MASTER

Robust active disturbance rejection control scheme for quadrotor UAVs
experimental prototyping

Zou, J.

Award date:
2018

Link to publication

Disclaimer
This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
Robust Active Disturbance Rejection Control Scheme for Quadrotor UAVs: Experimental Prototyping

Jiachi Zou, Luc Waeijen, Dip Goswami, Roland Toth

Abstract—In this paper, a robust active disturbance rejection control (ADRC) scheme is proposed for quadrotor attitude as well as altitude control for an improved tracking performance and ability for disturbance rejection. To suppress the effects of measurement noise and system delay in a real system, the proposed ADRC structure is optimized taking into account a number practical limitations of the quadrotor plant. To verify the effectiveness of proposed scheme, a Matlab simulator executed in software-in-the-loop (SITL) and a Gazebo simulator executed in hardware-in-the-loop (HITL) are developed. Towards this, the quadrotor dynamics are derived and parameters are identified on the real system. Finally, the simulation results are experimentally validated on the real quadrotor platform. The proposed method shows significant improvements in terms of signal tracking and disturbances rejection.

Index Terms—Active disturbance rejection control (ADRC), quadrotor modeling, robust control under uncertain disturbances, software-in-the-loop (SITL) and hardware-in-the-loop (HITL) quadrotor simulation.

I. INTRODUCTION

QUADROTORS or unmanned aerial vehicles (UAVs) are increasingly playing an important role in fields such as events coverage, transportation, agriculture and asset management. A variety of quadrotor systems have been designed targeting a wide range of application scenarios. To this end, a robust and reliable flight controller is crucial since one control design paradigm often fails to maintain both robustness and stability in different application scenarios.

PID control and its variants are widely used in state-of-the-art flight control systems due to their flexibility and simplicity to implement. Some of the most notable open-source autopilot projects, such as APM, PX4 and Paparazzi, are using PID based controllers as their key control method [1]. In most of the real-life scenarios, PID is sufficient to meet application requirements. However, there are cases where applications demand high performance, e.g., signal tracking and disturbances rejection ability are main considerations, PID often fails to fulfill such high performance demand [2]. Consider the mission scenario which uses a quadrotor to deliver packages. A common requirement is that the quadrotor maintains stability when the payload is employed or dropped and flies without perturbation under disturbances. In this context, with traditional PID control, such sudden disturbance takes long time to be eliminated by the integral action. That is, the aircraft tilts in one side, and the controller gradually brings it back to a target state. This is dangerous and potentially leads to damages and losses.

In the past years, various control approaches are explored to enhance the performance of quadrotors. Some notable control strategies are introduced by [3]–[8]. Most of these approaches depend on the accuracy of model (system dynamics) and in reality, it is hard or too expensive to obtain an accurate model for the given system. Inaccuracy in the models negatively influences the controller’s ability to reject disturbances which makes those technologies hard to be used in many real-life scenarios. To this end, the active disturbance rejection control (ADRC) is considered in recent works due to its ability to estimate the system dynamics and external disturbance without knowing system information. ADRC was originally proposed in 1998 [9], and has been applied in a broad range of domains [10]–[15]. ADRC is proven to achieve superior performance in terms of signal tracking and disturbance rejection simultaneously. ADRC is attractive since it allows for estimation of the total disturbance, which includes system unknown dynamics and external disturbances, in real time. The estimated disturbance is then compensated by a feed-forward component.

A number of recently reported simulation results indicate that ADRC achieves encouraging result in quadrotor control [16]–[20]. However, only few works have been done on the implementation on real-life quadrotors. There are two key challenges to be addressed before applying such a method to a quadrotor system. The first challenge is to deal with the measurement noise from inertial measurement units (IMUs), which is subjected to motor vibration and device inaccuracy. This is relevant to ADRC behavior since the extended state observer (ESO) is sensitive to noise [21] and the amplified noise leads to a chattering behavior. The second challenge is handling the time delay (mainly) caused by internal digital filters and actuator dynamics (e.g., the motor speed can not be changed instantly). Since the stability margin of ADRC dramatically decreases when delay increases [22], it is crucial to address above two challenges to avoid unexpected behavior, such as chattering, energy loss, or even device damage, while applying ADRC scheme in real-life systems.

In this paper, a novel robust ADRC scheme for quadrotor attitude and altitude control is designed, which provides reme-
dies to cope with the aforementioned challenges. First, The
proposed method is verified in the simulation environments
(Matlab and Gazebo). Next, the proposed algorithm is
implemented on a real quadrotor platform. The performance
of the proposed ADRC scheme and PID controller are com-
pared and evaluated both in simulations and experiments.
The quadrotor dynamics is derived and model parameters are
identified to obtain reliable simulation results. The simulation
and experimental results confirm that the proposed method
significantly improves quadrotor performance in terms of
signal tracking and disturbance rejection ability.

The rest of this paper is organized as follows. Section
II illustrates the coordinate system and dynamic model of
quadrotor. In Section III, the proposed scheme for attitude
and altitude control are detailed. In Section IV, the simulation
environments are introduced and the simulation result are
presented. The experimental results are illustrated in Section
V. Finally, the paper is concluded in Section VI.

II. QUADROTOR SYSTEM AND MODEL

In this section, the quadrotor dynamics is derived and the
corresponding model parameters for the experimental quadro-
tor platform are identified. The dynamics and parameters are
further used in the simulators.

A. Quadrotor Platform

The quadrotor platform used in this work is shown in Fig. 1.
Four brush-less motors are mounted in the end of the arm
with "X" airframe configuration. The flight control hardware
uses the Pixhawk [23], while a custom-designed flight control
system is running on it.

B. Coordinate Systems

We consider two coordinate systems as shown in Fig.2,
- The inertial frame: with a fixed orientation and position
that comply with the earth frame, also known as “world”
coordinates.
- The body frame: which represents the orientation and
position of the body of the quadrotor.

Both systems use north-east-down (NED) notation. The inertia
frame coordinates are denoted by \( {^b}v \) while coordinates in
the body frame are denoted by \( {^v}b \). The rotational matrices
\( R_{b\rightarrow i}(q) \) and \( R_{i\rightarrow b}(q) \) are used for coordinate transformations
between the frames. The rotational matrix \( R_{b\rightarrow i}(q) \) from body
to inertia frame is given by,

\[
R_{b\rightarrow i}(q) = \begin{bmatrix}
1 - 2y^2 - 2z^2 & 2(xy - wz) & 2(xz + wy) \\
2(xy + wz) & 1 - 2x^2 - 2z^2 & 2(yz - wx) \\
2(xz - wy) & 2(yz + wx) & 1 - 2x^2 - 2y^2
\end{bmatrix}
\] (1)

where \( q = [w, x, y, z] \) is unit quaternion [24]. On the other
hand, the rotational matrix \( R_{i\rightarrow b}(q) \) from inertia to body frame
is given by,

\[
R_{i\rightarrow b}(q) = R_{b\rightarrow i}(q^{-1}) = R_{b\rightarrow i}(q)^T
\] (2)

which is essentially transposed of \( R_{b\rightarrow i}(q) \).

C. Quadrotor Modeling

We start with the following two forces acting on the
airframe:

1) Gravity: The gravitational force acts along the z-axis of
the inertial frame. We transform it to the body frame by
the following.

\[
bF_g = R_{i\rightarrow b}(q) \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}
\] (3)

2) Thrust: The thrust generated by the motors is the driving
force for the quadrotor. All motors are mounted upwards
on the end of the arm. Therefore they only exert a force
in the vertical direction. The force exerted by motor i
is given by

\[
bF_{t,i} = C_T \rho A r^2 \Omega_i^2
\] (4)

where \( C_T \) is the thrust coefficient of the motor, \( \rho \) is the
density of air, \( A \) is the rotor disc area, \( r \) is the rotor
radius, and \( \Omega \) is the angular velocity of the rotor. The
equation (4) is rewritten as follows,

\[
bF_{t,i} = c_T \Omega_i^2
\] (5)
where \( c_T = C_T \rho A r^2 \). Next, the total thrust by all four motors is given by,
\[
\begin{align*}
\mathbf{F}_{t\mathbb{T}} &= -\sum_{i=1}^{4} \mathbf{F}_{t,i} = -\sum_{i=1}^{4} c_T \mathbf{\Omega}^2 \\
\end{align*}
\]
where\( \mathbf{\Omega} \) is the i-th motor speed. The negative sign implies an upward thrust.

Further, two types of torques act on the airframe.

1) Torque generated by motors: The torque generated by the i-th motor about the yaw axis is given by,
\[
\tau_{\psi} = c_Q \mathbf{\Omega}^2
\]
where \( c_Q \) is the torque constant. Using Eq. (5) for the four motors, the torque along each axis for an “X” configuration of the airframe can be expressed as
\[
\begin{align*}
\mathbf{\tau} &= \begin{bmatrix} \tau_{\phi} \\ \tau_{\theta} \\ \tau_{\psi} \end{bmatrix} \\
\end{align*}
\]
\[
\begin{bmatrix}
-Lc_T & -Lc_T & Lc_T & -Lc_T \\
Lc_T & -Lc_T & Lc_T & -Lc_T \\
c_Q & -c_Q & -c_Q & -c_Q \\
\end{bmatrix}
\]
where \( L = l \sin(45^\circ) \), \( l \) is the arm length from quadrotor hub center to a motor. The torque matrix for the “+” configuration can be derived similarly.

2) Gyroscopic moment: The gyroscopic moment is caused by so called gyroscopic precession phenomenon, which changes the orientation of the rotation axis of a rotating body. The gyroscopic moments are calculated as follows
\[
\mathbf{M}_{gyro} = \mathbf{I}_m \mathbf{\tilde{e}}_z \times \begin{bmatrix} \mathbf{\omega}_\phi \\ \mathbf{\omega}_\theta \\ \mathbf{\omega}_\psi \end{bmatrix} \mathbf{\Omega} \mathbf{I} (10)
\]
where \( \mathbf{\tilde{e}}_z = [0 \ 0 \ 1]^T \), \( \mathbf{\Omega} = -\mathbf{\Omega}_1 - \mathbf{\Omega}_2 + \mathbf{\Omega}_3 + \mathbf{\Omega}_4 \), and \( \mathbf{I}_m \) is the inertia of a propeller.

The total moment is a combination of the above two leads to,
\[
\mathbf{M} = \mathbf{b}T + \mathbf{b}M_{gyro} (11)
\]

Quadrotor attitude dynamics: Based on aforementioned equations, the attitude dynamics is derived as
\[
\mathbf{\dot{\omega}} = \mathbf{I}^{-1} \left( \mathbf{b}M - \mathbf{b} \times (\mathbf{I} \mathbf{\omega}) \right) (12)
\]
where \( \mathbf{b} \mathbf{\omega} = [\mathbf{\omega}_\phi \ \mathbf{\omega}_\theta \ \mathbf{\omega}_\psi]^T \) and \( \mathbf{I} \) is the inertia of the quadrotor, given by
\[
\begin{bmatrix}
I_{xx} & 0 & 0 \\
0 & I_{yy} & 0 \\
0 & 0 & I_{zz} \\
\end{bmatrix}
\]

Quadrotor position dynamics: In the inertial frame, the acceleration of the quadrotor is due to thrust, gravity, and linear friction. Since we assume the quadrotor flying at a slow speed, the linear friction can be neglected. Moreover, when quadrotor is flying with a circular path, the centrifugal acceleration is generated. To this end, the linear motion equation is given by,
\[
\mathbf{\ddot{x}} = \mathbf{g} + \mathbf{R}_{\mathbb{B}\rightarrow\mathbb{I}} (\mathbf{R}_{\mathbb{B}\rightarrow\mathbb{I}} (\mathbf{F}_{t\mathbb{B}} - \mathbf{b} \times \mathbf{V})) (13)
\]
where \( \mathbf{g} \) represents the quadrotor position in the inertial frame, \( \mathbf{g} \) is the gravity constant, and \( \mathbf{b} \mathbf{V} = [v_x \ v_y \ v_z]^T \) is the linear velocity of quadrotor in the body frame.

The attitude of quadrotor is expressed in quaternions. The following equation is employed to update quaternions.
\[
\dot{\mathbf{q}} = \frac{1}{2} \mathbf{W} (\mathbf{\omega} \mathbf{q}) (14)
\]

D. Rotor Dynamics

The motor is driven by electronic speed controllers (ESC), which converts the PWM input signal (throttle) into multiple out-of-phase voltage outputs to spin the motor. Therefore, ESCs do affect how the rotor dynamics are represented. The measured data, which is shown in Fig. 3, reveals the relationship between the throttle and motor speed. For the i-th motor, the motor speed is given by
\[
\Omega_i = k \Gamma_i + p (15)
\]
where \( \Gamma_i \) is the throttle command for i-th motor with the range of \([0, 1]\), \( k \) is the slope and \( p \) is the offset of the linear best-fit approximation of the throttle and motor speed.

Motor speed can not be changed instantly, which essentially introduces a delay before the motor reaches the steady-state speed. Therefore, it could be reasonably modelled as the first-order low-pass filter. Thus, the system can be described in Laplace domain as:
\[
\Omega(s) = \frac{\Omega_0(s)}{1 + s \tau} (16)
\]
where \( s \) is the complex frequency, \( \tau \) is the filter time constant, \( \Omega_0 \) is the desired motor speed and \( \Omega(s) \) is the actual motor speed experienced with delay. The time constant \( \tau \) is determined by identifying the system rise-time \( t_r \), and it is calculated by using the equation \( \tau = \frac{t_r}{2} \) [25]. Usually, the time constant for motor acceleration and deceleration are not the same. Therefore, we denote up and down time constants by \( \tau_{up} \) and \( \tau_{down} \) respectively.

E. Parameters Identification

The identification of the real model parameters is crucial for model accuracy and the reliability of the simulation results. To measure the motor parameters, a motor with propeller is mounted with a weight (e.g., a heavy stone). By giving different throttle command, the generated thrust leads to the change in the weight, which was measured by a digital scale, and the motor speed is measured using a tachometer. The measured result and its polynomial curve fitting are shown in Fig. 3. Finally, the experimental quadrotor platform’s parameters are listed in Table I.
Fig. 3: Polynomial curve fitting for measured motor speed and generated thrust versus throttle command $\Gamma$

TABLE I: measured parameters for experimental quadrotor platform

<table>
<thead>
<tr>
<th>parameter</th>
<th>value</th>
<th>unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m$</td>
<td>0.886</td>
<td>kg</td>
</tr>
<tr>
<td>$I_{xx}$</td>
<td>0.016</td>
<td>kg·m$^2$</td>
</tr>
<tr>
<td>$I_{yy}$</td>
<td>0.016</td>
<td>kg·m$^2$</td>
</tr>
<tr>
<td>$I_{zz}$</td>
<td>0.0274</td>
<td>kg·m$^2$</td>
</tr>
<tr>
<td>$c_T$</td>
<td>1.239e-5</td>
<td>N/(rad/s)$^2$</td>
</tr>
<tr>
<td>$c_Q$</td>
<td>1.982e-7</td>
<td>N·m/(rad/s)$^2$</td>
</tr>
<tr>
<td>$\tau_{wp}$</td>
<td>0.0125</td>
<td>s</td>
</tr>
<tr>
<td>$\tau_{down}$</td>
<td>0.025</td>
<td>s</td>
</tr>
<tr>
<td>$k$</td>
<td>718.078</td>
<td>rad/s</td>
</tr>
<tr>
<td>$p$</td>
<td>88.448</td>
<td>rad/s</td>
</tr>
</tbody>
</table>

III. ADRC CONTROLLER DESIGN

In this section, we first introduce the basic components of typical ADRC. Next, the proposed ADRC schemes for quadrotor attitude and altitude control are described.

A. Basic ADRC Components

Consider the control of a second-order system,

$$
\begin{cases}
\dot{x}_1 = x_2 \\
\dot{x}_2 = f(x_1, x_2, w(t), t) + bu \\
y = x_1
\end{cases}
$$

(17)

where $x_i$ are system states, $y$ is the system output, $w(t)$ is external disturbances. A typical ADRC scheme for this system takes the form as shown in Fig.4. It consists of three main components, namely the tracking differentiator (TD), the extended state observer (ESO) and the nonlinear state error feedback (NLSEF), which will be explained in the remainder of this section.

1) Tracking Differentiator: For a given reference signal, the TD generates a transient trajectory that the system can reasonably follow and extracts the derivative of the generated trajectory. The TD is constructed as follows.

$$
\begin{cases}
\dot{v}_1 = hv_2 \\
\dot{v}_2 = h \cdot fhan(v_1 - v, v_2, r_0, h_0)
\end{cases}
$$

(18)

where $h$ is sampling time, $v$ is the input reference signal, $r_0$ and $h_0$ are tunable parameters, which can be used to adjust the shape of transient trajectory, and $fhan(v_1, v_2, r_0, h_0)$ is defined as follows

$$
\begin{align*}
d &= h_0^2 r_0, \\
an &= h_0 v_2, \\
v &= v_1 + a_0 \\
a_1 &= \sqrt{d(d + 8|y|)} \\
d &= a_0 + \text{sign}(y)(a_1 - d)/2 \\
s_y &= \left( \text{sign}(y + d) - \text{sign}(y - d) \right)/2 \\
as &= \left( \text{sign}(a + d) - \text{sign}(a - d) \right)/2 \\
fhan &= -r_0(\frac{a}{d} - \text{sign}(a))s_a - r_0\text{sign}(a)
\end{align*}
$$

Note that (19) is a time-optimal solution which guarantees a fast convergence of $v_1$ to $v$ without overshoot [9].

2) Extended State Observer: The ESO is used to estimate the system states and the total disturbances. For the system (17), let $F(t) = f(x_1, x_2, w(t), t)$ being the "total disturbance", which is a multi-variable function including the system states $x_1, x_2$, the external disturbances $w(t)$, and the time $t$. If we are able to estimate the total disturbances $F(t)$, system (17) is transformed into a canonical integral-chain form [9], which can be easily controlled, by selecting the control signal with $u = (u_0 - F(t))/b$. In order to estimate the total disturbances $F(t)$, we treat $F(t)$ as an extended state $x_3$. Then the original system (17) converts into

$$
\begin{cases}
\dot{x}_1 = x_2 \\
\dot{x}_2 = x_3 + bu \\
\dot{x}_3 = F(t) \\
y = x_1
\end{cases}
$$

(20)

Based on the extended states system, the ESO can be constructed [9].

The ESO has many parameters, which makes it difficult to be applied in practice. In order to address this problem, Gao proposed the linear extended state observer (LESO) [26], in which all parameters are decided by observer bandwidth and workload of tuning parameters is dramatically reduced. The LESO equations for system (44) are given by

$$
\begin{align*}
e &= z_1 - y \\
\dot{z}_1 &= z_2 - \beta_1 e \\
\dot{z}_2 &= z_3 + b_0 u - \beta_2 e \\
\dot{z}_3 &= -\beta_3 e
\end{align*}
$$

(21)
where $z_1$, $z_2$ and $z_3$ are estimated states correspond to $x_1$, $x_2$ and $x_3$ respectively, $h_0$ is the approximation of the control input gain $b$, considering that $b$ is usually unknown in real system. $\beta_1$, $\beta_2$ and $\beta_3$ can be chosen based on bandwidth-parameterization [27],

$$
\beta_1 = 3\omega_o, \quad \beta_2 = 3\omega_o^2, \quad \beta_3 = \omega_o^3 \tag{22}
$$

where $\omega_o$ is the bandwidth of the observer. In most cases, a higher observer bandwidth generally gives better state and disturbance estimations but also makes the observer more sensitive to noise. Hence, $\omega_o$ should be selected by considering the trade-off between estimation accuracy and noise tolerance.

3) Nonlinear State Error Feedback: The NLSEF introduces a way to integrate the nonlinear function $f_{han}(e_1, ce_2, r_1, h_1)$ with feedback control, which sometimes achieves dramatically better performance than linear control [9]. The form of NLSEF is given by,

$$
e_1 = v_1 - z_1, \quad e_2 = v_2 - z_2
$$

$$
u_0 = -f_{han}(e_1, ce_2, r_1, h_1) \tag{23}
$$

where $r_1$ determines the upper and lower bound of the output, $h_1$ determines the width of the middle continuous smooth approximation area and $c$ is the damping factor.

B. Attitude ADRC

A high-precession attitude control is essential for the success of the quadrotor control. The control quality of attitude determines the ultimate flight performance to a large extent. The Euler angles are used to denote the quadrotor’s attitude, which are roll, pitch and yaw $[\phi, \theta, \psi]$. Assuming the quadrotor has a symmetrical structure, then the roll and pitch share the same control structure. Meanwhile, considering the heading control is less significant than horizontal control in practice, only the control scheme of pitch is discussed in this paper. In particular, the system dynamics for pitch can be described as

$$
\begin{cases}
\dot{\theta} = \omega \\
\dot{\omega} = f(\epsilon, \omega, \omega(t), t) + bu \\
y = \theta
\end{cases} \tag{24}
$$

where $\epsilon = [\phi \theta \psi]^T$, $\omega = [\omega_\phi \omega_\theta \omega_\psi]^T$. This is a second-order system and the controlled object is angle $\theta$. Naturally, the conventional second-order ADRC described in the previous section can be considered for controlling. However, it is not suitable to directly apply the conventional ADRC in this scenario because of the following limitations.

1) To apply the second-order ADRC, precise measurement of angle $\theta$ is required, but there is no available sensor or it is too expensive to directly measure the angle. The estimated angle $\hat{\theta}$ could be obtained by sensor fusion method [28], [29], but this generally introduces extra delay and inaccuracy. In this way, the performance of the ESO would be tightly bound to the timely and accurate estimation of the angle.

2) The second-order ADRC introduces extra delay in the system, since a higher order ESO incurs a larger phase lag in the ESO [30].

3) The conventional ADRC scheme lacks filtering components. As the ESO is noise sensitive, a noisy signal will dramatically damage the observer’s performance.

4) The system delay, mainly caused by internal digital filters and actuator dynamics, is not explicitly dealt with by the conventional ADRC framework. The existence of the delay negatively affects the performance of ADRC when applied to a real-life system.

5) In the conventional ADRC theory, the total disturbance is estimated and fully compensated by feed-forward in which the compensation intensity can not be adjusted. Under the existence of noise and delay, the full disturbance compensation would be particularly aggressive and, hence, over-actuate the system. As shown in Fig. 13c.

In order to enhance the performance and robustness of the ADRC scheme, all these limitations should be considered and dealt with. To this end, a robust ADRC scheme for attitude control is designed (Fig. 5). Next, the corresponding techniques to handle aforementioned limitations will be detailed.

Firstly, to deal with limitations (1) and (2), a reduced-order LESCO is employed. In this way, the system output becomes $y = \omega_\theta$, which can be directly measured by gyroscope. The phase lag of observer is reduced as well, since the lower order observer incurs smaller phase lag. Moreover, in order to simply the process of parameter adjustment, the linear ESO is used instead of nonlinear one. The construction of LESCO for the reduced system is given as,

$$
\begin{align*}
e_1 &= z_1 - y \\
\dot{z}_1 &= z_2 + b_0 u - \beta_1 e_1 \\
\dot{z}_2 &= -\beta_2 e_1
\end{align*} \tag{25}
$$

where $z_1$ is estimated angular velocity and $z_2$ for estimated total disturbances. Referred to [27], $\beta_1$ and $\beta_2$ can be decided by an given observer bandwidth $\omega_o$, which is $\beta_1 = 2\omega_o$, $\beta_2 = \omega_o^2$. The remaining undetermined parameter is $b_0$, which is the approximation of the control input gain $b$. In practice, if system information is unknown, $b_0$ can be regarded as a tunable parameter. In our work, the definition of $b_0$ for system (24) is derived. First considering the following equation for control mixing,

$$
\Gamma = \begin{bmatrix}
\Gamma_1 \\
\Gamma_2 \\
\Gamma_3 \\
\Gamma_4
\end{bmatrix} = \begin{bmatrix}
-1 & 1 & 1 \\
1 & -1 & 1 \\
1 & 1 & -1 \\
-1 & -1 & -1
\end{bmatrix} \begin{bmatrix}
u_\phi \\
u_\theta \\
u_\psi
\end{bmatrix} + \Gamma_{base} \tag{26}
$$

where $\Gamma_{base}$ is base throttle, which is control input generated by the altitude controller, $\Gamma_i$ is the $i$-th motor throttle, $[u_\phi, u_\theta, u_\psi]^T$ is the control input for roll, pitch and yaw. Assume $u_\phi = u_\psi = 0$ and let $u = u_\theta$, yields

$$
\Gamma_i = \Gamma_{base} + (-1)^{i+1} u \tag{27}
$$

On the basis of (7) and (15), the generated angular acceleration
of the pitch is given by,

$$\omega_y' = \sum_{i=1}^{4} (-1)^{i+1} c_T \Omega_i^2 l \cdot \sin(\pi/4)$$

$$= \sum_{i=1}^{4} (-1)^{i+1} c_T (k \Gamma_i + p)^2 l \cdot \sin(\pi/4)$$

$$= \frac{8k c_T l \cdot \sin(\pi/4)(k \cdot \Gamma_{base} + p)}{I_{gy}}$$

Comparing with (24), yields

$$b_0 = \frac{8k c_T l \cdot \sin(\pi/4)(k \cdot \Gamma_{base} + p)}{I_{gy}} \quad (29)$$

Obviously, \(b_0\) only relates to base throttle \(\Gamma_{base}\), while other parameters are constant.

Secondly, to deal with limitation (3), a low-pass filter is added into the feedback loop to attenuate the measurement noise, before the signal enters the LEO. In this respect, a third-order butter-worth filter is designed to filter the gyroscopic signal. Alternatively, some other methods are introduced in [31], [32] to reduce the measurement noise in observer, however, for our use case the butter-worth filter is sufficient.

Thirdly, to solve limitation (4), a time delay block [22] is deployed to postpone the control signal \(u\) before it goes into the LEO. Since the input of the LEO is already delayed due to the filter, this will synchronize the signal that goes into the observer and provide meaningful estimations. The delay block is defined as

$$u' = u(t - \tau_d) \quad (30)$$

where \(\tau_d\) is delay time constant. In practice, \(\tau_d\) includes the time delay of the filter as well as the communication delay of the IMU. As expressed in (16), the rotor dynamics can be described as a first-order system, which essentially introduces a delay to the dynamical system and thus potentially causes detrimental effects when not accounted for. To eliminate the actuator delay, the following approximation in Laplace domain needs to be satisfied,

$$\frac{\Omega(s)}{\Gamma + s \tau} \approx \frac{\Omega(s)}{\Omega(s) + \tau s \Omega(s)} \quad (31)$$

Substituting (15) in (31) and performing an inverse Laplace-transform, we obtain,

$$\tilde{\Gamma} = \frac{\Omega - p}{k} \approx \frac{\Omega + \tau \tilde{\Omega} - p}{k} = \Gamma + \tau \tilde{\Gamma} \quad (32)$$

where \(\Gamma = [\Gamma_1 \ \Gamma_2 \ \Gamma_3 \ \Gamma_4]^T\) is the target throttle command, \(\tilde{\Gamma} = [\tilde{\Gamma}_1 \ \tilde{\Gamma}_2 \ \tilde{\Gamma}_3 \ \tilde{\Gamma}_4]^T\) is the actual throttle command sent to ESC. The compensator block implements equation (32), in which four separate TDs are utilized to extract the derivative part \(\dot{\tilde{\Gamma}}\).

Finally, to address limitation (5), a scale block \(\gamma\) is added before the feed-forward process to adjust the disturbance compensation intensity. \(\gamma\) is in the range of \([0, 1]\), where 0 represents no disturbance compensation and 1 represents full disturbance compensation. The added scale block should not influence the observer’s normal behavior, that is to say, the steady-state estimated total disturbance remains consistent while the scale block is employed. To prove the added scale block adheres to this condition, the theorem 1 is given below.

**Theorem 1.** The steady-state estimated total disturbance is independent of \(\gamma\), which means \(\gamma\) does not influence the final estimated total disturbance.

**Proof.** Consider the following system

$$\begin{cases}
\dot{x}_1 = x_2 + bu \\
\dot{x}_2 = \dot{w}(t)
\end{cases} \quad (33)$$

where \(w(t)\) is the actual total disturbances acting on the system. The LEO can be constructed using the same form as (25) to estimate the system states and the total disturbance. Then the error system between (33) and (25) can be written as

$$\begin{cases}
e_1 = z_1 - x_1, \ e_2 = z_2 - x_2 \\
\dot{e}_1 = e_2 + (b - b_0)u - \beta_1 e_1 \\
\dot{e}_2 = -\dot{w}(t) - \beta_2 e_1
\end{cases} \quad (34)$$

where \(z_2\) is the estimated total disturbance, and \(x_2\) is the actual total disturbance. When the error system (34) reaches steady state, \(e_1 = e_2 = 0\) holds, then we have

$$e_1 = -\frac{\dot{w}(t)}{\beta_2} = -\frac{\dot{x}_2}{\beta_2}, \ e_2 = (b - b_0)u + \beta_1 e_1 \quad (35)$$

Selecting the control law as

$$u = (u_0 - \gamma z_2)/b_0 \quad (36)$$

and substituting (36) into (35) yields

$$e_2 = (b - b_0)u + \beta_1 e_1 = \frac{b}{b_0} - 1)(u_0 - \gamma z_2) + \beta_1 e_1 \quad (37)$$

![Fig. 5: The structure of proposed ADRC scheme for pitch control, where gyro is gyroscopic data measured by gyroscope.](image-url)
Again, since the system (33) is in the steady state, \( x_2 + bu = 0 \) holds. Combining with (34) (37), we have
\[
z_2 - x_2 = z_2 + bu \\
= z_2 + \frac{b}{b_0}(u_0 - \gamma z_2) \\
= \left(1 - \frac{b}{b_0}\right)z_2 + \frac{b}{b_0}\beta_1 e_1 \\
\]
which implies
\[
u_0 = z_2(\gamma - 1) + \beta_1 e_1 \\
\tag{39}
\]
Substituting (39) into (38), yields
\[
z_2 - x_2 = \left(1 - \frac{b}{b_0}\right)z_2 + \frac{b}{b_0}\beta_1 e_1 \\
\tag{40}
\]
Finally, we have
\[
z_2 = \frac{b_0}{b}x_2 - \frac{\beta_1}{\beta_2}\hat{z}_2 \\
\tag{41}
\]
where \( z_2 \) is unrelated to \( \gamma \), thus proving theorem 1.

ADRC can now be used for quadrotor attitude control, but it’s performance can even be enhanced further by utilizing a cascade control structure. The scheme that correctly employs a cascade architecture respond more effectively to disturbances because the inner loop is both closer to the source of the disturbance, and faster than the outer loop. The proportional controller (P controller) is widely used to serve as outer cascade architecture respond more effectively to disturbances. The scheme that correctly employs a controller output, \( \hat{v} \), which implies
\[
u_0 = z_2(\gamma - 1) + \beta_1 e_1 \\
\tag{39}
\]
Substituting (39) into (38), yields
\[
z_2 - x_2 = \left(1 - \frac{b}{b_0}\right)z_2 + \frac{b}{b_0}\beta_1 e_1 \\
\tag{40}
\]
Finally, we have
\[
z_2 = \frac{b_0}{b}x_2 - \frac{\beta_1}{\beta_2}\hat{z}_2 \\
\tag{41}
\]
where \( z_2 \) is unrelated to \( \gamma \), thus proving theorem 1.

The system dynamics for altitude control system can be described as
\[
\begin{align*}
\dot{z} &= v \\
\dot{v} &= f(z,v,z,w(t),t) + bu^2, \quad u \in [0, 1] \\
y &= z \\
\end{align*}
\tag{44}
\]
Notice that control signal \( u \) is quadratic to \( \dot{v} \) according to (5) (15). The equations for LESO and control law are modified accordingly. The definition of LESO for altitude control is then given by
\[
\begin{align*}
e &= z_1 - y \\
\dot{z}_1 &= z_2 + b_0u^2 + g - \beta_1 e \\
\dot{z}_2 &= -\beta_2 e \\
\end{align*}
\tag{45}
\]
where \( g \) is gravitational constant. The gravity is always acting on the \( z \) axis of inertia frame, which should not be included in the estimated total disturbance. Hence, \( g \) is added to the expression of \( \dot{z}_1 \). This ensures the gravity is not a component in the final estimated disturbance. The control law then becomes
\[
u_0 = -\text{fhan}(e_1, ce_2, r_1, h_1) + \Gamma_h, \quad u_0 \in [0, 1] \\
u = \sqrt{\left(u_0 - \frac{z_2}{b_0}\right)^2 + \left(u_0 - \frac{z_2}{b_0}\right)^2}, \quad \left(u_0 - \frac{z_2}{b_0}\right) \in [0, 1] \\
\tag{46}
\]
where \( \Gamma_h \) is the hover throttle, and \( u_0 \) and \( u_0 - \frac{z_2}{b_0} \) are clipped to the range of \([0,1]\). In summary, the structure of proposed ADRC scheme for altitude control is shown in Fig. 6.

IV. SIMULATION RESULT

In order to verify the performance of the proposed method, a MATLAB quadrotor simulator is developed, which executes in software-in-the-loop (SITL) mode. Moreover, a Gazebo quadrotor simulator is built to simulate in hardware-in-the-loop (HIL) mode.

A. MATLAB Simulation

Based on the kinetic equations (10), (12) and measured model parameters in Table I, a MATLAB quadrotor simulator is developed. The simulator implements the proposed attitude and altitude ADRC scheme. The PID controller is also implemented for comparison purpose. The GUI of the simulator is shown in Fig. 7.

The quadrotor simulator uses exactly the same parameters as the real quadrotor system. This SITL approach offers great flexibility for testing the proposed ADRC scheme before the deployment on the real aircraft and provides a way to observe the behavior of proposed method. The signal tracking and disturbance rejection ability have been evaluated in the simulator for the proposed ADRC and PID control.

The disturbance rejection simulation results are shown in Fig. 8. Given a constant or various (sinusoidal signal) disturbance, oscillation of the proposed method is much smaller than the cascade PID. Meanwhile, the angle convergences to origin rapidly after disturbance is employed or removed by using the proposed method. It also can be observed that by decreasing \( \gamma \) from 1 to 0.5, the disturbance-rejection ability becomes weaker but the observer is still able to estimate the total disturbance correctly.
Fig. 6: The structure of proposed ADRC scheme for altitude control, where \( \hat{v}_z \) is estimated velocity in z-axis and \( \hat{z} \) is estimated altitude.

Fig. 7: Matlab quadrotor simulator UI, the slides located in right side are used to set target attitude and target altitude.

Fig. 9 depicts the signal tracking simulation results. The result shows that given a step reference signal \( v \), the actual angle follows the generated transient trajectory \( v_1 \) rapidly with negligible overshoot by using the proposed method, while a larger overshoot and a slower tracking speed can be observed by using the cascade PID control. According to the result, the tracking speed is able to be adjusted by tuning \( r_2 \) and \( h_2 \) can be tuned to tailor the transient trajectory, that is, to make the transient trajectory smoother or sharper.

Regarding to altitude control, the simulation results are shown in Fig. 10. In time=3s, a constant disturbance is added, and in time=6s the disturbance is removed. The result shows that the proposed method achieves significant improvements for signal tracking and disturbance rejection performance. We can observe that at the point when disturbance is added or removed, the PID controlled system experiences a large altitude oscillation. For the proposed method, the oscillation is negligible even when the disturbance is active. The proposed method outperforms the PID control in term of tracking performance. For PID control, the actual altitude can not follow the reference signal accurately, and a large overshoot can be observed. In contrast, the actual altitude closely follows the transient trajectory with a negligible overshoot when the proposed method is used.

### B. Gazebo Hardware-in-the-loop Simulation

Before conducting the experiments on the real quadrotor platform, the proposed ADRC scheme of attitude control is tested in Gazebo simulator, which executes in HITL mode. The HITL simulation provides a more realistic simulation result since the autopilot software executes on the real hardware and, therefore, reduces the possibility to cause damage and loss on the real platform.

The proposed ADRC attitude scheme is coded in the C programming language and executed on a custom-designed flight control system, and related modules are developed to support the HITL simulation. The Gazebo quadrotor simulator is developed based on the open-source project [33], of which the quadrotor model is replaced with the measured parameters (Table. I). In this way, the simulation result can reflect the actual situation when applying to a real system.

Fig. 11 shows the communication between the Robotic Operating System (ROS) nodes and related topics while the Gazebo simulator is running, in which the ellipse represents the node and the block represents the topic. The flight control system executes on the Pixhawk, which communicates with the mavros node via USB cable. Various sensor data, specifically, accelerometer, magnetometer, gyroscope, barometer and gps, are periodically published by the gazebo node. After receiving the sensor data, the flight control system calculates the desired throttle command, which is then sent to the Gazebo simulator. According to the Gazebo simulation result, the quadrotor controlled by the proposed attitude ADRC scheme flies stably, and a wind is applied as disturbance. The screenshot of the Gazebo simulator is shown in Fig.12.

Fig. 12: Screenshot of Gazebo hardware-in-the-loop quadrotor simulation.
V. EXPERIMENTAL RESULT

In this section, we describes attitude control experimental results on a real quadrotor, as shown in Fig. 1. To simulate the influence of adding and removing the disturbance, a payload of 0.25kg is employed, which is mounted on the end of an arm. The flight control system is not informed of when payload is added or dropped.

A. Disturbance Rejection Performance Analysis

To verify the validity of the proposed method under unknown disturbances, the experiments are conducted and the results are shown in Fig. 13. We notice that the observer correctly estimates the total disturbance and the cascade PID control suffers extremely large angle oscillation while payload is deployed, with a maximal $23^\circ$ oscillation when payload added, and $16^\circ$ oscillation when payload dropped. In contrast, the proposed ADRC scheme achieves significant improvements, as maximal oscillation for payload adding and dropping is reduced to $6^\circ$ and $4^\circ$ respectively. By increasing the scale factor $\gamma$ from 0.5 to 0.8, the system starts to fluctuate when disturbance is deployed, which shows that too aggressive disturbance compensation would lead to system instability and, hence, reveals the importance of the scale block.

B. Tracking Performance Analysis

The signal tracking experiments are conducted by giving a series of step reference signals. The results are depicted in Fig. 14. A large angle overshoot, with a maximal overshoot of $12^\circ$, can be observed, and the actual angular velocity can not track the reference signal closely by using the cascade PID control. On the other hand, by using the proposed method, the maximal angle overshoot is reduced to $2^\circ$, and the actual angular velocity tracks the reference signal accurately.

To further explore the potential of the proposed method, the signal tracking experiments with payload deployed are also conducted, and the results are shown in Fig. 15. For PID control, there exists tracking error for a long time before getting eliminated by the integral action. For the proposed method, the static error disappears immediately, and the actual angle as well as the actual angular velocity track the reference signal closely and convergence quicker compared to the cascade PID control, under the condition that when disturbances are acting.

VI. CONCLUSION

In this paper, a robust ADRC scheme has been proposed for quadrotor attitude and altitude control, which is subjected to uncertain disturbances. By optimizing the structure of the
ADRC framework and designing corresponding components to deal with measurement noise and system delay, the impacts of these practical limitations are well suppressed. A scale block is designed to adjust the intensity of disturbance compensation, which is effective to avoid system over-actuation, and therefore increase the system robustness significantly. Moreover, A MATLAB simulator executed in SITL and a Gazebo simulator executed in HITL mode are developed to verify the proposed method’s performance and to test before deploying on the real platform. According to the simulation results, the proposed method outperforms the conventional PID control, by which both accurate signal tracking and active disturbance rejection ability can be achieved simultaneously. Finally, the proposed ADRC scheme is validated in a real quadrotor system. Based on the experimental results, the expected improvements are achieved, which proves the effectiveness of the proposed method. The test video can be accessed via https://www.youtube.com/watch?v=77-nF-qppA&t=113s.

ACKNOWLEDGMENT
In the end, I would like to take this opportunity to express my gratitude. I would like to thank all the people that help me and support me to complete the project. First and foremost, I would like to thank dr. Dip Goswami. Thank you for giving me this opportunity to work on this project and give me a lot of freedom to investigate in this area, especially for your guidance and patience, which lead me to finish the project as well as the paper. And many thanks go to ir. Luc Waeijen, who assisted me to finish internship project, and continually support with my graduation project. You patience and proficiency help me to overcome difficulties, and you always help a lot by providing useful advices in many ways. Also, I want to thank dr.ir. Rolan Toth. Thank you to provide me the experimental devices and extremely useful feedback. Furthermore, I would like to thank my parent and family, who always give me unconditional support. Finally, I would like to thank my girlfriend, who accompany me during the whole project and inspire me to go ahead.

APPENDIX

A. StarryPilot
The StarryPilot is the autopilot software that we used in this work, which is a completely self-developed flight control
Fig. 10: Altitude signal tracking and disturbance rejection. (a) PID (b) Proposed ADRC; Disturbance rejection (c) PID (d) Proposed ADRC.

B. Gazebo Simulator Usage Guide

The Gazebo is integrated in the ROS, hence, ROS should be installed correctly before using the Gazebo quadrotor simulator. The simulator is tested in ROS kinetic, and for other ROS distributions, it is not well tested.

To compile the simulator, first copy all package file into the workspace of ROS, and then execute the command,

```
$ catkin build
```

To launch the Gazebo quadrotor simulator, the following commands are used.

1. Launch mavros node.

2. Launch Gazebo quadrotor simulator.

```
$ roslaunch mavros starry.launch
$ roslaunch rotor_gazebo starry_hil.launch
```

C. Source Code

The source code of this work, which consists of ADRC library, Matlab simulator, Gazebo simulator and StarryPilot autopilot software are available in Github. The link of each project are listed below:

- ADRC Library: https://github.com/JcZou/ADRC
- StarryPilot Autopilot Software: https://github.com/JcZou/StarryPilot
- Matlab Quadrotor Simulator: https://github.com/JcZou/matlab_quadsim
- Gazebo Quadrotor Simulator: https://github.com/JcZou/gazebo_quadsim

REFERENCES

Fig. 11: The profile of gazebo quadrotor simulator, which shows communication between the ROS nodes.

Fig. 13: Disturbance rejection experimental results, where $\omega_0 = 150$, and $z_2$ is estimated total disturbance. A payload is deployed to mimic the disturbance added or removed process. (a) Disturbance responses of cascade PID. (b) Disturbance responses of proposed ADRC where $\gamma = 0.5$. (c) Disturbance responses of proposed ADRC where $\gamma = 0.8$. (d) Estimated total disturbance with cascade PID. (e) Estimated total disturbance with proposed ADRC where $\gamma = 0.5$. (f) Estimated total disturbance with proposed ADRC where $\gamma = 0.8$. 
Fig. 14: Signal tracking experimental results, where $\gamma = 0.5$ and $\omega_o = 150$. (a) angle tracking of cascade PID. (b) angle tracking of proposed ADRC. (c) angular velocity tracking of cascade PID. (d) angle tracking of proposed ADRC.

Fig. 15: Signal tracking with payload deployed experimental results, where $\gamma = 0.5$ and $\omega_o = 150$. (a) angle tracking of cascade PID. (b) angle tracking of proposed ADRC. (c) angular velocity tracking of cascade PID. (d) angle tracking of proposed ADRC.
Fig. 16: StarryPilot framework.


