Evaluation of PID and Adaptive Backstepping Controllers in Trajectory Tracking Regulation of a Quadrotor System

Yi Yang*

December 2024

Abstract

This technical report presents the development and analysis of a mathematical model for an autonomous quadrotor, focusing on advanced nonlinear control strategies to enhance its stability and performance. Two distinct control techniques are designed and implemented: a classical Proportional-Integral-Derivative (PID) controller and a more advanced nonlinear backstepping approach. The PID controller, widely used for its simplicity and robustness, provides baseline control performance, while the backstepping technique offers improved trajectory tracking and disturbance rejection through a systematic Lyapunov-based design. To assess the effectiveness of these controllers, extensive simulations are conducted under various operational scenarios, including external disturbances and dynamic flight conditions. The comparative analysis provides insights into the advantages and limitations of each control strategy, demonstrating the potential of backstepping for improved quadrotor autonomy.

1 Introduction

1.1 Background

Flying objects have long fascinated humanity, inspiring continuous research and technological advancement. Recent progress in sensors, actuators, processors, and energy storage solutions has substantially expanded the capabilities of unmanned aerial vehicles (UAVs). Among various UAV platforms, quadrotor systems are especially appealing due to their unique flight characteristics, such as the ability to hover, achieve vertical takeoff and landing, and maneuver with high agility [3]. These features make quadrotors well-suited for a wide range of applications, including building surveillance, search and rescue missions, and intervention in hazardous or confined environments. As the technological barriers to building and deploying quadrotors have decreased, the academic and industrial communities have devoted significant effort toward developing robust modeling, control, and path-planning methods [1, 7].

Despite their advantages, quadrotors are inherently underactuated and exhibit strongly coupled, nonlinear dynamics. This poses challenges in accurate trajectory tracking and stabilization, especially when facing wind disturbances, model uncertainties, or actuator/input delay issues [7, 2]. In addition, advanced tasks such as real-time path planning in dynamically changing environments remain intricate [4]. Prior work has highlighted multiple avenues for control, ranging from classical proportional–integral–derivative (PID) schemes to more sophisticated adaptive, robust, or intelligent control frameworks [20, 6, 8]. Although PID controllers are often praised for simplicity, they may struggle under severe model uncertainties or constraints such as actuator saturations or large external disturbances [10, 5].

To overcome these limitations, the literature explores diverse techniques. For instance, robust backstepping controllers [10] address nonlinearity by recursively constructing control laws for each subdynamic equation of the quadrotor's state. Adaptive methods, such as adaptive fuzzy backstepping, aim to compensate unknown parameters or exter-

nal disturbances [6], while neural-network-based approaches attempt to learn or approximate uncertainties online, preserving stability guarantees [9]. The combination of backstepping with concepts like sliding-mode control or super-twisting sliding surfaces has also been shown to mitigate chattering and handle modeling errors [5, 7]. Meanwhile, output-feedback designs incorporate observers to estimate unmeasured states, crucial for quadrotors with limited sensor payloads [26].

Another active research thread involves vision-based navigation and control. Some quadrotor systems rely on monocular cameras and inertial sensors for simultaneous localization and mapping. For example, image-based visual servoing (IBVS) is employed to track reference poses or detect objects in indoor settings without GPS, as discussed by Zheng et al. [27]. Elsewhere, data-driven detectors (e.g., Faster R-CNN) have been combined with tracking algorithms such as KCF (Kernelized Correlation Filters) to robustly follow multiple targets in dynamic scenes [1]. These approaches underscore how perception and control have become increasingly integrated, enabling more autonomous flights with less reliance on external infrastructure.

Beyond integral backstepping and intelligent observers, fractional-order control strategies have emerged as a promising area, given their ability to capture complex system behavior more accurately than integer-order models [14, 16, 15]. Fractional calculus has been leveraged in designing controllers with improved robustness and tuning flexibility for systems ranging from mechatronics to automotive idle-speed regulation [21, 18]. Specific to quadrotors, Timis et al. [8] compared classical PID controllers against fractional-order controllers in an adaptive structure, demonstrating that fractional-order designs can yield better closed-loop performance. Furthermore, fractional-order backstepping control can address uncertain quadrotor dynamics and unknown time delays while achieving precise trajectory tracking [17].

Other studies incorporate fractional-order PID controllers with optimization algorithms, such as particle swarm optimization (PSO) or genetic algorithms, to tune gains automatically [19, 12]. In certain cases, advanced or combined optimization processes, like multi-objective multi-verse optimization, guide the path planning tasks of UAVs to avoid collisions with moving obstacles [4]. Similarly, robust control can be realized through novel estimation methods, such as uncertainty and disturbance estimators, which can handle external wind gusts or unmodeled dynamics [2]. The synergy of these methodologies—fractional-order control, robust backstepping, adaptive fuzzy or neural network approaches—continues to push the boundaries of performance, reliability, and autonomy in UAV applications.

Meanwhile, the modeling side has not been overlooked. Researchers such as Yang and Zhang [13] have studied a range of fractional-order modeling and control techniques applicable in engineering and technology, culminating in broader treatments of fractional calculus [14] and its advanced applications in mechatronics and robotics [19]. Indeed, fractional-order approaches have been extended to more intricate tasks, like automotive idle speed control [21], or combined with advanced identification procedures for rotary inverted pendulums and other benchmark platforms [12]. These explorations serve as stepping stones for improved UAV modeling, particularly in uncertain or time-varying contexts. Additionally, multi-layer spiking neural networks have begun to appear in UAV control frameworks, introducing novel training paradigms such as fractional-order spike-timing-dependent gradient descent [17], reflecting an emerging intersection of neuromorphic computing with UAV guidance.

Finally, real-world constraints often necessitate meeting size, weight, and power (SWaP) requirements. As described in a variety of technical reports and doctoral dissertations [22, 23, 25, 24, 11], researchers continue to refine both hardware and software integration for quadrotors, aiming to transition from simulation or laboratory demonstrations to field deployment. The combination of robust or adaptive backstepping, fractional-order control, and advanced state estimators marks a critical path toward building next-generation aerial robots that can navigate confined indoor spaces or challenging outdoor settings with minimal human intervention. In summary, the pursuit of accurate, stable, and robust quadrotor control remains an active and fertile research area, drawing on innovative control theories and computational intelligence to meet the demands of modern aerial applications.

1.2 Aerodynamic Modelling of Quadrotor Aircraft

The model will have 12 states: displacement in three directions(x,y,z), velocity in three directions($\dot{x}, \dot{y}, \dot{z}$), roll angle($\phi$), pitch angle($\theta$), yaw angle($\psi$) and three angular velocities($\dot{\phi}, \dot{\theta}, \dot{\psi}$) with respect to these three angles. Let $[\phi, \dot{\phi}, \theta, \dot{\theta}, \psi, \dot{\psi}, z, \dot{z}, x, \dot{x}, y, \dot{y}] = [x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}]$. The original nonlinear dynamic model for this system is provided as:

$$
\begin{aligned}
\dot{x_1} &= x_2 \\
\dot{x_2} &= \frac{(I_y - I_z)}{I_x} x_4 x_6 + \frac{U_2 l}{I_x} \\
\dot{x_3} &= x_4 \\
\dot{x_4} &= \frac{(I_z - I_x)}{I_y} x_2 x_4 + \frac{U_3 l}{I_y} \\
\dot{x_5} &= x_6 \\
\dot{x_6} &= \frac{(I_x - I_y)}{I_z} x_2 x_4 + \frac{U_4}{I_z} \\
\dot{x_7} &= x_8 \\
\dot{x_8} &= cos(x_1) cos(x_3) \frac{U_1}{m} - g \\
\dot{x_9} &= x_{10} \\
\dot{x_{10}} &= (sin(x_3) cos(x_1) cos(x_5) + sin(x_1) sin(x_5)) \frac{U_1}{m} \\
\dot{x_{11}} &= x_{12} \\
\dot{x_{12}} &= (sin(x_3) cos(x_1) sin(x_5) - \sin(x_1) cos(x5)) \frac{U_1}{m}
\end{aligned}
$$

where

$$
\begin{aligned}
U_1 &= b(\Omega_1^2 + \Omega_2^2 + \Omega_2^2 + \Omega_4^2) \\
U_2 &= b(\Omega_4^2 - \Omega_2^2) \\
U_3 &= b(\Omega_3^2 + \Omega_1^2) \\
U_4 &= d(-\Omega_1^2 + \Omega_2^2 - \Omega_2^2 + \Omega_4^2)
\end{aligned}
$$

We choose the system model parameters as values in Table 1.

2 System Design Based on PID Method

2.1 Design the PID Controller

From the aerodynamic equations of quadrotors we know the attitude angles and angular velocities are independent of translation velocity. However, the translation motions depend on angular motion. Therefore, we can divide the whole system into two subsystems: linear motion subsystem and angular motion subsystem, they have semi-coupling relations as illustrated by the following figure:

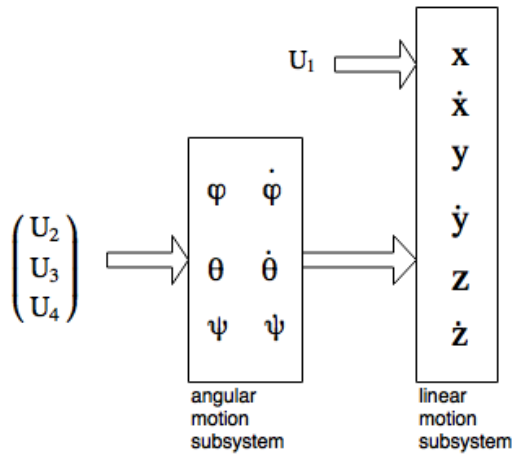


Figure 1: System with controller

According to this semi-coupled relation, we can first design the inner loop(attitude control) and then outer loop(position control), as shown in Figure2b. The inner loop is to regulate the attitude angles and ensure the stability when the flight is steadily flying or staying in a fixed position. The outer loop is to control the flight to fly in our target trajectory and finally make sure the quadrotor attain our goal destination. (x_d, y_d, z_d) is the reference target position, $(\phi_d, \theta_d, \psi_d)$ is the expected value of our inner loop subsystem.

First we simplify the dynamic equation

$$
\begin{aligned}
\dot{x_{10}} &= Ux \frac{U_1}{m} \\
\dot{x_{12}} &= Uy \frac{U_1}{m}
\end{aligned}
$$

where

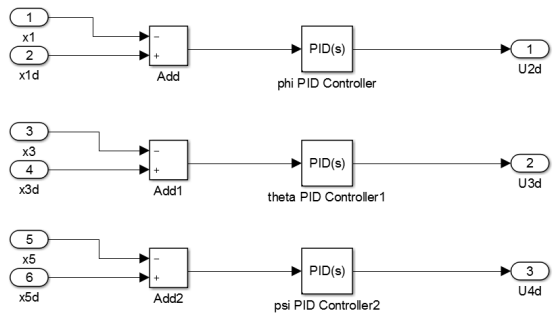
$$
\begin{aligned}
U_x &= sin(x_3) cos(x_1) sin(x_5) - \sin(x_1) cos(x5) \\
U_y &= sin(x_3) cos(x_1) cos(x_5) + sin(x_1) sin(x_5)
\end{aligned}
$$

The XYZ controller use the information from feedback and desired x, y, z to decide the desired U_{xd} and U_{yd}, then U_{xd} and U_{yd} is used to compute the desired $x_{1d}(\phi)$ and $x_{3d}(\theta)$, where

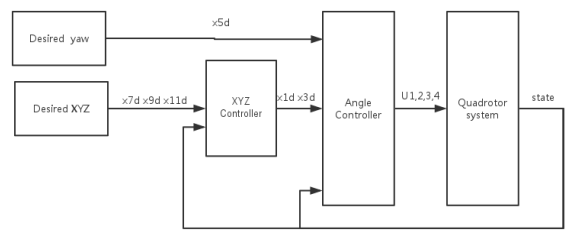
$$
\begin{aligned}
x_{1d} &= arcsin(u_x sin(x_5) - u_y cos(x_5)) \\
x_{3d} &= arcsin[\frac{u_x - sin(x_1) sin(x_5)}{cos(x_1) cos(x_5)}]
\end{aligned}
$$

g	m	L	I_x	I_y	I_z	d	b
9.81	0.53	0.232	$6.228e-3$	$6.228e-3$	$1.121e-2$	$7.5e-7$	$3.13e-5$

Table 1: Geometric parameters of the quadrotor system.



(a) Example of PID control



(b) A simplified chart of how the control system work

Figure 2: PID control system

Use desired x1d and x3d as reference for the Angle Controller to compute the desired U1d U2d U3d and U4d. Finally compute the needed voltage, which is the real input.

The PID parameter we have chosen is shown in Table2. These parameters are used in the continuous case and in the 0.05s sampling case. However in the 0.1s sampling case the parameters of Angle PID Controller are modified to keep stable.

	K_p	K_i	K_d
ϕ	0.5	0.1	0.15
θ	0.5	0.1	0.15
ψ	0.1	0.05	0.1
x	2	0.25	3
y	2	0.35	3
z	1	0.2	1

Table 2: Parameters for PID Controller

Use the linearized system we plot the open loop Bode plot for the phi,see Figure 3 The gain margin is 24.1 db, the phase margin is 46.4deg, which is satisfactory.

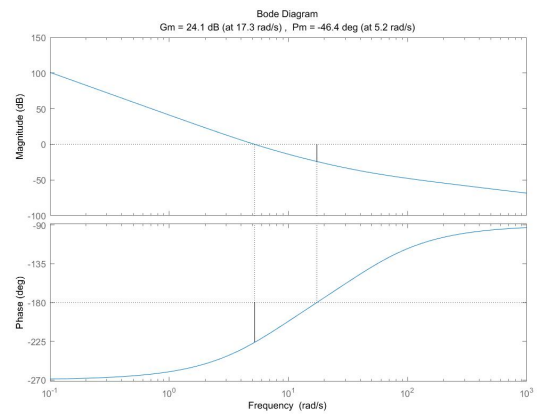


Figure 3: Bode plot of the linearized system

2.2 Simulation Results and Analysis

We first test the controller in continuous system. The initial condition is set to be $\phi = 0.4$, $\theta = 0.3$, $\psi = 0.2$, $x = 0$, $y = 0$, $z = 0$. The destination is $x = 1$, $y = 1.5$, $z = 2$. The result is shown in the Figure 4. In fact a more optimal choice of the PID parameters can be chosen for the continuous system, however for consistency, we chose the optimal parameters that can make the 0.05s discrete system stable. Figure 5 shows the result of sampling time 0.05s. It can be observed that the Angles oscillate more and has higher amplitude. As for positions,

Oscillation is introduced during rise. The advantage of this discrete controller is that the rise time and the overshoot of x y is reduced. Figure 6 show the result of sampling time 0.1. The stable range of parameters shrank as the sampling time increase, so we chose a new set of parameter for this case, the performance is limited compared with previous cases because any aggressive choice of parameters will cause unstable.

We perform more tests in the case of sampling time 0.05s. First we add disturbances to the system. A 2m/s upwind occur at 15s and end at 20s, an extra weight of 0.5kg is added to the quadrotor at 15s and is not removed until the end. The result is shown in Figure 7. It can be seen that the PID controller successfully handle these two case. However the controller fail when the wind speed reach 18m/s. There is no limitation for the extra weight in our simulation. However the overshoot will be very large if the extra weight is too high. It is also known that in real life the extra weight is limited by the motors capacity.

Then we add noise to our feedbacks, the gaussian noise is majorly limited in 0.03m and 0.03rad. Figure 8 shows the result. From the plot we can observe that z is not significantly affected by the noise, because the noise influence in four rotors is neutralized. The converge process of x and y basically remain the same except that they preserve the noise after settle. In the case of phi and theta, the noise is amplified in the steady-state response.

2.3 Summary

In conclusion, the PID controller perform well in both continuous and discrete case. It also show robustness under disturbances and noises.

	K_p	K_i	K_d
ϕ	0.5	0.1	0.15
θ	0.5	0.1	0.15
ψ	0.1	0.05	0.1
x	2	0.25	3
y	2	0.35	3
z	1	0.2	1

Table 3: Parameters for PID Controller

3 System Design Based on Backstepping Method

Linear feedback is the most common controller design method for MIMO system. It abstract linear ingredient from the nonlinear model by linearization. To control the system with high precision, it requires the system with highly linear characteristic. On the contrary, Backstepping control does not have to eliminate the nonlinear portion, which makes it efficient in controlling nonlinear system. In this chapter, we will show a method named backstepping and induce an expression of the control variables. Then, trying to control the quadrotor system by Backstepping method.

3.1 Backstepping controller design

3.1.1 Simulation of Backstepping control with simplified inputs

According to Appendix A.1 and A.1.1, we could built a Backstepping controller in Matlab Simulink as it is shown in Figure 9.

In the following simulation, we test the Backstepping, a nonlinear control method. We simulate the system in continuous time, tying to control the quadrotor fly from point (0,0,0) to point (1,1.5,2). Total simulation time is 30 seconds with the initial condition $X = [\phi\ \dot{\phi}\ \theta\ \dot{\theta}\ \psi\ \dot{\psi}\ z\ \dot{z}\ x\ \dot{x}\ y\ \dot{y}]^T = [0.4, 0, 0.3, 0, 0.2, 0, 0, 0, 0, 0, 0, 0]^T$. Moreover, parameters c_1——c_{12} also have influence on the simulation results. As c_1——c_{12} getting large, control ability gets stronger, as well as the control error. On the other hand, if c_1——c_{12} become smaller, oscillation and overshoot happens more frequently. In the results shown in Figure 10, we choose $c_1 = 10; c_2 = 3; c_3 = 5; c_4 = 8; c_5 = 3; c_6 = 2; c_7 = 2; c_8 = 2; c_9 = 2; c_{10} = 2; c_{11} = 2; c_{12} = 2$; (Readers could run **run_backstepping.m** file first, then run **backstepping.slx** file)

Simulation result in Figure 10 seems quite satisfying. The whole system become steady in about 3 seconds without over shoot along x,y,z axes.

3.1.2 Simulation of Backstepping control with real inputs

In practical cases, the only inputs we can directly reach are the voltages of four rotors. In the next simulation, we solved the desired inputs of rotor

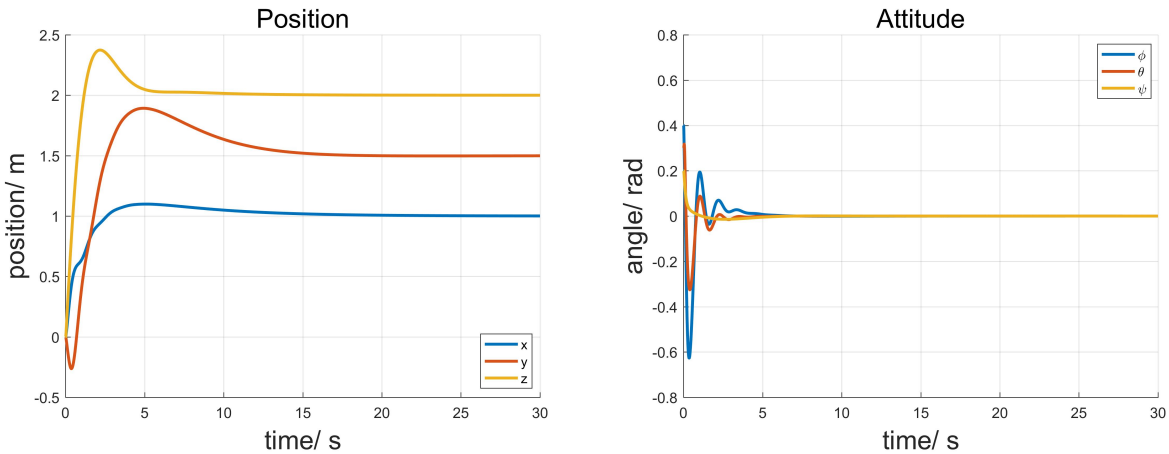


Figure 4: PID control in continuous time

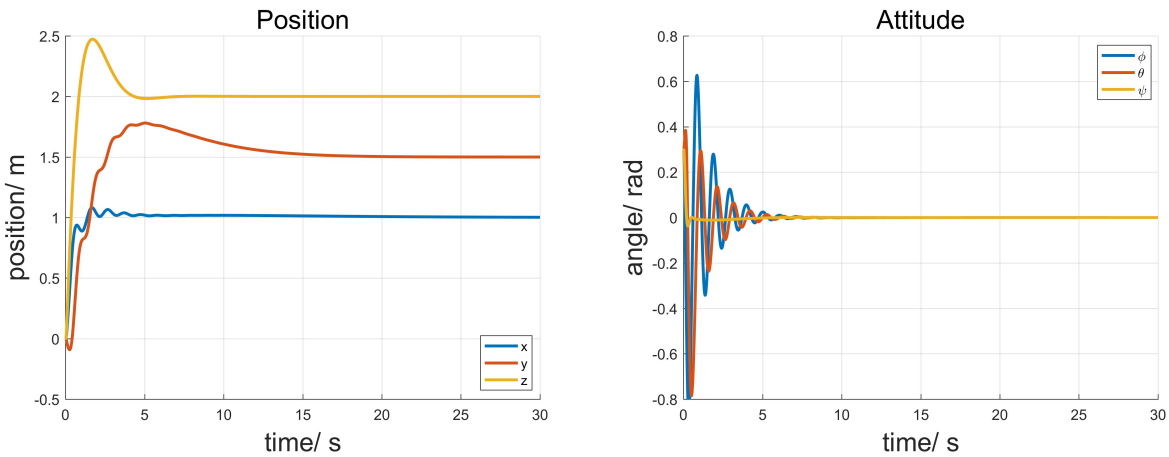


Figure 5: PID control in sampling time=0.05s

voltage with a "U trans V" sub-block and considered the transfer function of electric rotor as $\frac{800}{0.01s+1}$. The rotating speed of rotors grow 800r/s as the input voltage grow 1V. Initial condition and parameters remain unchanged. (Readers could run **run_backstepping.m** file first, then run **backstepping_real.slx** file)

From Figure 11, we noticed a slight difference from the real situation to the simplified condition. It took about 2 seconds for the quadrotor stay steady after time 0. Fortunately, though the rotor conducted a minor delay of the rotating speed, the response time of the system were still kept at about 3 seconds.

3.1.3 Simulation of Backstepping control in discrete time

As we know, in practical researches, controllers are working under discrete time. Figure 12 and Figure 13 tested the controlling results in different sampling time, 0.05s and 0.1s. (Readers could run **run_backstepping_discrete.m** file first and change the parameter **samplingtime**, then run **backstepping_real_discrete.slx** file)

As shown in those pictures, vibration occurs when sampling time is changed. The amplitude of vibration is about $0.01rad \approx 0.573deg$ maximum, which is acceptable under after all. To keep the same pace of PID control, we choose sampling time=0.05s in the following sections.

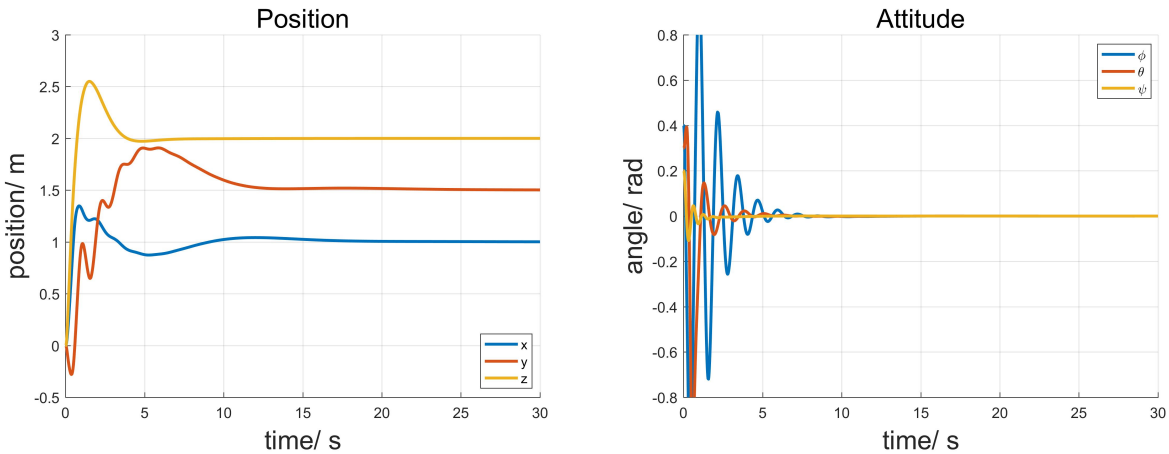


Figure 6: PID control in sampling time=0.1s

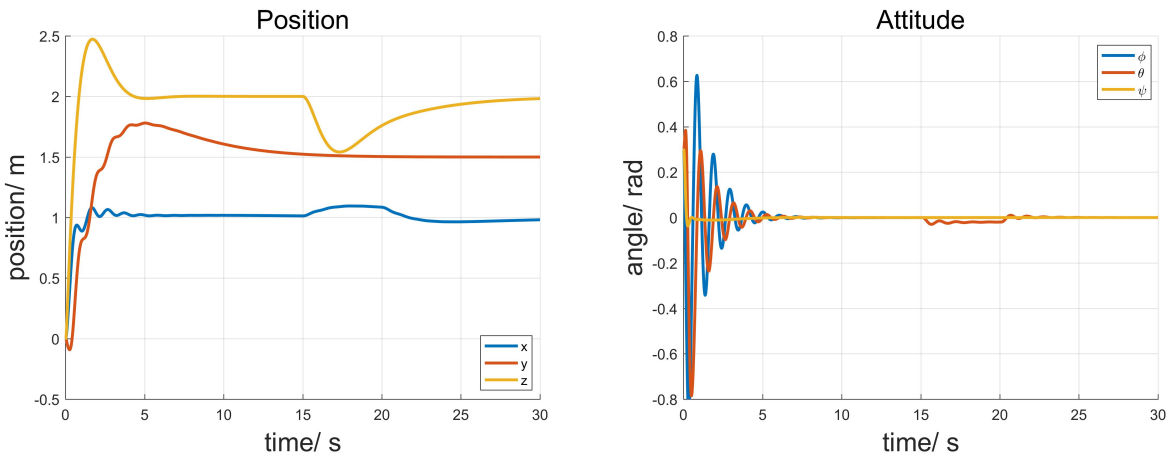


Figure 7: PID control in discrete time under disturbance

3.1.4 Backstepping control in discrete time under disturbance

To test the robustness of backstepping control, we applied several disturbances on the quadrotor. A 2m/s horizontal wind on x axis starts at t=15s and ends at t=20s. An additional weight m=0.5kg will be add on the quadrotor at t=15 and maintained until t=30s. Simulation results are shown in Figure 14. (Readers could run **run_backstepping_discrete_disturb.m** file first and then run **backstepping_real_disc rete_disturb.slx** file)

The simulation time in Figure 14 is 30 seconds. We noticed, from plot of z, that the system will be stable after disturbance had happened for about 3 seconds and stay at the new stable point. Due to the plot of position x, the quadrotor will not flying back to point (1,1.5,2) as long as the disturbance exists. To conduct the system adaptive to the disturbances, an Adaptive Backstepping approach will be deduced in the next subsection.

3.1.5 Simulation of Adaptive Backstepping control in discrete time under disturbances

We are capable to solve the unknown disturbance problem after building an Adaptive Backstepping Controller as written in Appendix A.2.

Figure 15 shows the simulation result of Adaptive Backstepping control when disturbances are applied,i.e. a 2m/s horizontal wind on x axis starts at t=15s and ends at t=20s, an unknown addi-

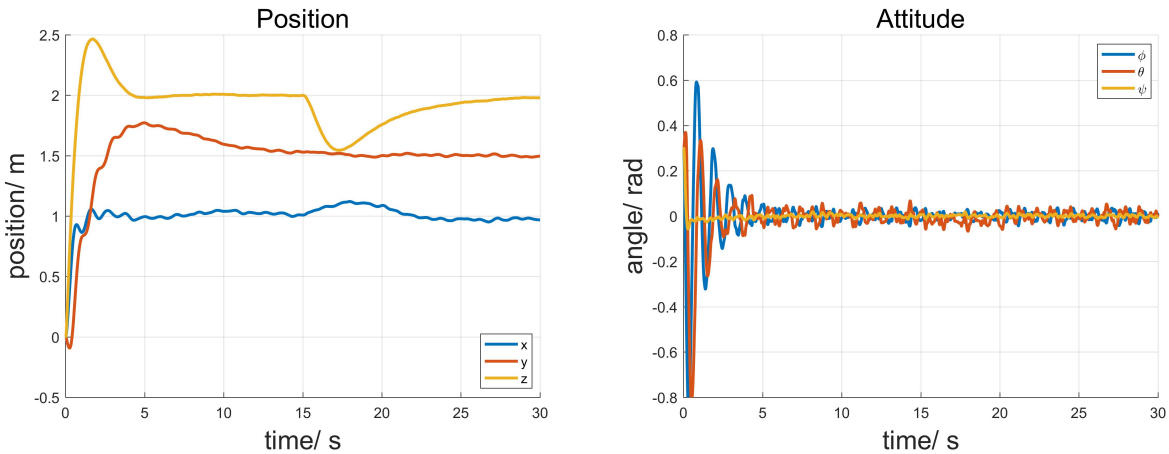
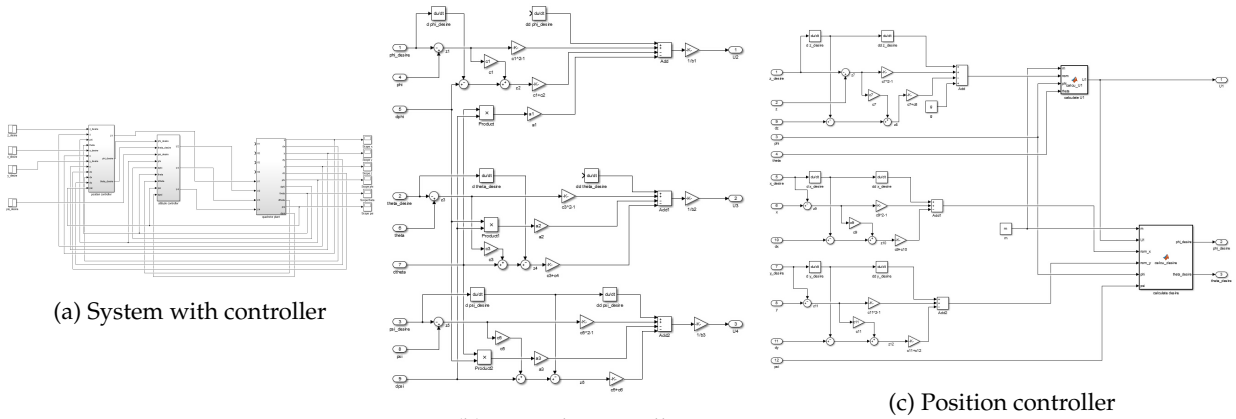


Figure 8: PID control with noise



(a) System with controller

(b) Attitude controller

(c) Position controller

Figure 9: Backstepping control system

tional weight m=0.5kg added on the quadrotor at t=15 and maintained until t=30s. From the plot of attitude, we can conclude that adaptive backstepping control converges faster than the non-adaptive one. Moreover, adaptive control successfully estimate the unknown disturbance and fly back to the desired position. (Readers could run **run_backstepping_discrete_disturb.m** file first and then run **backstepping_real_disc rete_disturb_adp.slx** file)

3.1.6 Simulation of Adaptive Backstepping control with noise

In the next simulation, disturbances and other parameters are kept same as before. Furthermore, noise of amplitude$\approx$ 0.6 are added in $\ddot{x}, \ddot{y}, \ddot{z}, \ddot{\phi}, \ddot{\theta}, \ddot{\psi}$. Note that, in this case, noises will also be added back to the system itself, which could be regarded as a combination of motion error and sensor error. We could discover from the simulation result(Figure 16) that vibration will occur due to noises. The amplitude of vibration is about 0.005m along x and y axis and $0.01rad \approx 0.05deg$ on rolling angleϕ and pitch angle θ. (Readers could run **run_backstepping_discrete_disturb.m** file first and then run **backstepping_real_disc rete_disturb_adp_noise.slx** file)

4 Comparison

From the comparison we can find that the Backstepping has a more keen response than PID. What's

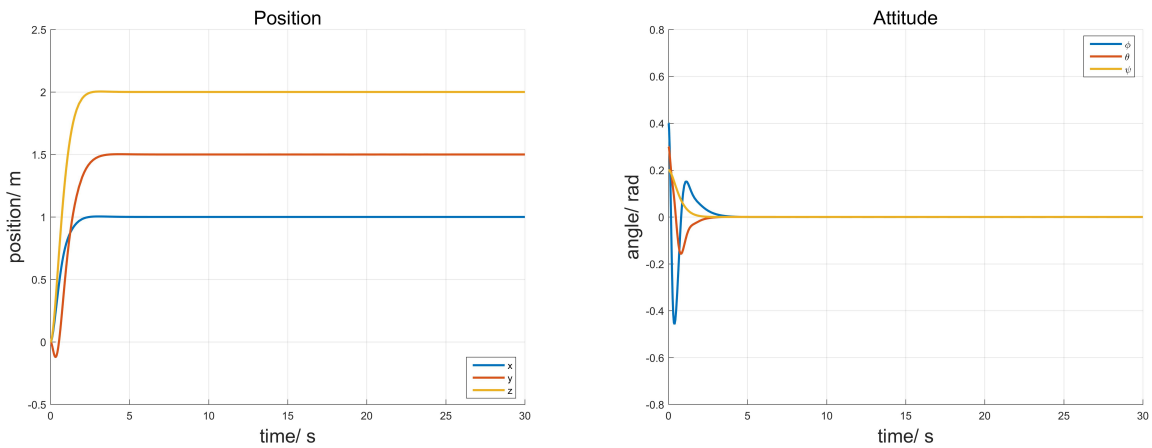


Figure 10: Backstepping control in continuous time

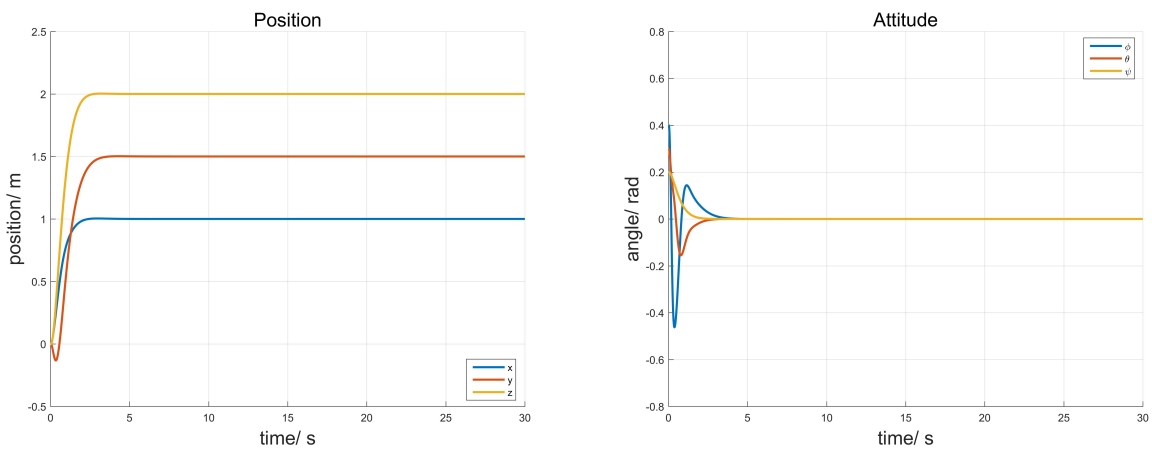


Figure 11: Backstepping control with inputs $V1, V2, V3, V4$ in continuous time

more, the Backstepping has less overshoot, which means that in reality it has less chance hit obstacles which is not in the expected trajectory. The angle oscillation frequency and amplitude in Backstepping are also less than those in PID.

Future research on adaptive backstepping control could be expanded to encompass fractional-order systems, leveraging the latest advancements and applications of fractional calculus. For a comprehensive list of relevant literature, please refer to Reference list.

A Backstepping control

A.1 Backstepping attitude control

As shown in chapter two, the attitude controller will not influence the position controller. Hence, we could design three Backstepping controllers to control the three angles ϕ, θ, ψ respectively. Take the rolling angle ϕ as an example, we could deduce the expression of U_2 by Backstepping method:

$$\begin{cases} \dot{x_1} &= \dot{\phi} &= x_2 \\ \dot{x_2} &= a_1\dot{\theta}\dot{\psi} + b_1 U_2 &= a_1 x_4 x_6 + b_1 U_2 \end{cases}$$

where $a_1 = (I_y - I_z)/I_x$, $b_1 = L/I_x$.

Step 1

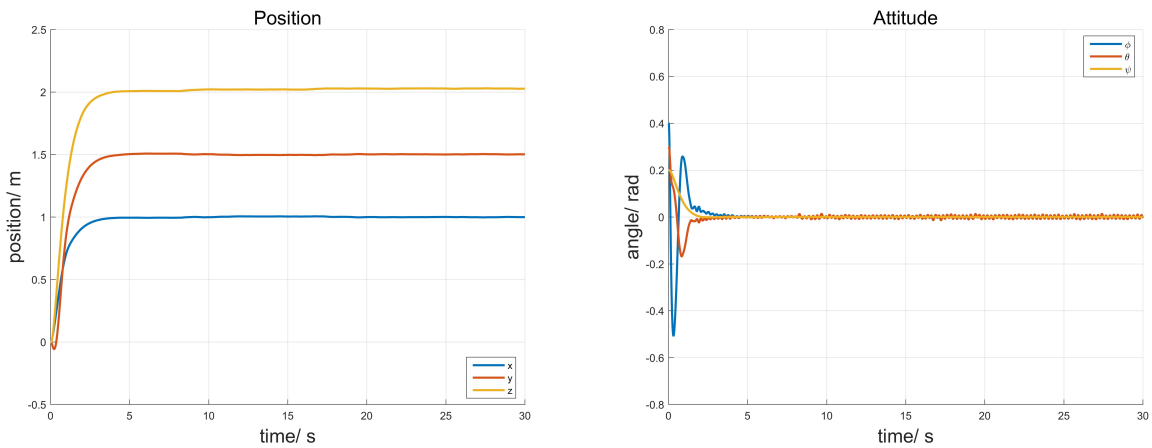


Figure 12: Backstepping control in sampling time=0.05s

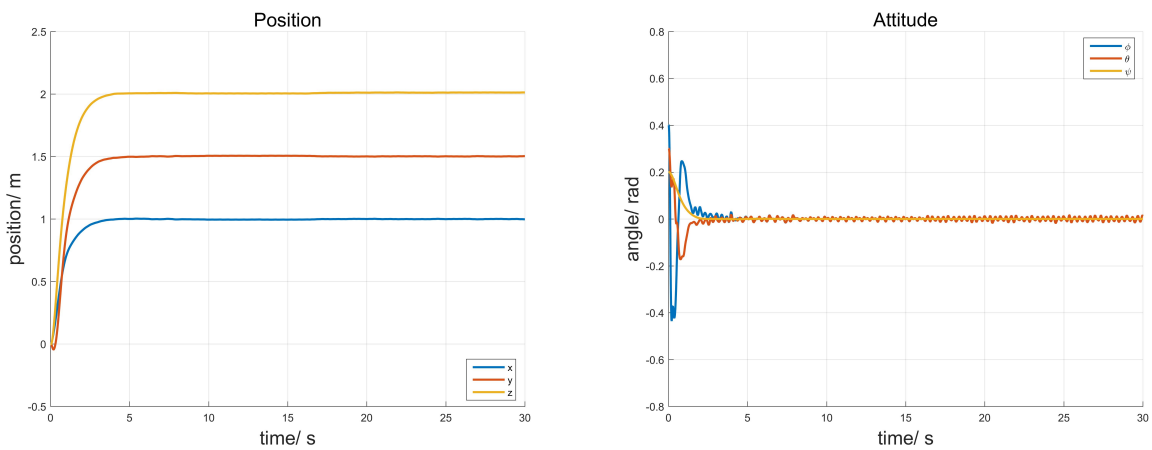


Figure 13: Backstepping control in sampling time=0.1s

Denote the desired rolling angle as $\phi_d = x_{1d}$, and the error variable $z_1 = x_1 - x_{1d}$. Then take the derivative of z_1, we have:

$$\dot{z_1} = \dot{x_1} - \dot{x_{1d}} = x_2 - \dot{x_{1d}}$$

Then, we define a virtual control variable α_1, such that $z_2 = x_2 - \alpha_1$, and choose the Lyapunov function $V_1 = \frac{1}{2}z_1^2$. Thus:

$$\dot{V_1} = z_1 = z_1(x_2 - \dot{x_{1d}}) = z_1(z_2 + \alpha_1 - \dot{x_{1d}})$$

According to Lyapunov stability criterion, we need to choose $\alpha_1 = \dot{x_{1d}} - c_1 z_1$ such that $\dot{V_1} < 0$, where $c_1 > 0$. Then we could have:

$$\dot{V_1} = -c_1 z_1^2 + z_1 z_2$$
$$z_2 = x_2 - \dot{x_{1d}} + c_1 z_1$$

We could see that if $z_2 = 0$, then z_1 is asymptotically stable. Since, z_2 is not always zero, we need another control variable for $z_1 z_2$.

Step 2

Define Lyapunov function $V_2 = \frac{1}{2}z_2^2 + V_1$, since

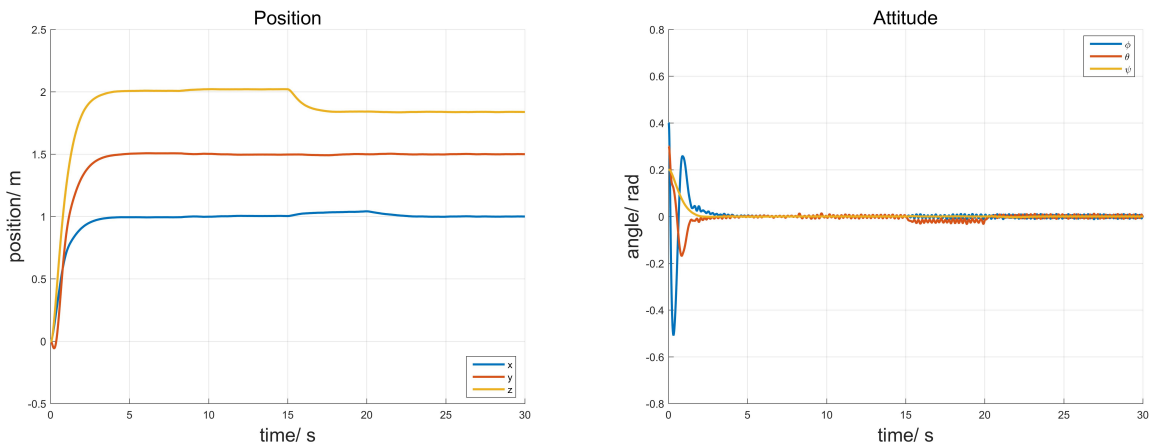


Figure 14: Backstepping control in discrete time under disturbance

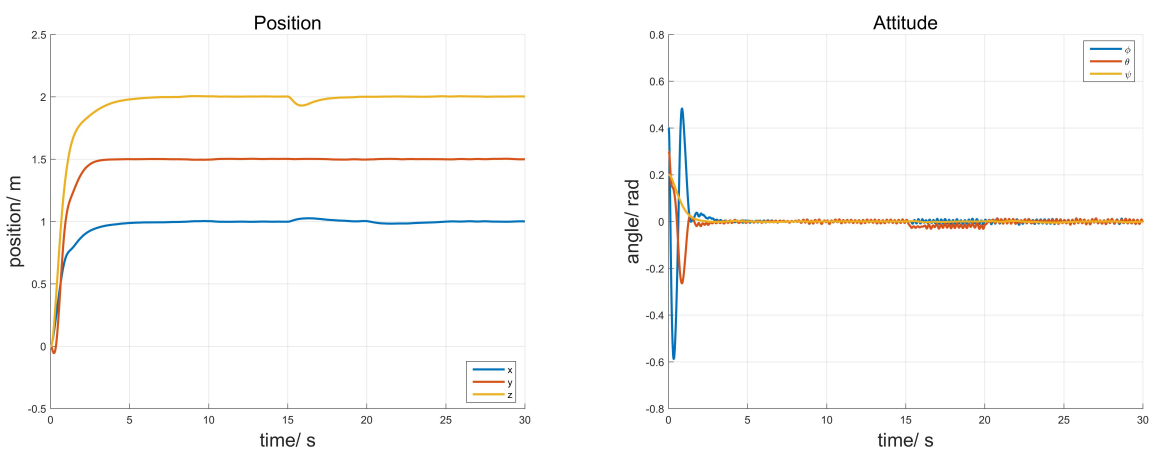


Figure 15: Adaptive Backstepping control in discrete time under disturbance

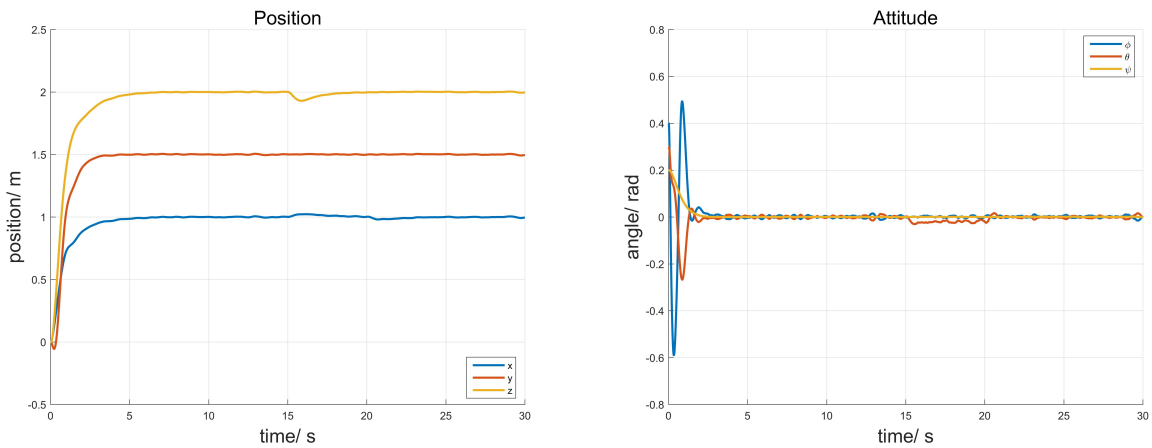


Figure 16: Adaptive Backstepping control with sensor noise

Response Properties	PID	Backstepping
time to the expected altitude	5s	4s
time to the target horizontal position	15s	4s
time to recover from disturbance	10s	5s

Table 4: Performance of Different Methods

$\dot{z_2} = \dot{x_2} - \ddot{x_{1d}} + c_1\dot{z_1}$, we have:

$$\begin{aligned}\dot{V_2} &= z_2\dot{z_2} + V_1 = z_2(\dot{x_2} - \ddot{x_{1d}} + c_1\dot{z_1}) - c_1z_1^2 + z_1z_2 \\ &= z_2(b_1U_2 + a_1x_4x_6 - \ddot{x_{1d}} + c_1\dot{z_1}) - c_1z_1^2 + z_1z_2 \\ &= z_2(b_1U_2 + a_1x_4x_6 - \ddot{x_{1d}} + c_1(z_2 - c_1z_1)) - c_1z_1^2 + z_1z_2\end{aligned} \tag{1}$$

Then, the control input should be:

$$\begin{aligned}U_2 &= \frac{1}{b_1}(\ddot{x_{1d}} - z_1 - a_1x_4x_6 - c_1z_2 + c_1^2z_1 - c_1z_2) \\ &= \frac{1}{b_1}(\ddot{x_{1d}} + (c_1^2 - 1)z_1 - a_1x_4x_6 - (c_1 + c_2)z_2)\end{aligned} \tag{2}$$

where $c_2 > 0$, $\ddot{x_{1d}} = 0$, $z_1 = x_1 - x_{1d}$, $z_2 = x_2 - \dot{x_{1d}} + c_1z_1$. Put (2) into (1), we have:

$$\dot{V_2} = -c_1z_1^2 - c_2z_2^2 < 0$$

According to Lyapunov stability criterion, U_2 can ensure that z_1 and z_2 are asymptotically stable.

Same reasoning, we could deduce the control variable for pitch angle θ and yaw angle ψ as:

$$\begin{cases}U_3 = \frac{1}{b_2}[\ddot{x_{3d}} + (c_3^2 - 1)z_3 - a_2x_2x_6 - (c_3 + c_4)z_4] \\ U_4 = \frac{1}{b_3}[\ddot{x_{5d}} + (c_5^2 - 1)z_5 - a_3x_2x_4 - (c_5 + c_6)z_6]\end{cases}$$

where:

$$\begin{aligned}z_3 &= \theta - \theta_d = x_3 - x_{3d} \\ z_4 &= x_4 - \dot{x_{3d}} + c_3z_3 \\ z_5 &= \psi - \psi_d = x_5 - x_{5d} \\ z_6 &= x_6 - \dot{x_{5d}} + c_5z_5\end{aligned}$$

A.1.1 Backstepping position control

Position control is a combination of altitude control and horizontal control. The altitude z is controlled by input U_1, and horizontal position x,y are not directly controlled by inputs. Since x, y are directly correlated with ϕ and θ, we could define some virtual input U_x, U_y, expressed by ϕ and θ, parallel to U_1. Take the altitude controller as an example, we could design a Backstepping controller.

The dynamic equations of z and $\dot{z}$ are:

$$\begin{cases}\dot{x_7} &= x_8 \\ \dot{x_8} &= -g + (cos(x_1)cos(x_3))\frac{U_1}{m}\end{cases} \tag{3}$$

Step 1

For a given desired altitude $z_d = x_{7d}$, define the error variable $z_7 = x_7 - x_{7d}$ and calculate its derivation:

$$\dot{z_7} = \dot{x_7} - \dot{x_{7d}} = x_8 - \dot{x_{7d}}$$

Then, define α_7, where $z_8 = x_8 - \alpha_7$, and choose the Lyapunov function $V_7 = \frac{1}{2}z_7^2$, then:

$$\dot{V_7} = z_7\dot{z_7} = z_7(x_8 - \dot{x_{7d}}) + z_7z_8$$

To ensure z_7 asymptotically stable at the origin, according to Lyapunov stability criterion, we should have $\dot{V_7} < 0$, i.e. $\alpha_7 = \dot{x_{7d}} - c_7z_7(c_7 > 0)$. Thus:

$$\begin{aligned}\dot{V_7} &= -c_7z_7^2 + z_7z_8\\ z_8 &= x_8 - \dot{x_{7d}} + c_7z_7\end{aligned}$$

Same as we did in attitude control, we should define another Lyapunov function V_8 such that $z_8 \to 0$, in order to express our control variables.

Step 2

Define Lyapunov function $V_8 = \frac{1}{2}z_8^2 + V_7$, since $\dot{z_8} = \dot{x_8} - \ddot{x_{7d}} + c_7\dot{z_7}$, then:

$$\dot{V_8} = z_8\dot{z_8} + V_1 = z_8(\dot{x_8} - \ddot{x_{7d}} + c_7\dot{z_7}) - c_7z_7^2 + z_7z_8 \tag{4}$$

$$\begin{aligned}&= z_8[\dot{x_8} - \ddot{x_{7d}} + c_7(z_8 - c_7z_7)] - c_7z_7^2 + z_7z_8\\ &= -c_7z_7^2 + z_8[z_7 - \ddot{x_{7d}} + c_7(z_8 - c_7z_7) + \dot{x_8}]\end{aligned}$$

To guarantee $\dot{V_8} < 0$, choose:

$$z_7 - \ddot{x_{7d}} + c_7(z_8 - c_7z_7) + \dot{x_8} = -c_8z_8(c_8 > 0) \tag{5}$$

Put (5) into (4), we have:

$$\begin{aligned}\dot{V_8} &= -c_7z_7^2 + z_8[z_7 - \ddot{x_{7d}} + c_7\dot{z_7} + \dot{x_8}]\\ &= -c_7z_7^2 - c_8z_8^2 < 0\end{aligned}$$

Moreover, (5) can be written as:

$$\dot{x_8} = \ddot{x_{7d}} + (c_7^2 - 1)z_7 - (c_7c_8)z_8(c_7 > 0, c_8 > 0) \tag{6}$$

Put (6) into (3):

$$U_1 = \frac{m}{cos(x_1)cos(x_3)}[\ddot{x_{7d}} + (c_7^2 - 1)z_7 + g - (c_7 + c_8)z_8]$$

Now, we have U_1 such that z_7 and z_8 are asymptotically stable.

Same reasoning, we could deduce the expression for virtual control variables:

$$\begin{cases} u_x &= \frac{m}{U_1}[\ddot{x_{9d}} + (c_9^2 - 1)z_9 - (c_9 + c_{10})z_{10}]\\ u_y &= \frac{m}{U_1}[\ddot{x_{11d}} + (c_{11}^2 - 1)z_{11} - (c_{11} + c_{12})z_{12}]\end{cases} \tag{7}$$

where

$$\begin{aligned}z_9 &= x - x_d = x_9 - x_{9d}\\ z_{10} &= x_{10} - \dot{x_{9d}} + c_9z_9\\ z_{11} &= y - y_d = x_{11} - x_{11d}\\ z_{12} &= x_{12} - \dot{x_{11d}} + c_{11}z_{11}\end{aligned}$$

Similar to what we did in PID control, we need the outputs of rolling angle and pitch angle of the position controller and take them as the desired inputs of the attitude controller. Thus, according to the system model, we acquire the nonlinear relationship between ϕ_d, θ_d and u_x, u_y:

$$\phi_d = arcsin(u_xsin(\psi) - u_ycos(\psi)) \tag{8}$$

$$= arcsin(u_xsin(x_5) - u_ycos(x_5)) \tag{9}$$

$$\theta_d = arcsin[\frac{u_x - sin(\phi)sin(\psi)}{cos(\phi)cos(\psi)}] \tag{10}$$

$$= arcsin[\frac{u_x - sin(x_1)sin(x_5)}{cos(x_1)cos(x_5)}] \tag{11}$$

Put (9) and (11) into (7), we could deduce the final expression of inputs:

$$\begin{cases} U_1 &= \frac{m}{cos(x_1)cos(x_3)}[\ddot{x_{7d}} + (c_7^2 - 1)z_7 + g - (c_7 + c_8)z_8]\\ U_2 &= \frac{1}{b_1}[\ddot{x_{1d}} + (c_1^2 - 1)z_1 - a_1x_4x_6 - (c_1 + c_2)z_2]\\ U_3 &= \frac{1}{b_2}[\ddot{x_{3d}} + (c_3^2 - 1)z_3 - a_2x_2x_6 - (c_3 + c_4)z_4]\\ U_4 &= \frac{1}{b_3}[\ddot{x_{5d}} + (c_5^2 - 1)z_5 - a_3x_2x_4 - (c_5 + c_6)z_6]\end{cases}$$

A.2 Adaptive Backstepping control in discrete time under disturbance

In the real environment, a quadrotor will always be influenced by unknown parameters and unexpected disturbances denoted as D. In order to online estimate disturbance D and calculate adaptive feedback by Backstepping method, we could express the dynamic equation of position subsystem as following:

$$\begin{cases} m\ddot{x} &= u_xU_1 + D_x\\ m\ddot{y} &= u_yU_1 + D_y\\ m\ddot{z} &= cos(\phi)cos(\theta)U_1 - mg + D_z\end{cases}$$

where $u_x = sin(\theta)cos(\phi)cos(\psi) + sin(\phi)sin(\psi), u_y = sin(\theta)cos(\phi)sin(\psi) - sin(\phi)cos(\psi)$ and $D_i(i = x, y, z)$ is denoted as disturbance on x,y,z axes.

Take the altitude state z as an example, it's state space representation is:

$$\begin{cases} \dot{x_7} = & x_8\\ \dot{x_8} = & \frac{1}{m}(cos(x_3)cos(x_1)U_1 - mg + D_z)\end{cases}$$

where $x_7 = z, x_8 = \dot{z}, x_1 = \phi, x_3 = \theta$ and D_z is the disturbance added on z axis. Since D_z is unknown, we need to online estimate it and define the estimated D_z as $\hat{D}_z$, the estimate error $\tilde{D}_z = D_z - \hat{D}_z$.

For a given desired altitude $z_d = x_{7d}$, define the altitude error variable as $z_7 = x_7 - x_{7d}$, speed error variable $z_8 = x_8 - \alpha_7$, where α_7 is a virtual control variable.

Step 1: design for z_7 subsystem.

$$\dot{z_7} = \dot{x_7} - \dot{x_{7d}} = x_8 - \dot{x_{7d}}$$

Take x_8 as the control variable, if $x_8 = \dot{x_{7d}} - c_7 z_7$, then z_7 subsystem is asymptotically stable. Since x_8 is a transient control variable, we can only express x_8 by α_7. Thus, design:

$$\alpha_7 = \dot{x_{7d}} - c_7 z_7 (c_7 > 0)$$

Choose the lyapunov function of the first subsystem as $V_7 = \frac{1}{2} z_7^2$, then:

$$\begin{aligned} \dot{V_7} &= z_7 \dot{z_7} = z_7(x_8 - \dot{x_{7d}}) \\ &= z_7(z_8 + \alpha_7 - \dot{x_{7d}}) \\ &= z_7 z_8 - c_7 z_7^2 \end{aligned}$$

If $\dot{V_7} < 0$, $z_7 z_8$ will be canceled in the next step and the first subsystem would be stable.

Step 2: calculate for the actual control law and the adaptive control law.

$$\dot{z_8} = \dot{x_8} - \ddot{x_{7d}} + c_7 \dot{z_7}$$

Define Lyapunov function as $V_8 = \frac{1}{2} z_8^2 + V_7$, then:

$$\begin{aligned} \dot{V_8} &= z_8 \dot{z_8} + \dot{V_7} = z_8(\dot{x_8} - \ddot{x_{7d}} + c_7 \dot{z_7}) - c_7 z_7^2 + z_7 z_8 \\ &= z_8[\dot{x_8} - \ddot{x_{7d}} + c_7(z_8 - c_7 z_7)] - c_7 z_7^2 + z_7 z_8 \\ &= z_8[\frac{1}{m}(cos(x_3)cos(x_1)U_1 - mg + D_z) - \ddot{x_{7d}} + \\ &\quad c_7(z_8 - c_7 z_7)] - c_7 z_7^2 + z_7 z_8 \end{aligned}$$

Now, the system input:

$$\begin{aligned} U_1 = &\frac{m}{cos(x_3)cos(x_1)}(x_{7d} - z_7 - c_8 z_8 - c_7 z_8 + c_7^2 z_7 + \\ &g) - \frac{D_z}{m} \end{aligned}$$

Since D_z is unknown, we could substitute D_z by $\hat{D}_z$, then:

$$\begin{aligned} U_1 = &\frac{m}{cos(x_3)cos(x_1)}(x_{7d} - z_7 - c_8 z_8 - c_7 z_8 + c_7^2 z_7 + \\ &g) - \frac{\hat{D}_z}{m} \end{aligned}$$

To declare the adaptive law of $\hat{D}_z$, set the Lyapunov function as$V_8 = V_7 + \frac{1}{2} z_8^2 + \frac{\tilde{D_z}^2}{2m\lambda}\ \lambda$, $(\lambda > 0)$, then we could have its derivative as:

$$\begin{aligned} \dot{V_8} &= z_8 \dot{z_8} + \dot{V_7} - \frac{1}{m\lambda} \tilde{D_z} \dot{\hat{D}_z} \\ &= -c_7 z_7^2 - c_8 z_8^2 + \frac{1}{m} z_8 (D_z - \hat{D_z}) - \frac{1}{m\lambda} \tilde{D_z} \dot{D_z} \\ &= -c_7 z_7^2 - c_8 z_8^2 + \frac{\hat{D_z}}{m}(z_8 - \frac{1}{\lambda} \dot{\hat{D}_z}) \end{aligned}$$

where $\dot{\hat{D}_z} = \lambda_1 z_8$, $(\lambda > 0)$

Thus, we could keep Lyapunov function $\dot{V_8} = -c_7 z_7^2 - c_8 z_8^2 < 0$,$(c_7, c_8 > 0)$ and guarantee that $z_7, z_8, \tilde{D_z}$ are asymptotically stable.

Same reasoning, we could apply adaptive backstepping method on D_x and D_y:

$$\begin{cases} u_x = & \frac{m}{U_1}[\ddot{x_{9d}} + (c_9^2 - 1)z_9 - (c_9 + c_{10})z_{10} - \frac{\hat{D_x}}{m}] \\ u_y = & \frac{m}{U_1}[\ddot{x_{11d}} + (c_{11}^2 - 1)z_{11} - (c_{11} + c_{12})z_{12} - \frac{\hat{D_y}}{m}] \end{cases} \tag{12}$$

where

$$\begin{cases} \dot{\hat{D}_x} = \lambda_2 z_{10} & (\lambda_2 > 0) \\ \dot{\hat{D}_z} = \lambda_3 z_{12} & (\lambda_3 > 0) \end{cases} \tag{13}$$

$$\begin{aligned} z_9 &= x - x_d = x_9 - x_{9d} & z_{11} &= y - y_d = x_{11} - x_{11d} \\ z_{10} &= x_{10} - \dot{x_{9d}} + c_9 z_9 & z_{12} &= x_{12} - \dot{x_{11d}} + c_{11} z_{11} \end{aligned} \tag{14}$$

Then, put (12),(13),(14)into (9) and (11), we could solve for the desired rolling angle and pitch angle adaptive to unknown disturbances.

References

[1] F. Bu, Y. Cai, and Y. Yang. Multiple object tracking based on faster-rcnn detector and kcf tracker. Technical Report [Online], 2016. Available at: `https://pdfs.semanticscholar.org`.

[2] Dinesh D. Dhadekar, Prithvi D. Sanghani, K. K. Mangrulkar, and S. E. Talole. Robust Control of Quadrotor using Uncertainty and Disturbance Estimation. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 101(3):1–21, mar 2021.

[3] Shweta Gupte, Paul Infant Teenu Mohandas, and James M. Conrad. A survey of quadrotor unmanned aerial vehicles. In *2012 Proceedings of IEEE Southeastcon*, Orlando, FL, USA, mar 2012. IEEE.

[4] Raja Jarray, Mujahed Al-Dhaifallah, Hegazy Rezk, and Soufiene Bouallègue. Path Planning of Quadrotors in a Dynamic Environment Using a Multicriteria Multi-Verse Optimizer. *Computers, Materials & Continua*, 69(2):2159–2180, jul 2021.

[5] Zhenyue Jia, Jianqiao Yu, Yuesong Mei, Yongbo Chen, Yuanchuan Shen, and Xiaolin Ai. Integral backstepping sliding mode control for quadrotor helicopter under external uncertain disturbances. *Aerospace Science and Technology*, 68:299–307, sep 2017.

[6] Hongyi Li, Lijie Wang, Haiping Du, and Abdesselem Boulkroune. Adaptive fuzzy backstepping tracking control for strict-feedback systems with input delay. *IEEE Transactions on Fuzzy Systems*, 25(3):642–652, jun 2017.

[7] Omid Mofid, Saleh Mobayen, Chunwei Zhang, and Balasubramanian Esakki. Desired tracking of delayed quadrotor UAV under model uncertainty and wind disturbance using adaptive super-twisting terminal sliding mode control. *ISA Transactions*, 123:455–471, apr 2022.

[8] Daniel D. Timis, Cristina I. Muresan, and Eva H. Dulf. Design and Experimental Results of an Adaptive Fractional-Order Controller for a Quadrotor. *Fractal and Fractional*, 6(4):204, apr 2022.

[9] Yuanyuan Xu and Bing Chen. Adaptive neural network control for nonlinear non-strict feedback time-delay systems. *Systems Science & Control Engineering*, 9(S2):81–92, 2021.

[10] Nguyen Xuan-Mung and Sung Kyung Hong. Robust Backstepping Trajectory Tracking Control of a Quadrotor with Input Saturation via Extended State Observer. *Applied Sciences*, 9(23):5184, nov 2019.

[11] Y. Yang. *Electromechanical Characterization of Organic Field-Effect Transistors with Generalized Solid-State and Fractional Drift-Diffusion Models*. PhD thesis, Purdue University, 2021.

[12] Y. Yang and H. H. Zhang. Stability study of lqr and pole-placement genetic algorithm synthesized input-output feedback linearization controllers for a rotary inverted pendulum system. *International Journal of Engineering Innovations and Research*, 7(1):62–68, 2018.

[13] Y. Yang and H. H. Zhang. Control applications in engineering and technology. In *Fractional Calculus with its Applications in Engineering and Technology*, pages 67–89. Springer International Publishing, Cham, 2019.

[14] Y. Yang and H. H. Zhang. *Fractional calculus with its applications in engineering and technology*. Morgan & Claypool Publishers, 2019.

[15] Y. Yang and H. H. Zhang. Fractional-order controller design. In *Fractional Calculus with its Applications in Engineering and Technology*, pages 43–65. Springer International Publishing, Cham, 2019.

[16] Y. Yang and H. H. Zhang. Preliminary tools of fractional calculus. In *Fractional Calculus with its Applications in Engineering and Technology*, pages 3–42. Springer International Publishing, Cham, 2019.

[17] Y. Yang and H. H. Zhang. Neural network-based adaptive fractional-order backstepping control of uncertain quadrotors with unknown input delays. *Fractal and Fractional*, 7(3):232, 2023.

[18] Y. Yang and H. H. Zhang. Optimal model reference adaptive fractional-order proportional integral derivative control of idle speed system under varying disturbances. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, page 09596518241266670, 2024.

[19] Y. Yang, H. H. Zhang, and R. M. Voyles. Rotary inverted pendulum system tracking and stability control based on input-output feedback linearization and pso-optimized fractional order pid controller. In *Automatic Control, Mechatronics and Industrial Engineering*, pages 79–84. CRC Press, 2019.

[20] Y. Yang, H. H. Zhang, and R. M. Voyles. Optimal fractional-order proportional–integral–derivative control enabling

full actuation of decomposed rotary inverted pendulum system. *Transactions of the Institute of Measurement and Control*, 45(10):1986–1998, 2023.

[21] Y. Yang, H. H. Zhang, W. Yu, and L. Tan. Optimal design of discrete-time fractional-order pid controller for idle speed control of an ic engine. *International Journal of Powertrains*, 9(1-2):79–97, 2020.

[22] Yi Yang. Prediction of hotel price with regression models. Technical report, School of Public Health, University of Michigan, USA, 2023. Presented at a Machine Learning and Biostatistics Online Course, April 2023. Available at: `https://yeeyoung.github.io/assets/regression_english.pdf`.

[23] Yi Yang. Voorspelling van hotelprijzen met regressiemodellen. Technisch rapport, School of Public Health, Universiteit van Michigan, VS, 2023. Gepresenteerd tijdens een online cursus Machine Learning en Biostatistiek, april 2023. Beschikbaar op: `https://yeeyoung.github.io/assets/regression_dutch.pdf`.

[24] Yi Yang. Identificatie van fractionele-orde systeemdynamica voor een roterende omgekeerde slinger. Technisch verslag, Afdeling Werktuigbouwkunde, Universiteit van Michigan, VS, 2024. Gepresenteerd tijdens een online cursus Mechatronica en Robotica, december 2024. Beschikbaar op: `https://yeeyoung.github.io/assets/Identification_dutch.pdf`.

[25] Yi Yang. Identification of fractional-order system dynamics for a rotary inverted pendulum. Technical report, Department of Mechanical Engineering, University of Michigan, USA, 2024. Presented at a Mechatronics and Robotics Online Course, December 2024. Available at: `https://yeeyoung.github.io/assets/Identification_english.pdf`.

[26] Yongliang Zhan, Xiaomei Li, and Shaocheng Tong. Observer-Based Decentralized Control for Non-Strict-Feedback Fractional-Order Nonlinear Large-Scale Systems With Unknown Dead Zones. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[27] Dongliang Zheng, Hesheng Wang, Jingchuan Wang, Siheng Chen, Weidong Chen, and Xinwu Liang. Image-based visual servoing of a quadrotor using virtual camera approach. *IEEE/ASME Transactions on Mechatronics*, 22(2):972–982, apr 2017.