-life implementation, the optimal controller $u^*$ in (14) is unable to be precisely acquired. Nevertheless, it is possible to use the sliding mode controller to address this matter. Before the required controller is developed, the nominal model (4) can be rephrased as:

$$\ddot{x} = f_n(x) + g_n(x)u \tag{16}$$

where $f_n$ and $g_n$ denote the nominal behavior of $f$ and $g$, correspondingly. Considering uncertainties and external disturbances, (16) can be further revised as:

$$\ddot{x} = (f_n + \Delta f) + (g_n + \Delta g)u + d = f_n + g_n u + w \tag{17}$$

where $d$ is the external disturbance, i.e., wind, $\Delta f$ and $\Delta g$ are system uncertainties, $w$ is lumped with uncertainties and defined as $w = \Delta f + \Delta g u + d$, with the assumption $|w| \leq W$, where $W$ is a positive constant.

The significant idea for sliding mode control is to confirm that the system meets the essential condition and is guaranteed to possess a sliding environment. Thus, a sliding surface can be stated as:

$$s = \dot{e} + k_1 e + k_2 \int e dt \tag{18}$$

where $k_1$ and $k_2$ are both positive coefficients. The law for sliding-mode control, on the other hand, is given as:

$$u_{sc} = u_{eq} + u_{ht} \tag{19}$$

where $u_{eq}$ is the comparable controller and can be described as:

$$u_{eq} = g_n^{-1}\left(-f_n + \ddot{x} + k_1\dot{e} + k_2 e\right) \tag{20}$$

while the hitting controller, $u_{ht}$ is designed to guarantee system stability and is described as:

$$u_{ht} = \frac{W}{g_n} sgn(s) \& \ where \ sgn(s) \begin{cases} +1 \ if \ s < 0 \\ -1 \ if \ s > 0 \end{cases} \tag{21}$$

The derivative of (18) gives:

$$\dot{s} = \ddot{e} + k_1\dot{e} + k_2 e \tag{22}$$

Inserting (19–21) into (17) yields:

$$\ddot{e} + k_1\dot{e} + k_2 e = -w - Wsgn(s) = \dot{s} \tag{23}$$

For the purpose of stability, a Lyapunov function is expressed as:

$$V_1 = \frac{1}{2}s^2 \tag{24}$$

Differentiating (24) in conjunction with time reads:

$$\dot{V}_1 = s\dot{s} \tag{25}$$

For stability $\dot{V}_1 \leq 0$, substituting (23) into (25) yields:

$$\dot{V}_1 = -(W - |w|)|s| \tag{26}$$

In short, the sliding mode control law that is established according to the Lyapunov theorem will ensure the stability of the system. However, a larger control gain, $W$, will cause a chattering effect. Furthermore, the switching function is not easily implementable because of the existence of physical limitations on the rotor deployed by the MAV.

### 3.3.1. PID Controller Design

A conventional PID controller can normally be defined as:

$$u_{pid} = \hat{k}_p e + \hat{k}_i \int edt + \hat{k}_d \dot{e} \tag{27}$$

where $\hat{k}_p$, $\hat{k}_i$, and $\hat{k}_d$ are the value of proportional gain, integral gain, and derivative gain, respectively.

For stability, $\dot{s} = 0$, and by using (14) and (22) it can be obtained that:

$$\dot{s} = \ddot{e} + k_1\dot{e} + k_2 e = -f - gu + A_d \tag{28}$$

where $A_d = \ddot{x}_d + k_1\dot{e} + k_2 e$, substitute (27) into (28) and multiply by $s$,

$$s\dot{s} = -s\left[g\left(u_{pid}\right) + f - A_d\right] \tag{29}$$

Applying the methods of gradient and the rule of chain, $\hat{k}_p$, $\hat{k}_i$ and $\hat{k}_d$ gains are revised using the rules as in:

$$\begin{aligned} \dot{\hat{k}}_p &= s\beta_p sgn(g)e = \beta_p se \\ \dot{\hat{k}}_i &= s\beta_i sgn(g) \int edt = \beta_i s \int edt \\ \dot{\hat{k}}_d &= s\beta_d sgn(g)\dot{e} = \beta_d s\dot{e} \end{aligned} \tag{30}$$

where $\beta_p$, $\beta_i$, and $\beta_d$ are the positive learning rates of $\dot{\hat{k}}_p$, $\dot{\hat{k}}_i$, and $\dot{\hat{k}}_d$, respectively. Furthermore, the designed approach necessitates the use of the $g$ value, which can be readily attained from the physical features of the controlled system [32,33].

### 3.3.2. Design of Fuzzy Compensator

For the fuzzy compensator, three fuzzy rules are fixed, as outlined in (31), where $P$ is positive, $Z$ is zero, and $N$ is negative.

$$\begin{aligned} &Rule\ 1: \ If\ s\ is\ P,\ the\ u_{fc}\ is\ P \\ &Rule\ 2: \ If\ s\ is\ Z,\ the\ u_{fc}\ is\ Z \\ &Rule\ 3: \ If\ s\ is\ N,\ the\ u_{fc}\ is\ N \end{aligned} \tag{31}$$

where the triangular and singleton functions are applied to describe the input and the output membership functions. This is shown in Figure 4a,b, correspondingly. Through the center of gravity approach, the defuzzification is achieved as:

$$u_{fc} = \frac{\sum_{i=1}^{3} r_i w_i}{\sum_{i=1}^{3} w_i} = r_1 w_1 + r_2 w_2 + r_3 w_3 \tag{32}$$

where $w_i$ $(i = 1, 2, 3)$ refers to the firing strength and restricted to $(0 \le w_i \le 1)$. The total of $w_i$ is less than or equal to one. This condition is unique in the case of the triangle membership function. In order to reduce the loading of calculations, $r_1 = \hat{r}$, $r_2 = 0$, and $r_3 = -\hat{r}$ settings are made. Hence, only one of the four cases, as mention in [32,33], will occur at any set value of input $s$, as in:

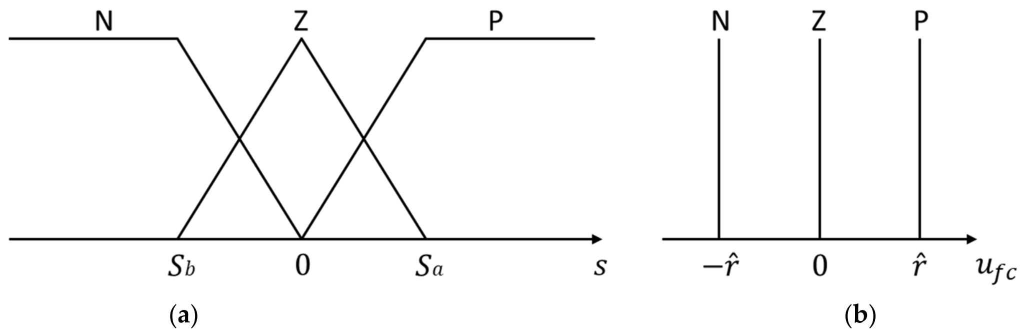

**Figure 4.** Membership function of (**a**) fuzzy input and (**b**) fuzzy output, where P is positive, Z is zero, and N is negative.

Case 1: Only rule 1 is triggered $(s > s_a,\ w_1 = 1,\ w_2 = w_3 = 0)$

$$u_{fc} = r_1 = \hat{r} \tag{33}$$

Case 2: Rules 1 and 2 are triggered simultaneously $(0 < s \leq s_a,\ 0 < w_1,\ w_2 \leq 1,\ w_3 = 0)$

$$u_{fc} = r_1 w_1 = \hat{r} w_1 \tag{34}$$

Case 3: Rules 2 and 3 are triggered simultaneously $(s_b < s \leq 0,\ w_1 = 0,\ 0 < w_2,\ w_3 \leq 1)$

$$u_{fc} = r_3 w_3 = -\hat{r} w_3 \tag{35}$$

Case 4: Only rule 3 is triggered $(s \leq s_b,\ w_1 = w_2 = 0,\ w_3 = 1)$

$$u_{fc} = r_3 = -\hat{r} \tag{36}$$

Then, (33)–(36) can be reproduced as:

$$u_{fc} = \hat{r}(w_1 - w_3) \tag{37}$$

Furthermore, it can be seen from [32,33]

$$s(w_1 - w_3) = |s||(w_1 - w_3)| \geq 0 \tag{38}$$

## *3.4. Stability Analysis*

By substituting (12) into (4), it is revealed that

$$\ddot{x} = f + g\left(u_{pid} + u_{fc}\right) \tag{39}$$

The error equation representing the system can also be acquired after direct exploitation of (14), (22), and (39), as follows:

$$\ddot{e} + k_i \dot{e} + k_2 e = g\left(u^* - u_{pid} - u_{fc}\right) = \dot{s} \tag{40}$$

In case of any estimation error, the optimal controller equation can be readjusted as:

$$u^* = u_{pid}\left(\hat{k}_p, \hat{k}_i, \hat{k}_d\right) + \varepsilon \tag{41}$$

where $\varepsilon$ refers to an estimation error and is thought to be restricted by $0 \leq |\varepsilon| \leq E$, where $E$ is a positive constant. Therefore, (40) is rewritten as:

$$\dot{s} = g\left(\varepsilon - u_{fc}\right) \tag{42}$$

A Lyapunov function can then be selected as:

$$V_2 = \frac{1}{2}s^2 \tag{43}$$

Differentiating (43) with regard to time, as well as deploying (37) and (42), gives:

$$\dot{V}_2 = -g|s||w_1 - w_3|\left(\hat{r} - \frac{|\varepsilon|}{|w_1 - w_3|}\right) \tag{44}$$

If inequality occurs in

$$\hat{r} > \frac{|\varepsilon|}{|w_1 - w_3|} \tag{45}$$

then it is probable to fulfil the sliding condition $\dot{V}_2 \leq 0$. Having the unknown lumped uncertainties causes the value $\hat{r}$ to be unachievable in advance, for practical applications. Corresponding to (37), in order to attain the minimum sliding value, an optimal $r^*$ value needs to be obtained, as follows:

$$r^* = \frac{|\varepsilon|}{|w_1 - w_3|} + \kappa \tag{46}$$

where $\kappa$ denotes a positive constant. Thus, a direct adaptive algorithm is used to gauge the optimal value of $r^*$, and the expected error can be described as:

$$\widetilde{r} = r^* - \hat{r} \tag{47}$$

where, $\hat{r}$ is the projected ideal value of $r^*$. Next, a new Lyapunov function can be outlined as:

$$V_3 = \frac{1}{2}s^2 + \frac{g}{2\eta_r}\widetilde{r}^2 \tag{48}$$

where, $\eta_r$ is a positive constant learning rate, differentiating (48) with regard to time and deploying (37), (42), and (46), gives:

$$\dot{V}_3 \leq g|s||\varepsilon| - \widetilde{r}g\left[s(w_1 - w_3) + \frac{\dot{\widetilde{r}}}{\eta_r}\right] - r^* s g(w_1 - w_3) \tag{49}$$

The parameter tuning law is chosen as:

$$\dot{\widetilde{r}} = -\dot{\hat{r}} = -\eta_r s(w_1 - w_3) \tag{50}$$

and by (46), (49) becomes:

$$\dot{V}_3 = -g\kappa|s||w_1 - w_3| \leq 0 \tag{51}$$

Therefore, $\dot{V}_3$ is negative and semidefinite, where $V_3(s, \widetilde{r}, t) \leq V_3(s, \widetilde{r}, 0)$. This suggests that $s$ and $\widetilde{r}$ are restricted. Let function $\Omega(\tau) \equiv g\kappa|s||w_1 - w_3| \leq -\dot{V}_3$, and $\Omega(t)$ is integrated with respect to time:

$$\int_0^t \Omega(\tau)d\tau \leq V_3(s, \widetilde{r}, 0) - V_3(s, \widetilde{r}, t) \tag{52}$$

Since $V_3(s, \widetilde{r}, 0)$ is restricted, and $V_3(s, \widetilde{r}, t)$ is non-increasing and restricted, the following output can be acquired:

$$\lim_{t \to \infty} \int_0^t \Omega(\tau)d\tau < \infty \tag{53}$$

Furthermore, since $\dot{\Omega}(t)$ is restricted, as shown by Barbalat's Lemma [33], $\lim_{t \to \infty} \Omega(t) = 0$, $s \to 0$ as $t \to \infty$. Therefore, the stability of the proposed APIDC system can be assured.

To conclude, if the $\beta_p$, $\beta_i$, and $\beta_d$ learning rate or initial $\hat{k}_p$, $\hat{k}_i$, and $\hat{k}_d$ of PID gains are not properly picked, the state of the system will deviate. Hence, the learning rate can be tuned manually or by applying the optimization technique. Meanwhile, the fuzzy compensator in (34) plays a vital part, offering additional input for the state reversal. The algorithm to apply this control method is as described by [32,33]:

1. APIDC system parameter initialization.
2. Utilization of tracking error, $e = x_d - x$, and the sliding surface, $s$, as denoted in (18)
3. Implementation of the PID controller, $u_{pid}$, as shown in (27), using the gains parameter $\hat{k}_p$, $\hat{k}_i$ and $\hat{k}_d$, as revised by (30).
4. Usage of the fuzzy compensator, as described in (37), with the parameter $\hat{r}$ is as projected by (50).
5. Application of the control law, as specified in (12).
6. Return to Step 2

## 4. Simulation Results

Four simulations and experiments were conducted to show the proposed controller scheme performance. A comparison with a classical PID plus gravity compensator was made to show the effectiveness of proposed controller scheme. The proposed APIDC controller is designed in the Flight Control System of the Quadcopter Flight Simulation Model for the simulations and then deployed wirelessly through Bluetooth to the embedded system on the Parrot Mambo Minidrone for real-time implementation, as shown in Figure 5. The state estimation is declared beforehand in the provided package, with the sensor fusion algorithm in the Sensor Model designed by a complementary filter and a Kalman filter to process the already available onboard sensors, such as the ultrasonic sensor, IMU, air pressure sensor, and optical flow sensor [15]. The Sensor Model and the Environment Model were the perturbation injected to the Nonlinear Airframe of the Multi-copter Model.

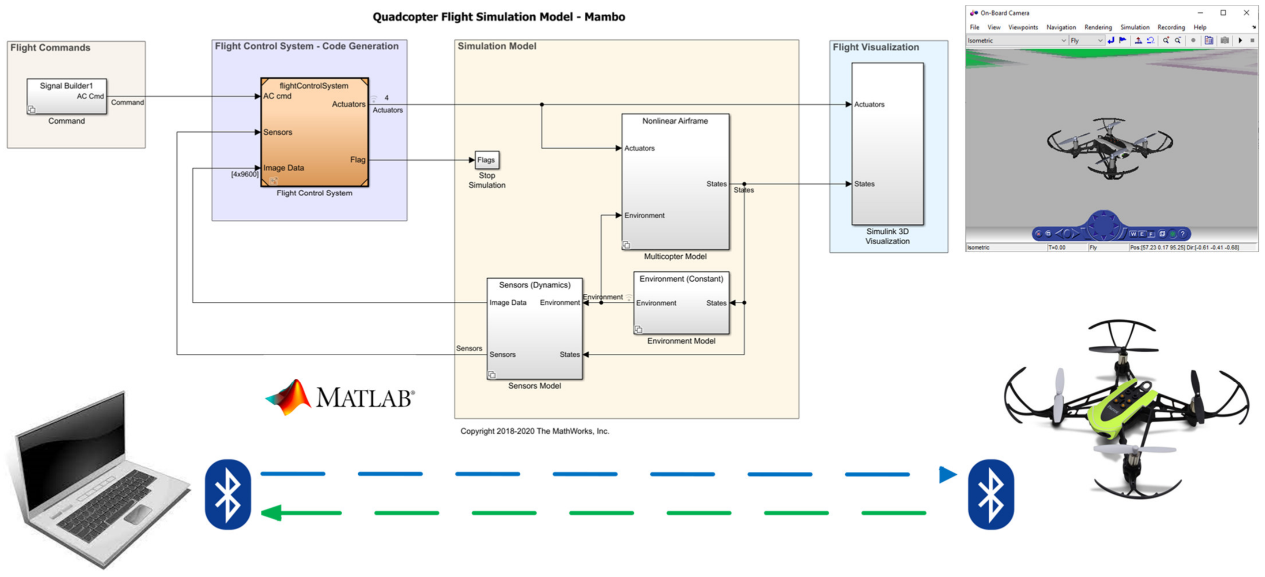

**Figure 5.** Simulation and experimental setup for the Parrot Mambo Minidrone.

All the gains parameters of the PID, APID, and the fuzzy compensator are listed in Tables 2 and 3, respectively. These PID gains were the default setting of the Quadcopter Flight Simulation Model, and the APID and fuzzy compensator gains were set by trial and error until we obtained the best results, both for the simulations and the experiments, which were set to last for 60 s. Four simulations and experiments were conducted to show the performance of the altitude quadrotor under various scenarios. The simulation results and experimental data for each of these scenarios are presented in the following sections.

**Table 2.** PID gains parameter.

| | **Position/Angle Control** | **Velocity Control** | | |
|---|---|---|---|---|
| **State** \ **Gains** | **Kp** | **Kp** | **Ki** | **Kd** |
| ***Z***: PID | | 0.80000 | 0.24000 | 0.5000000 |
| ***Y***: P-PI | 0.7 | 0.20000 | 0.10000 | |
| ***X***: P-PI | 0.7 | 0.20000 | 0.10000 | |
| $\psi$: PID | | 0.00400 | 0.00400 | 0.0012000 |
| $\theta$: P-PID | 4 | 0.00300 | 0.00600 | 0.0001200 |
| $\phi$: P-PID | 4 | 0.00243 | 0.00486 | 0.0000972 |

**Table 3.** APID and fuzzy compensator gains parameters.

| | **Sliding** | | **Learning Rate APID** | | | **Learning Rate Fuzzy Compensator** |
|---|---|---|---|---|---|---|
| State | $k_1$ | $k_2$ | $\beta_p$ | $\beta_i$ | $\beta_d$ | $\eta_{fz}$ |
| ***Z*** | 0.1 | 0.5 | 1 | 0.1 | 0.1 | $-0.001$ |

## 4.1. Hovering Test

The initial condition for the test was set above ground level with the Parrot Mambo hovering at a height of 1.1 m for 60 s. The Landing Logic Algorithm will trigger 5 s before the simulation ends to let the drone land safely. Figure 6 shows the controller response during the simulation, which shows a slight overshoot for the APID and APIDFC compared to the PID controller. We consider that the learning rate takes a few seconds to adjust the parameter gains to provide good performance. In this simulation, both APID and APIDFC responses overlap with each other, and we cannot yet determine the difference between these controllers. Figure 7 shows that all controllers require the same speed of around 250 rotations/second to maintain the quadrotor hovering height.

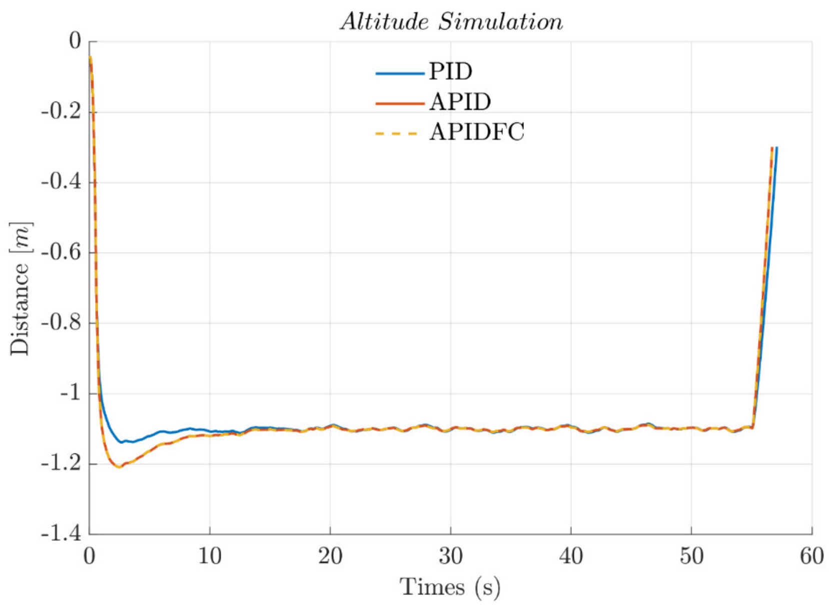

**Figure 6.** Hovering simulation.

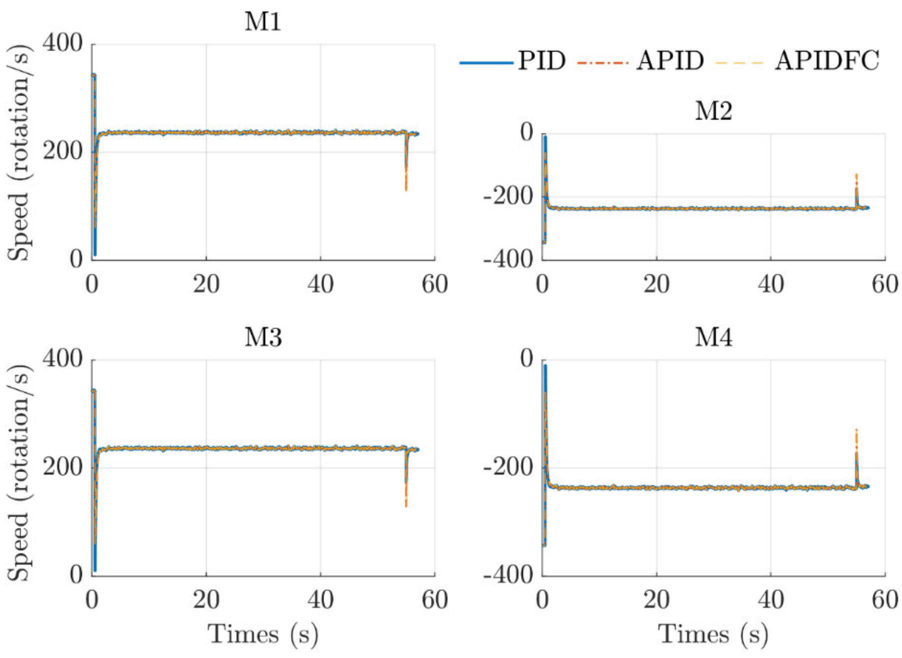

**Figure 7.** Motor rotation speed in the hovering simulation.

### *4.2. Altitude Tracking Test*

Three simulations were conducted in this tracking; sine wave, square wave, and trapezium wave tracking. Sine wave tracking is conducted to observe the smoothness of the quadrotor in following the curve signals, while square wave and trapezium wave tracking are performed to check the response of the quadrotor when the altitude changes to a certain level. From the simulations, all controllers can adequately track the desired responses with minimum error. However, the PID controller produces a slight overshoot compared to the APID and APIDFC. Both APID and APIDFC still overlap with each other, and we cannot yet determine which one is better. All the tracking responses are shown in Figures 8–10.

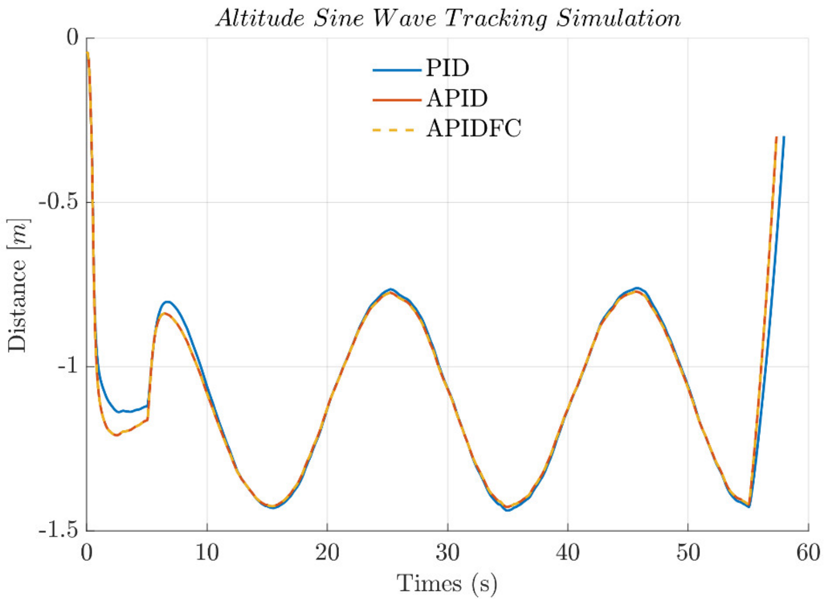

**Figure 8.** Sine wave tracking in the simulation.

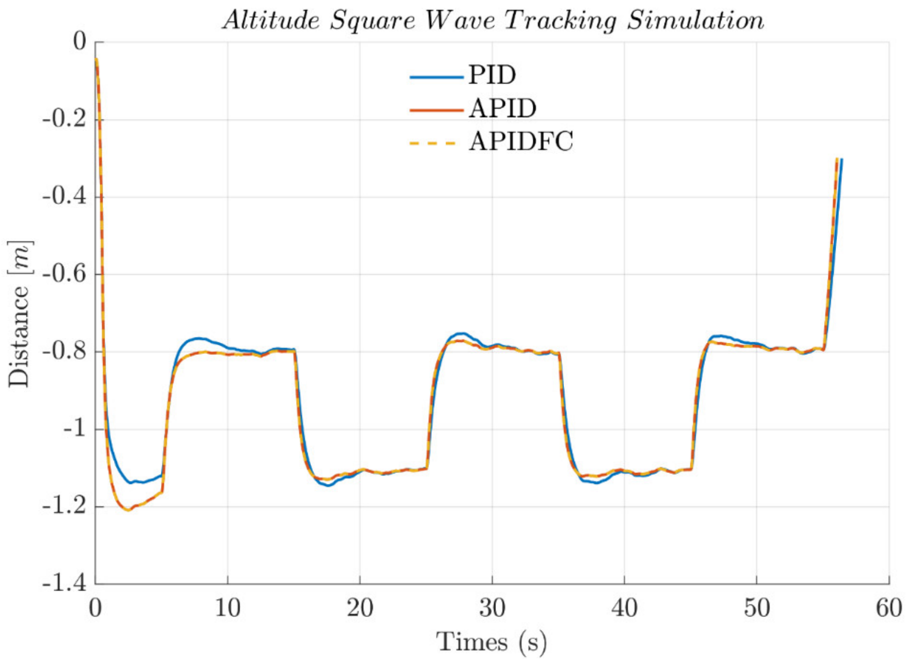

**Figure 9.** Square wave tracking in the simulation.

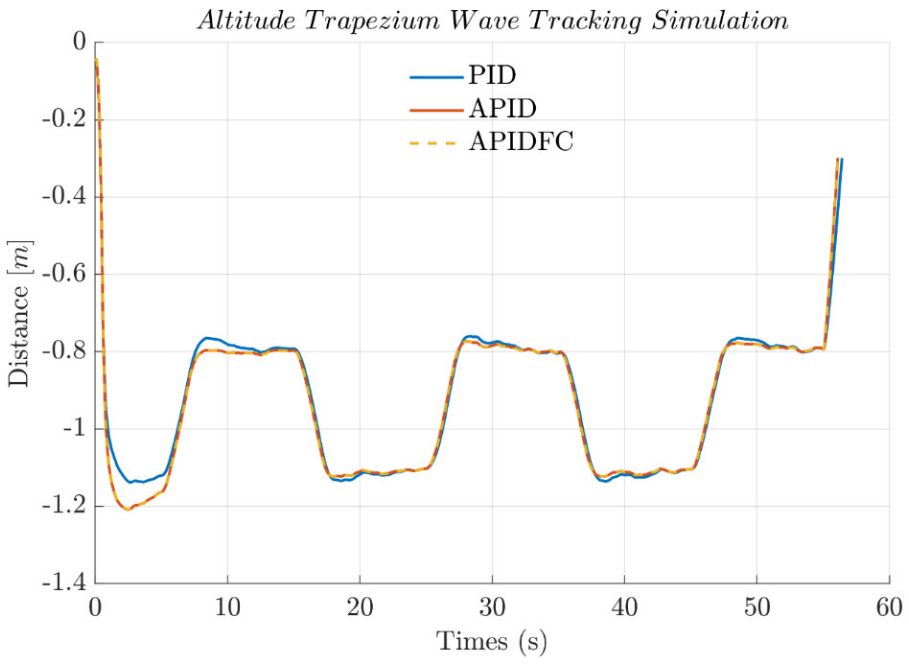

**Figure 10.** Trapezium wave tracking in the simulation.

### *4.3. Changing the Mass during the Hovering Test*

The final simulation involves altering the mass during the hover test to assess how resilient the controls are to perturbations. A perturbation of 0.122625 N was injected into the control signal after 15 s to observe the controllers' behaviors. From the results shown in Figures 11 and 12, all the controllers perform well and can restabilize the drone within 5 s to the setting height of 1.1 m. However, the PID controller produces a slight overshoot compared to the APID and APIDFC. In this simulation, the propellor rotation speeds overlap, and the performance of the controller cannot be justified.

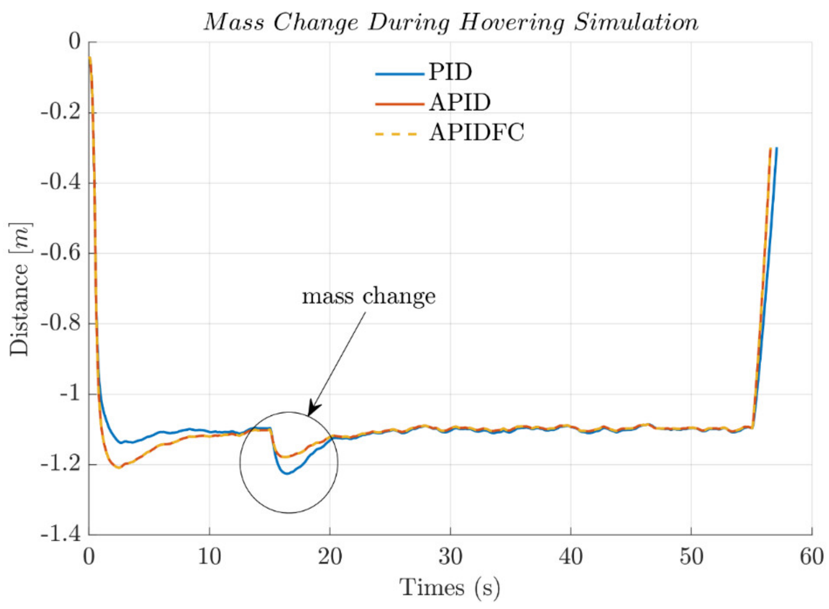

**Figure 11.** Changing mass during hovering in the simulation.

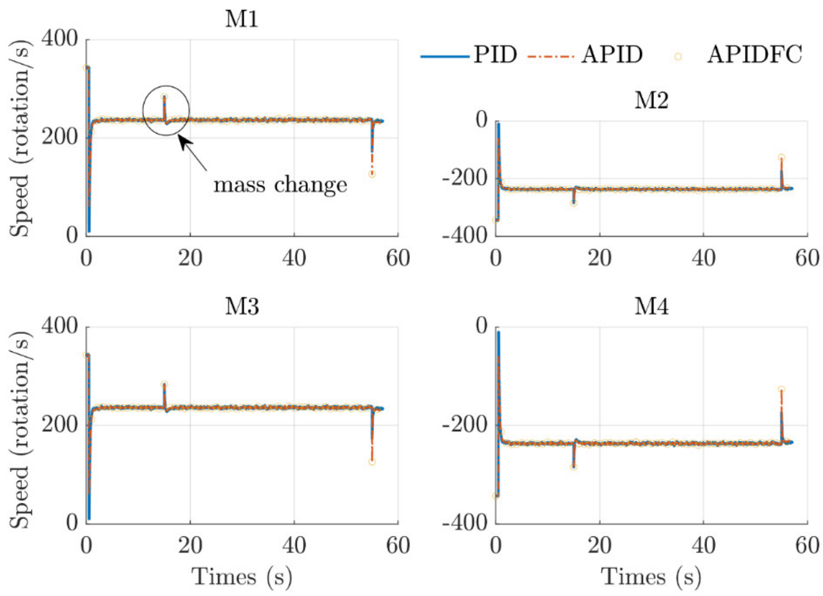

**Figure 12.** Motor speed rotation for mass change during hovering in the simulation.

## 5. Experimental Results

The layout and test field is shown in Figure 13. Masking tape on the black surface of the floor was utilized to create contrast to improve the optical flow sensitivity for altitude estimation. All the settings and controller designs were followed for the simulation, and the "Flight Control System—Code Generator" block, highlighted with orange color in Figure 5, was built and deployed wirelessly to the Parrot Mambo Minidrone. Additionally, the experiments' index performances were recorded, based on integral square error (ISE) and tabulated in Table 4 for comparison. In this experiment, the assumptions were as follows:

1. The experiment is conducted in an air conditioning hall.
2. The Parrot Mambo Minidrone structure is assumed to be in a good condition.
3. The propellors are assumed to be in good condition, with no dents.
4. All the motors are assumed to be in good condition.
5. The execution starting point is always the same.
6. The lighting condition is considered fair to good.

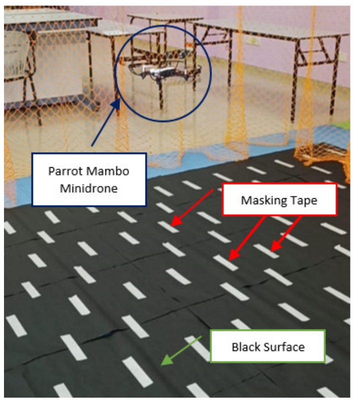

**Figure 13.** Test field and a Parrot Mambo Minidrone during execution.

**Table 4.** Experiment performance index for PID, APID, and APIDFC.

| | **PID** | **APID** | **APIDFC** |
|---|---|---|---|
| Hover | 1.4586 | 1.3801 | 1.2877 |
| Sine | 1.6831 | 1.3675 | 1.3048 |
| Square | 1.6761 | 1.4327 | 1.3952 |
| Trapezium | 1.5323 | 1.2611 | 1.1983 |
| Mass Change | 1.4700 | 0.8003 | 0.7937 |

## *5.1. Hovering Test*

Figure 14 shows the transient response of the altitude for 60 s of flight. The desired altitude is set at −1.1 m, as predefined in the "MATLAB Original Setting". There was no overshot produced by the PID controller, but overshot was noted for the APID and APIDC. The results are still acceptable, since the adaptive mechanism needs time to compute the controller's gain parameters, finally settling and stabilizing within 10 s. APIDC produces a 13.29% overshoot, while APID produces a 14.62% overshoot. Figure 15 shows that the controller requires the motors to rotate about 380–400 per second to maintain the altitude at −1.1 m from ground. Then, based on the ISE, Table 4 clearly shows both that the results for APID (1.3801), and APIDFC (1.2877) were better than for PID (1.4586).

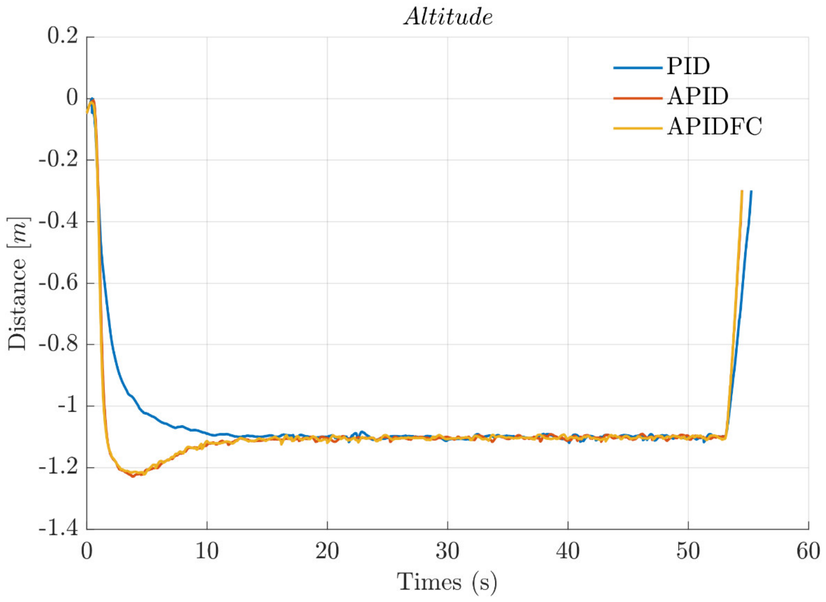

**Figure 14.** Altitude response comparison according to the experiments.

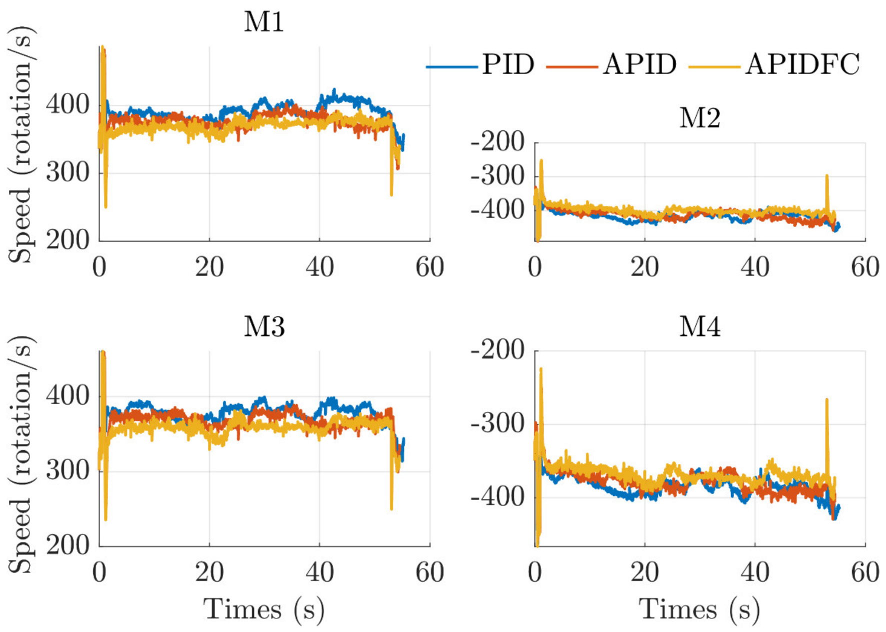

**Figure 15.** Motor rotation speed comparison between controllers.

## *5.2. Altitude Tracking Test*

The experiment continues to show how the Parrot Mambo Minidrone reacts to following sine wave, square wave, and trapezium wave tracking. For all cases, the altitude cannot be set lower than 0.6 m due to the optical sensor and ultrasonic sensor limitations. As a result, the MAV can follow the dedicated references, as shown in Figures 16–18. However, we can claim that the APID and APIDFC controllers provided better performance compared to the PID controller, as proved by the ISE tabulated in Table 4. During the executions, observations show that the MAV was able to follow the sine, trapezium, and square references. Figure 19 shows the motor 1 (M1) speed reactions during this execution. The spike in the figure shows the immediate change of rotation speed of the motor. Suddenly changing attitude will immediately reduce the motor speed from higher to lower, which can cause an error in altitude estimation due to the limits of the optical flow and ultrasonic sensors.

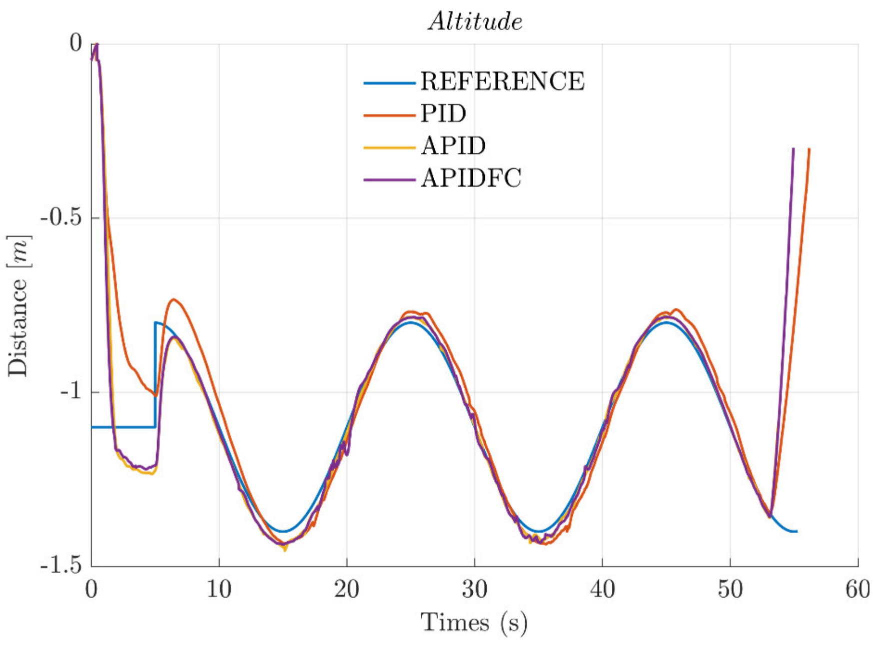

**Figure 16.** Altitude response to following the sine wave references.

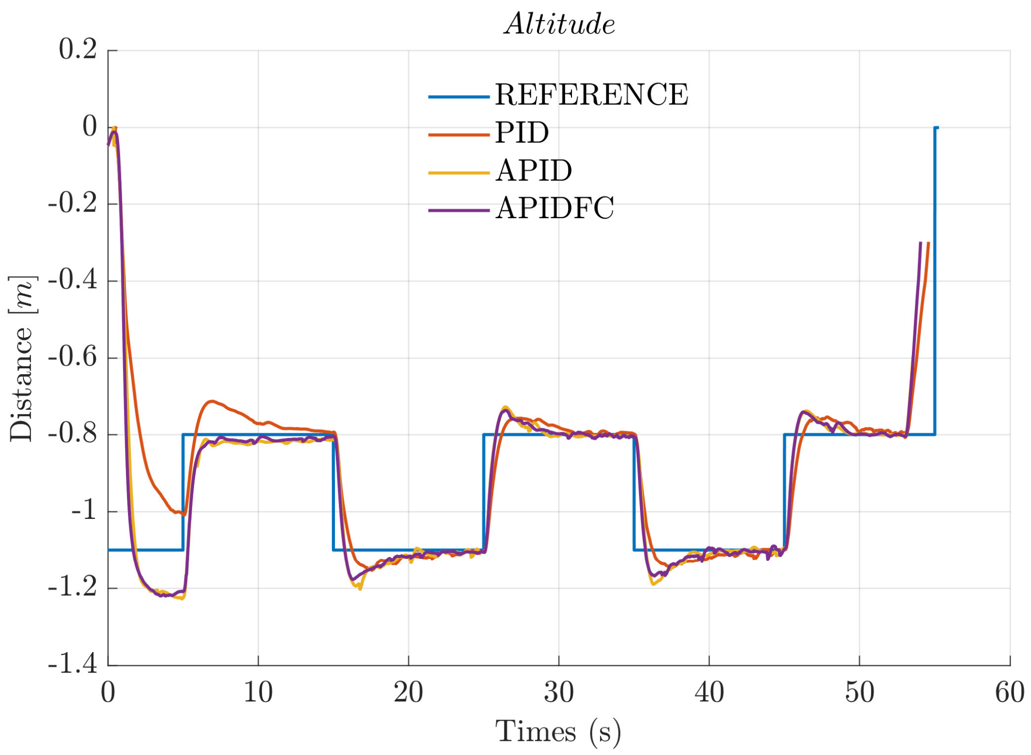

**Figure 17.** Altitude response to following the square wave reference.

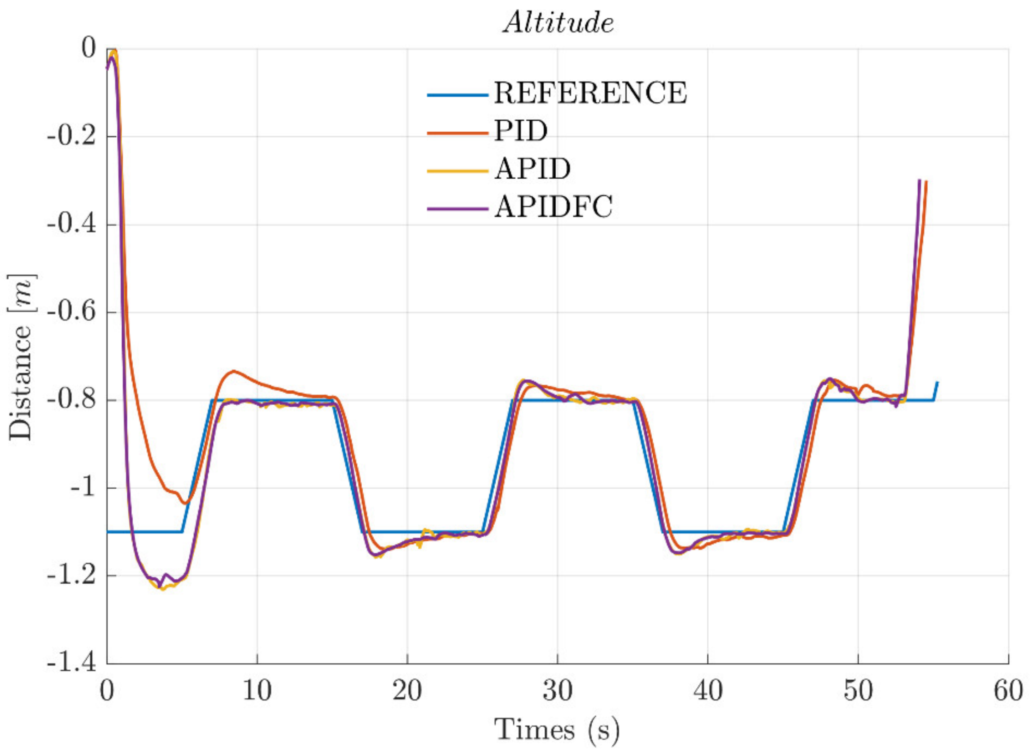

**Figure 18.** Altitude response to following the trapezium wave reference.

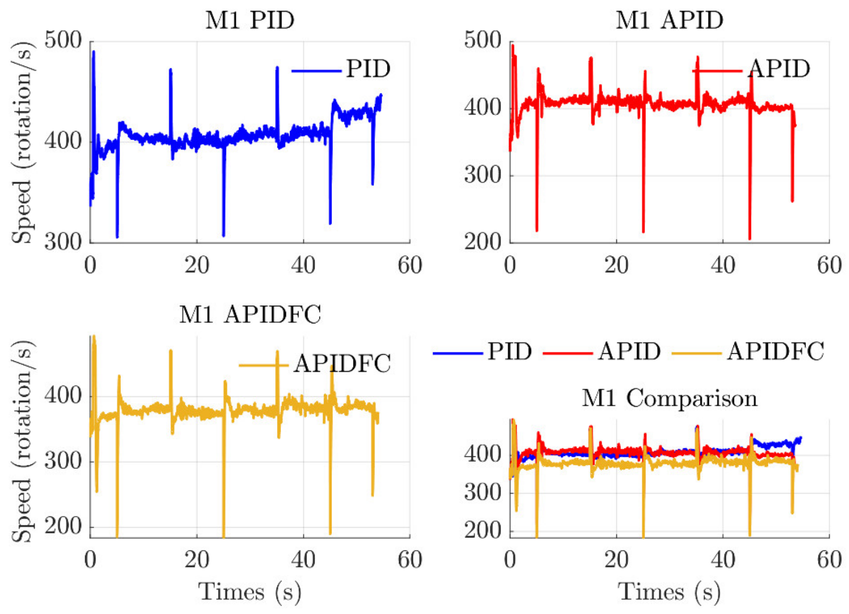

**Figure 19.** Motor M1 rotation speed comparison between controllers.

### *5.3. Changing Mass during Hovering Test*

In this test, a cylindrical type of mass of about 12.5 g was immediately added to the Parrot Mambo after it had been hovering for about 15 s. For all cases, the PID controller, APID controller, and APID with fuzzy compensator controller managed to balance the mass and stabilized within few seconds, as shown in Figure 20. Even though there were slight spikes that occurred throughout, the steady-state error is small and acceptable for the case of quadrotor. In addition, the APID and APIDFC controllers show better reactions to stabilize the drone with less overshoot. Figure 21 shows a comparison for all motor rotation speeds of the controllers during the experiments. The results clearly show that PID requires greater speed, about 400 rotation per second, to adapt to the change in mass during hovering, compared to 300–380 rotation per second for the APID and APIDFC controllers. This shows the proposed control scheme APID, without or with fuzzy compensator, was better than the PID controller, with less energy consumption and better performance.

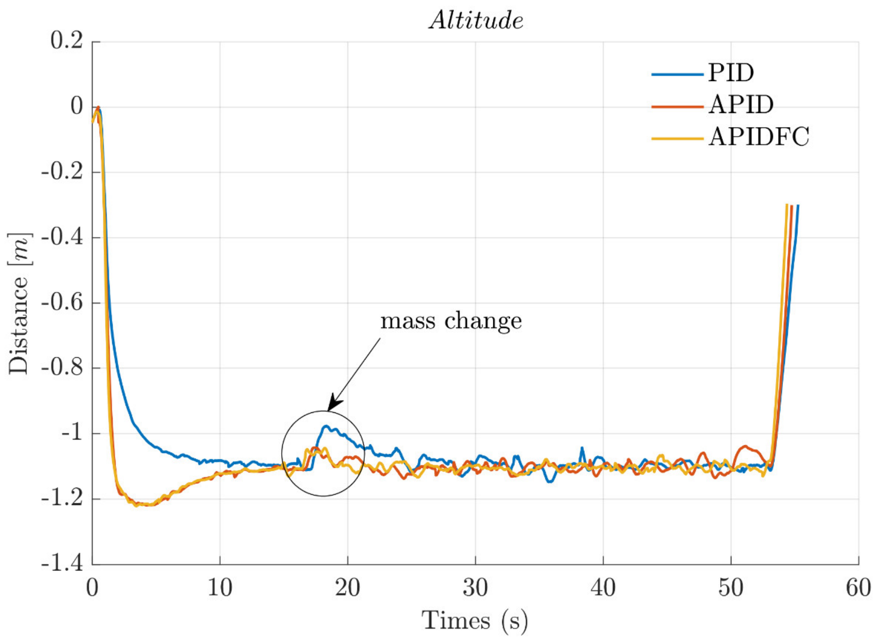

**Figure 20.** Altitude response comparison from the experiment.

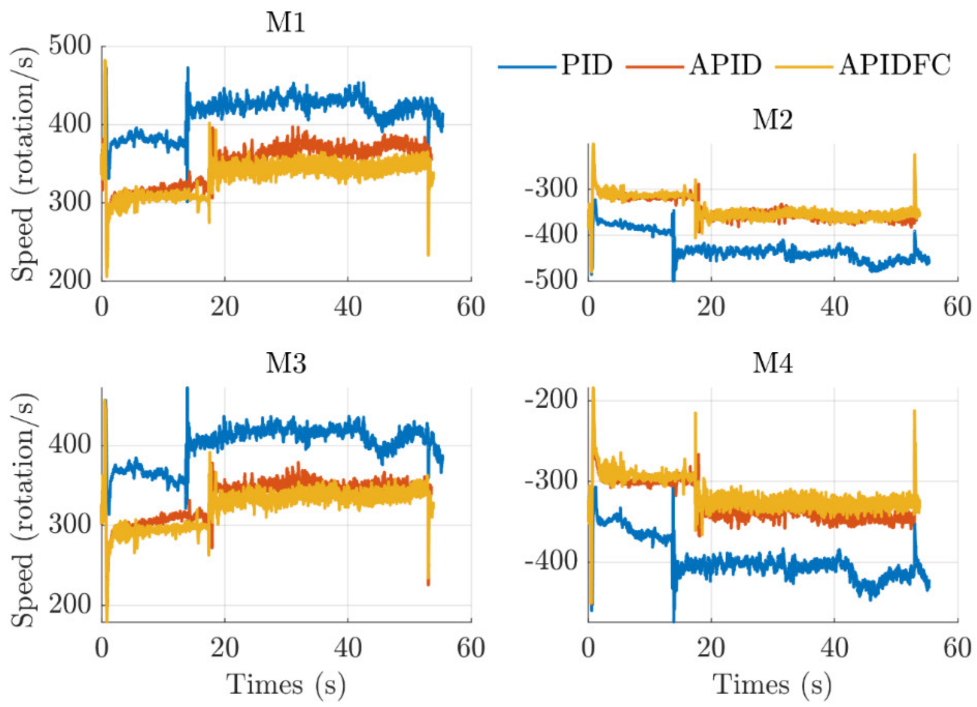

**Figure 21.** Motors rotation comparison from the experiment.

The control effort comparison between the controllers on $U1$ (Figure 22) clearly shows that the APID and APIDFC controllers require slightly less effort than the PID. Then even though $U2$, $U3$, and $U4$ use the same control technique, the control effort for this signal improves significantly when the altitude controller is based on APID and APIDFC. Moreover, it was found that in the experiment that the APID and APIDFC consumed 2% and 4% less power, respectively, when compared to the PID when hovering the MAV.

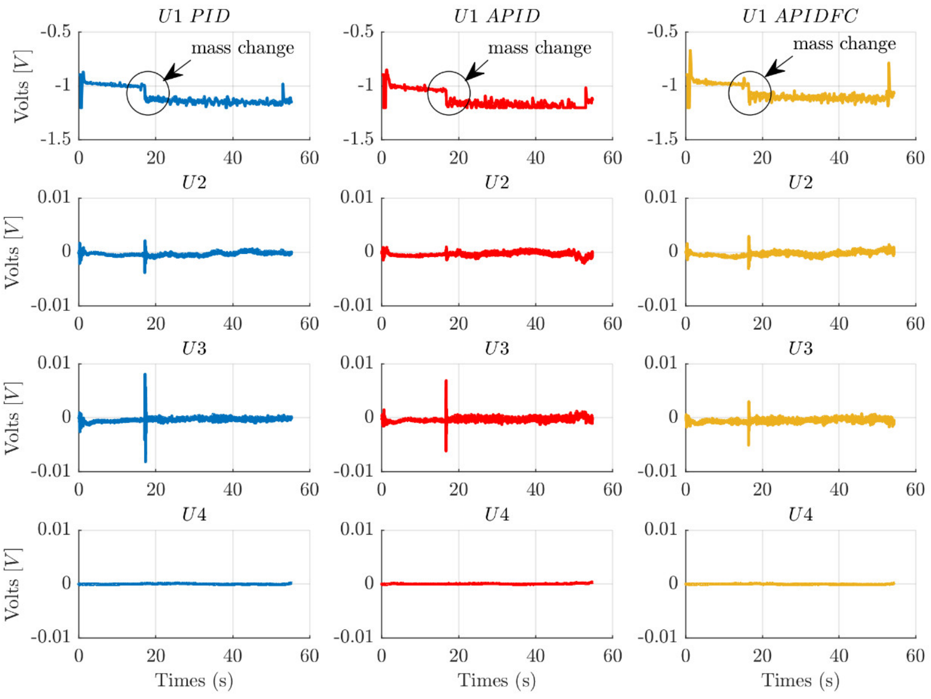

**Figure 22.** Control effort comparison between controllers.

Figure 23 shows that the adaptive gains for both APID and APIDFC can converge to a certain point, while Figure 24 shows less chattering produced by the second-order sliding mode, with an integral or sliding manifold for both controllers. By introducing a fuzzy compensator, it reduces the occurrence of chattering phenomena on the input signal by about 6%. According to Table 4, for the mass change event during hovering, under PID control, the MAV produces 1.4700 ISE. The ISE does, however, drop by 45% under APID control and 46% while under APIDFC control. Hence, with a fuzzy compensator, the flight control system can provide additional improvement to the MAV during flight control. Thus, we can claim that both APID and APIDFC were more robust compared to the PID control scheme.

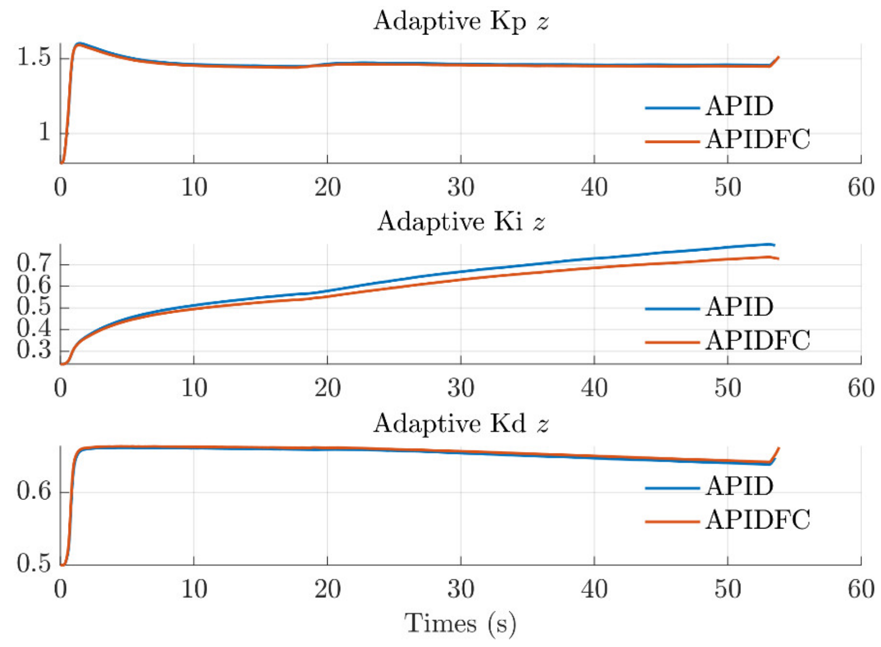

**Figure 23.** Adaptive parameter for altitude $Z$ from the experiment.

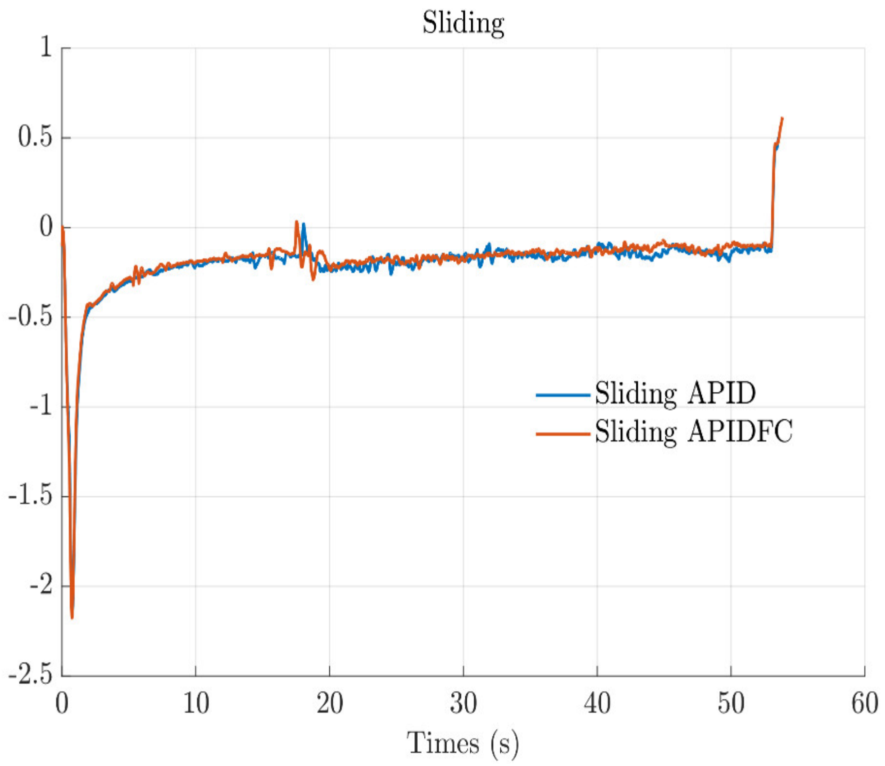

**Figure 24.** Sliding manifold for APID and APIDFC controllers.

## 6. Conclusions

This paper presents the real-time implementation of altitude controllers using PID, adaptive PID (APID), and APID with a fuzzy compensator (APIDFC) for a MAV, specifically, for a Parrot Mambo Minidrone. The adaptive mechanism, based on a second-order sliding mode control, is utilized to alter the classical parameters of the PID controller of the altitude controller. Four experiments were conducted: hovering, as well as sine, square, and trapezium tracking. In addition, the robustness of the controller was tested by adding mass of about 12.5 g, 15 s after flight launch, at 1.1 m altitude above ground. In the experiment, the propeller rotations were about 30–60 rotations less when using APID and APIDFC controllers. Then, by introducing a fuzzy compensator, it reduces the occurrence of chattering phenomena on the input signal by about 6%, which is due to the implementation of a sliding mode control. Furthermore, both methods also consume 2% and 4% less power, respectively, when compared to the PID controller used to hover the MAV. Hence, based on this observation, the APID and APIDFC controllers were significantly robust and energy-efficient compared to the PID controller.

**Author Contributions:** Conceptualization, M.A.M.B.; Methodology, A.N.; Software, A.N.; Validation, M.A.M.B.; Formal analysis, A.N.; Investigation, A.N.; Resources, A.N.; Writing—original draft, A.N.; Writing—review & editing, M.A.M.B. and Z.M.; Supervision, M.A.M.B. and Z.M. All authors have read and agreed to the published version of the manuscript.

**Funding:** The authors would like to thank Universiti Teknikal Malaysia Melaka (UTeM), Universiti Teknologi Malaysia (UTM), under the Research University Grant UTMFR (Q.J130000.3823.22H67), the Ministry of Higher Education (MOHE), and the Advanced Academia-Industrial Testing Laboratory (AiTL), UTeM for supporting this research.

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

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

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

## References

1. Huang, H.; Hoffmann, G.M.; Waslander, S.L.; Tomlin, C.J. Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; IEEE Press: New York, NY, USA, 2009; pp. 3277–3282. [CrossRef]
2. Lori, A.A.R.; Danesh, M.; Amiri, P.; Ashkoofaraz, S.Y.; Azargoon, M.A. Transportation of an Unknown Cable-Suspended Payload by a Quadrotor in Windy Environment under Aerodynamics Effects. In Proceedings of the 2021 7th International Conference on Control, Instrumentation and Automation (ICCIA), Tabriz, Iran, 23–24 February 2021; IEEE Press: New York, NY, USA, 2021. [CrossRef]
3. Candan, F.; Beke, A.; Kumbasar, T. Design and Deployment of Fuzzy PID Controllers to the nano quadcopter Crazyflie 2.0. In Proceedings of the 2018 Innovations in Intelligent Systems and Applications (INISTA), Thessaloniki, Greece, 3–5 July 2018; IEEE Press: New York, NY, USA, 2018. [CrossRef]
4. Preiss, J.A.; Wolfgang, H.; Sukhatme, G.S. Ayanian. Crazyswarm: A Large Nano-Quadcopter Swarm. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; IEEE Press: New York, NY, USA, 2017; pp. 3299–3304.
5. Yoo, J.; Jang, D.; Kim, H.J.; Johansson, K.H. Hybrid Reinforcement Learning Control for a Micro Quadrotor Flight. *IEEE Control Syst. Lett.* **2021**, *5*, 505–510. [CrossRef]
6. Hönig, W.; Ayanian, N. Flying multiple UAVs using ROS. In *Robot Operating System (ROS)*; Springer: Cham, Switzerland, 2017; Volume 707, pp. 83–118.
7. KSubash, V.V.; Srinu, M.V.; Siddhartha, M.R.V.; Harsha, N.C.S.; Akkala, P. Object Detection using Ryze Tello Drone with Help of Mask-RCNN. In Proceedings of the 2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), Bangalore, India, 5–7 March 2020; IEEE Press: New York, NY, USA, 2020; pp. 484–490. [CrossRef]
8. Pohudina, O.; Kovalevskyi, M.; Pyvovar, M. Group Flight Automation Using Tello EDU Unmanned Aerial Vehicle. In Proceedings of the 2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT), LVIV, Ukraine, 22–25 September 2021; IEEE Press: New York, NY, USA, 2021; Volume 2, pp. 151–154. [CrossRef]

9. Giernacki, W.; Rao, J.; Sladic, S.; Bondyra, A.; Retinger, M.; Espinoza-Fraire, T. DJI Tello Quadrotor as a Platform for Research and Education in Mobile Robotics and Control Engineering. In Proceedings of the 2022 International Conference on Unmanned Aircraft Systems (ICUAS), Dubrovnik, Croatia, 21–24 June 2022; IEEE Press: New York, NY, USA, 2022; pp. 735–744. [CrossRef]
10. Saito, T.; Mase, K. Dronepilot.NET development: AR.drone SDK supporting native and managed code. In Proceedings of the 2013 International Conference on Advanced Computer Science Applications and Technologies, Kuching, Malaysia, 23–24 December 2013; IEEE Press: New York, NY, USA, 2013; pp. 60–64. [CrossRef]
11. Indrawati, V.; Prayitno, A.; Utomo, G. Comparison of two fuzzy logic controller schemes for position control of AR.Drone. In Proceedings of the 2015 7th International Conference on Information Technology and Electrical Engineering (ICITEE), Chiang Mai, Thailand, 29–30 October 2015; IEEE Press: New York, NY, USA, 2015; pp. 360–363. [CrossRef]
12. Zhao, T.; Jiang, H. Landing system for AR.Drone 2.0 using onboard camera and ROS. In Proceedings of the 2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC), Nanjing, China, 12–14 August 2016; IEEE Press: New York, NY, USA, 2016; pp. 1098–1102.
13. Babu, V.M.; Das, K.; Kumar, S. Designing of self tuning PID controller for AR drone quadrotor. In Proceedings of the 2017 18th International Conference on Advanced Robotics (ICAR), Hong Kong, China, 10–12 July 2017; IEEE Press: New York, NY, USA, 2017; pp. 167–172. [CrossRef]
14. Kaplan, M.R.; Eraslan, A.; Beke, A.; Kumbasar, T. Altitude and Position Control of Parrot Mambo Minidrone with PID and Fuzzy PID Controllers. In Proceedings of the 2019 11th International Conference on Electrical and Electronics Engineering (ELECO), Bursa, Turkey, 28–30 November 2019; IEEE Press: New York, NY, USA, 2020; pp. 785–789. [CrossRef]
15. Castañeda, H.; Gordillo, J.L. Embedded flight control based on adaptive sliding mode strategy for a quadrotor micro air vehicle. *Electronics* **2019**, *8*, 793. [CrossRef]
16. Alqaisi, W.; Kali, Y.; Ghommam, J.; Saad, M.; Nerguizian, V. Position and attitude tracking of uncertain quadrotor unmanned aerial vehicles based on non-singular terminal super-twisting algorithm. *Proc. Inst. Mech. Eng. Part I J. Syst. Control. Eng.* **2020**, *234*, 396–408. [CrossRef]
17. Noordin, A.; Basri, M.A.M.; Mohamed, Z. Simulation and experimental study on pid control of a quadrotor MAV with perturbation. *Bull. Electr. Eng. Inform.* **2020**, *9*, 1811–1818. [CrossRef]
18. Liu, Z.; Liu, X.; Chen, J.; Fang, C. Altitude control for variable load quadrotor via learning rate based robust sliding mode controller. *IEEE Access* **2019**, *7*, 9736–9744. [CrossRef]
19. Najm, A.A.; Ibraheem, I.K. Nonlinear PID controller design for a 6-DOF UAV quadrotor system. *Eng. Sci. Technol. Int. J.* **2019**, *22*, 1087–1097. [CrossRef]
20. Pounds, P.E.I.; Bersak, D.R.; Dollar, A.M. Stability of small-scale UAV helicopters and quadrotors with added payload mass under PID control. *Auton. Robot.* **2012**, *33*, 129–142. [CrossRef]
21. Mellinger, D.; Lindsey, Q.; Shomin, M.; Kumar, V. Design, modeling, estimation and control for aerial grasping and manipulation. In Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Francisco, CA, USA, 25–30 September 2011; IEEE Press: New York, NY, USA, 2011; pp. 2668–2673. [CrossRef]
22. Min, B.C.; Hong, J.H.; Matson, E.T. Adaptive Robust Control (ARC) for an altitude control of a quadrotor type UAV carrying an unknown payloads. In Proceedings of the 2011 11th International Conference on Control, Automation and Systems, Gyeonggi-do, Korea, 26–29 October 2011; IEEE Press: New York, NY, USA, 2011; pp. 1147–1151.
23. Ashis, C.K.; Sharma, K.R. Dynamic Modeling and Altitude Control of Parrot Rolling Spider using LQR. In Proceedings of the 2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT), Kannur, India, 5–6 July 2019; IEEE Press: New York, NY, USA, 2019; pp. 1377–1381. [CrossRef]
24. Roy, R.; Islam, M.; Sadman, N.; Mahmud, M.A.P.; Gupta, K.D.; Ahsan, M.M. Review on Comparative Remarks, Performance Evaluation and Improvement Strategies of Quadrotor Controllers. *Technologies* **2021**, *9*, 37. [CrossRef]
25. Okasha, M.; Kralev, J.; Islam, M. Design and Experimental Comparison of PID, LQR and MPC Stabilizing Controllers for Parrot Mambo Mini-Drone. *Aerospace* **2022**, *9*, 298. [CrossRef]
26. Noordin, A.; Basri, M.A.M.; Mohamed, Z.; Lazim, I.M. Position and Attitude Control of Quadrotor MAV Using Sliding Mode Control with Tanh Function. *Lect. Notes Electr. Eng.* **2022**, *900*, 193–204. [CrossRef]
27. Xu, L.; Shao, X.; Zhang, W. USDE-Based Continuous Sliding Mode Control for Quadrotor Attitude Regulation: Method and Application. *IEEE Access* **2021**, *9*, 64153–64164. [CrossRef]
28. Abro, G.E.M.; Zulkifli, S.A.B.M.; Asirvadam, V.S.; Ali, Z.A. Model-free-based single-dimension fuzzy smc design for underactuated quadrotor uav. *Actuators* **2021**, *10*, 191. [CrossRef]
29. Guo, Y.; Luo, L.; Bao, C. Design of a Fixed-Wing UAV Controller Combined Fuzzy Adaptive Method and Sliding Mode Control. *Math. Probl. Eng.* **2022**, *2022*, 2812671. [CrossRef]
30. Nettari, Y.; Labbadi, M.; Kurt, S. ScienceDirect Adaptive robust finite-time tracking control for quadrotor subject to disturbances. *Adv. Space Res.* **2022**, in press. [CrossRef]
31. Hassani, H.; Mansouri, A.; Ahaitouf, A. Backstepping-based supertwisting sliding mode attitude control for a quadrotor aircraft subjected to wind disturbances: Experimental validation. *Int. J. Dyn. Control* **2022**. [CrossRef]

32. Noordin, A.; Basri, M.A.M.; Mohamed, Z.; Lazim, I.M. Adaptive PID Controller Using Sliding Mode Control Approaches for Quadrotor UAV Attitude and Position Stabilization. *Arab. J. Sci. Eng.* **2020**, *46*, 963–981. [CrossRef]
33. Hsu, C.; Chiu, C.; Tsai, J. Auto-tuning PID controller design using a sliding-mode approach for DC servomotors. *Int. J. Intell. Comput. Cybern.* **2011**, *4*, 93–110. [CrossRef]