System Design of Quadrotor

Yukai Gong, Weina Mao, Bu Fan, Yi Yang

Mar. 29, 2016

A final project of MECHENG 561.

Supervised by Prof. Vasudevan.
Abstract

In this report, an autonomous quadrotor is designed. Based on two control techniques drawn from a classical PID scheme in classical control theory and a backstepping control scheme in nonlinear control theory, we proposed a simplified mathematical model and conduct a simulation in Simulink. Different perturbation has been exerted to the system, the performance in stability and robustness has been analyzed and compared between two different controller schemes. For simplicity of the project, we do not involve an experimental validation even if we can consider to construct a real quadrotor controlled by an embedded Arduino Board for the next step.
# Contents

1 Introduction ........................................... 1  
   1.1 Background ........................................... 1  
   1.2 Aerodynamic Modelling of Quadrotor Aircraft .......... 1

2 System Design Based on PID Method .................... 2  
   2.1 Design the PID Controller ............................ 2  
   2.2 Simulation Results and Analysis ..................... 3  
   2.3 Summary ............................................... 5

3 System Design Based on Backstepping Method .......... 6  
   3.1 Backstepping controller design ....................... 6  
      3.1.1 Simulation of Backstepping control with simplified inputs ........................................... 6  
      3.1.2 Simulation of Backstepping control with real inputs ...................................................... 6  
      3.1.3 Simulation of Backstepping control in discrete time ..................................................... 7  
      3.1.4 Backstepping control in discrete time under disturbance ............................................... 8  
      3.1.5 Simulation of Adaptive Backstepping control in discrete time under disturbances ............... 9  
      3.1.6 Simulation of Adaptive Backstepping control with noise ................................................. 10

4 Comparison ............................................. 10

A Backstepping control .................................... 11  
   A.1 Backstepping attitude control ......................... 11  
      A.1.1 Backstepping position control ....................... 12  
   A.2 Adaptive Backstepping control in discrete time under disturbance ...................................... 13
1 Introduction

1.1 Background

Flying objects have always exerted a great fascination on man encouraging all kinds of research and development. [1] The important recent technological progress in sensors, actuators, processors and power storage devices represents a real jump ahead, which enable the emergence of new applications like the indoor micro aerial robots. Compared with the other flying principles, quadrotor systems have specific characteristics which allow the execution of applications that would be difficult or impossible otherwise, such as building surveillance and intervention in hostile environments.

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. [2] Let \([\phi, \dot{\phi}, \theta, \dot{\theta}, \psi, \dot{\psi}, x, \dot{x}, y, \dot{y}, z, \dot{z}] = [x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}]\). The original nonlinear dynamic model for this system is provided as:

\[
\begin{align*}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= \frac{(I_y - I_z)}{I_x} x_4 x_6 + \frac{U_2 l}{I_z} \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= \frac{(I_z - I_x)}{I_y} x_2 x_4 + \frac{U_3 l}{I_y} \\
\dot{x}_5 &= x_6 \\
\dot{x}_6 &= \frac{(I_x - I_y)}{I_z} x_2 x_4 + \frac{U_4 l}{I_z} \\
\dot{x}_7 &= x_8 \\
\dot{x}_8 &= \cos(x_1) \cos(x_3) \frac{U_1}{m} - g \\
\dot{x}_9 &= x_{10} \\
\dot{x}_{10} &= (\sin(x_3) \cos(x_1) \cos(x_5) + \sin(x_1) \sin(x_5)) \frac{U_1}{m} \\
\dot{x}_{11} &= x_{12} \\
\dot{x}_{12} &= (\sin(x_3) \cos(x_1) \sin(x_5) - \sin(x_1) \cos(x_5)) \frac{U_1}{m}
\end{align*}
\]

where

\[
\begin{align*}
U_1 &= b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\
U_2 &= b(\Omega_3^2 - \Omega_2^2) \\
U_3 &= b(\Omega_4^2 + \Omega_1^2) \\
U_4 &= d(-\Omega_1^2 + \Omega_2^2 - \Omega_2^2 + \Omega_4^2)
\end{align*}
\]

We choose the system model parameters as follows:

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(g)</td>
<td>(m)</td>
<td>(L)</td>
<td>(I_x)</td>
<td>(I_y)</td>
<td>(I_z)</td>
<td>(d)</td>
</tr>
<tr>
<td>9.81</td>
<td>0.53</td>
<td>0.232</td>
<td>6.228e - 3</td>
<td>6.228e - 3</td>
<td>1.121e - 2</td>
<td>7.5e - 7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3.13e - 5</td>
</tr>
</tbody>
</table>
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:

![System with controller](image)

According to this semi-coupled relation, we can first design the inner loop (attitude control) and then outer loop (position control), as shown in Figure 2b. 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.\[^3\] \((x_d, y_d, z_d)\) is the reference target position, \((\phi_d, \theta_d, \psi_d)\) is the expected value of our inner loop subsystem.

![PID control system](image)

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

Figure 2: PID control system

First we simplify the dynamic equation

\[
x_{10} = U_x \frac{U_i}{m} \\
x_{12} = U_y \frac{U_i}{m}
\]

where

\[
U_x = \sin(x_3)\cos(x_1)\sin(x_5) - \sin(x_1)\cos(x_5) \\
U_y = \sin(x_3)\cos(x_1)\cos(x_5) + \sin(x_1)\sin(x_5)
\]
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

$$x_{1d} = \arcsin(u_x \sin(x_5) - u_y \cos(x_5))$$

$$x_{3d} = \arcsin\left[\frac{u_x - \sin(x_1) \sin(x_5)}{\cos(x_1) \cos(x_5)}\right]$$

Use desired $x_{1d}$ and $x_{3d}$ as reference for the Angle Controller to compute the desired $U_{1d}$ $U_{2d}$ $U_{3d}$ and $U_{4d}$. Finally compute the needed voltage, which is the real input.

The PID parameter we have chosen is shown in Table 1. 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.

<table>
<thead>
<tr>
<th></th>
<th>$K_p$</th>
<th>$K_i$</th>
<th>$K_d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\phi$</td>
<td>0.5</td>
<td>0.1</td>
<td>0.15</td>
</tr>
<tr>
<td>$\theta$</td>
<td>0.5</td>
<td>0.1</td>
<td>0.15</td>
</tr>
<tr>
<td>$\psi$</td>
<td>0.1</td>
<td>0.05</td>
<td>0.1</td>
</tr>
<tr>
<td>x</td>
<td>2</td>
<td>0.25</td>
<td>3</td>
</tr>
<tr>
<td>y</td>
<td>2</td>
<td>0.35</td>
<td>3</td>
</tr>
<tr>
<td>z</td>
<td>1</td>
<td>0.2</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 1: 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.

![Bode plot of the linearized system](image)

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.

![Figure 7: PID control in discrete time under disturbance](image)

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.

![Figure 8: PID control with noise](image)

### 2.3 Summary

In conclusion, the PID controller perform well in both continuous and discrete case. It also show robustness under disturbances and noises.
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[4].

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

\[
X = \begin{bmatrix}
\phi & \dot{\phi} & \theta & \dot{\theta} & \psi & \dot{\psi} & z & \dot{z} & x & \dot{x} & y & \dot{y}
\end{bmatrix}^T = [0.4, 0, 0.3, 0, 0.2, 0, 0, 0, 0, 0, 0, 0]^T.
\]

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)

![Figure 9: Backstepping control system](image)

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 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 to stay steady after time 0. Fortunately, though the rotor conducted a minor delay of the rotating speed, the response time of the system was 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 $0.01\text{rad} \approx 0.573\text{deg}$ 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.

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 $t=15$s and ends at $t=20$s. An additional weight $m=0.5\text{kg}$ will be add on the quadrotor at $t=15$ and maintained until $t=30$s. Simulation results are shown in Figure 14.

(Readers could run run_backstepping_discrete_disturb.m file first and then run backstepping_real_discrete_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 additional 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 nonadaptive 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_discrete_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 0.01 rad $\approx 0.05$ deg on rolling angle $\phi$ and pitch angle $\theta$.

(Readers could run `run_backstepping_discrete_disturb.m` file first and then run `backstepping_real_discrete_disturb_adp_noise.slx` file)

4 Comparison

<table>
<thead>
<tr>
<th>Response Properties</th>
<th>PID</th>
<th>Backstepping</th>
</tr>
</thead>
<tbody>
<tr>
<td>time to the expected altitude</td>
<td>5s</td>
<td>4s</td>
</tr>
<tr>
<td>time to the target horizontal position</td>
<td>15s</td>
<td>4s</td>
</tr>
<tr>
<td>time to recover from disturbance</td>
<td>10s</td>
<td>5s</td>
</tr>
</tbody>
</table>

Table 2: Performance of Different Methods

From the comparison we can find that the Backstepping has a more keen response than PID. What's 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.
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 $\phi, \theta, \psi$ respectively. Take the rolling angle $\phi$ as an example, we could deduce the expression of $U_2$ by Backstepping method:

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

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

**Step 1**

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:

$$
z_1 = \dot{x}_1 - x_{1d} = x_2 - x_{1d}
$$

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

$$
\dot{V}_1 = z_1 = z_1(x_2 - x_{1d}) = z_1(z_2 + \alpha_1 - x_{1d})
$$

According to Lyapunov stability criterion, we need to choose $\alpha_1 = x_{1d} - c_1 z_1$ such that $\dot{V}_1 < 0$, where $c_1 > 0$. Then we could have:

$$
\dot{V}_1 = -c_1 z_1^2 + z_1 z_2
$$

$$
z_2 = x_2 - x_{1d} + c_1 z_1
$$

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

$$
\dot{V}_2 = z_2 \ddot{z}_2 + V_1 = z_2(\dot{x}_2 - \dot{x}_{1d} + c_1 \dot{z}_1) - c_1 z_1^2 + z_1 z_2
$$

$$
= z_2(b_1 U_2 + a_1 x_4 x_6 - x_{1d} + c_1 z_1) - c_1 z_1^2 + z_1 z_2
$$

$$
= z_2(b_1 U_2 + a_1 x_4 x_6 - x_{1d} + c_1(z_2 - c_1 z_1)) - c_1 z_1^2 + z_1 z_2
$$

Then, the control input should be:

$$
U_2 = \frac{1}{b_1} (x_{1d} - z_1 - a_1 x_4 x_6 - c_1 z_2 + c_1^2 z_1 - c_1 z_2)
$$

$$
= \frac{1}{b_1} (x_{1d} + (c_1^2 - 1) z_1 - a_1 x_4 x_6 - (c_1 + c_2) z_2)
$$

where $c_2 > 0, x_{1d} = 0, z_1 = x_1 - x_{1d}, z_2 = x_2 - x_{1d} + c_1 z_1$. Put (2) into (1), we have:

$$
\dot{V}_2 = -c_1 z_1^2 - c_2 z_2^2 < 0
$$

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 $\theta$ and yaw angle $\psi$ as:

$$
\begin{cases}
  U_3 = \frac{1}{b_2} [x_{3d} + (c_3^2 - 1) z_3 - a_2 x_2 x_6 - (c_3 + c_4) z_4] \\
  U_4 = \frac{1}{b_3} [x_{3d} + (c_5^2 - 1) z_5 - a_3 x_2 x_4 - (c_5 + c_6) z_6]
\end{cases}
$$
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 \( \phi \) and \( \theta \), we could define some virtual input \( U_x, U_y \), expressed by \( \phi \) and \( \theta \), 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:

\[
\begin{cases}
    x^7 = x_8 \\
    x^8 = -g + (\cos(x_1)\cos(x_3)) \frac{U_1}{m}
\end{cases}
\]  

(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:

\[
z_7 = x_7 - x_7d = x_8 - x_8d
\]

Then, define \( a_7 \), where \( z_8 = x_8 - a_7 \), and choose the Lyapunov function \( V_7 = \frac{1}{2}z_7^2 \), then:

\[
V_7 = z_7z_8 = z_7(x_8 - x_8d) + z_7z_8
\]

To ensure \( z_7 \) asymptotically stable at the origin, according to Lyapunov stability criterion, we should have \( V_7 < 0 \), i.e. \( a_7 = x_8d - c7z_7(c7 > 0) \). Thus:

\[
\dot{V}_7 = -c_7z_7^2 + z_7z_8
\]

\[
z_8 = x_8 - x_8d + c_7z_7
\]

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 \( z_8 = x_8 - x_8d + c_7z_7 \), then:

\[
V_8 = z_8z_8 + V_1 = z_8(x_8 - x_8d + c_7z_7) - c_7z_7^2 + z_7z_8
\]

(4)

\[
= z_8[x_8 - x_8d + c_7(z_8 - c_7z_7)] - c_7z_7^2 + z_7z_8
\]

\[
= -c_7z_7^2 + z_8[z_7 - x_8d + c_7(z_8 - c_7z_7) + x_8]
\]

To guarantee \( \dot{V}_8 < 0 \), choose:

\[
z_7 - x_8d + c_7(z_8 - c_7z_7) + x_8 = -c_8z_8(c_8 > 0)
\]

(5)

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

\[
\dot{V}_8 = -c_7z_7^2 + z_8[z_7 - x_8d + c_7z_7 + x_8]
\]

\[
= -c_7z_7^2 - c_8z_8^2 < 0
\]

Moreover, (5) can be written as:

\[
x_8 = x_8d + (c_7^2 - 1)z_7 - (c_7c_8)z_8(c_7 > 0, c_8 > 0)
\]

(6)
Put (6) into (3):

$$U_1 = \frac{m}{\cos(x_1)\cos(x_3)}[x_7d + (c_2^2 - 1)z_7 + g - (c_7 + c_8)z_8]$$

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:

$$\begin{align*}
    u_x &= \frac{\dot{x}_0d + (c_2^2 - 1)z_9 - (c_9 + c_{10})z_{10}}{\cos(x_1)\cos(x_3)} \\
    u_y &= \frac{\dot{x}_{11d} + (c_{11}^2 - 1)z_{11} - (c_{11} + c_{12})z_{12}}{\cos(x_1)\cos(x_3)}
\end{align*}$$

(7)

where

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

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 $\phi_d, \theta_d$ and $u_x, u_y$:

$$\begin{align*}
    \phi_d &= \arcsin(u_x \sin(\psi) - u_y \cos(\psi)) = \arcsin(u_x \sin(x_5) - u_y \cos(x_5)) \\
    \theta_d &= \arcsin\left(\frac{u_x - \sin(\phi)\sin(\psi)}{\cos(\phi)\cos(\psi)}\right) = \arcsin\left(\frac{u_x - \sin(x_1)\sin(x_5)}{\cos(x_1)\cos(x_5)}\right)
\end{align*}$$

(8) (9)

Put (8) and (9) into (7), we could deduce the final expression of inputs:

$$\begin{align*}
    U_1 &= \frac{m}{\cos(x_1)\cos(x_3)}[x_7d + (c_2^2 - 1)z_7 + g - (c_7 + c_8)z_8] \\
    U_2 &= \frac{1}{\cos(x_1)\cos(x_3)}[x_{11d} + (c_{11}^2 - 1)z_{11} - (c_{11} + c_{12})z_{12}] \\
    U_3 &= \frac{1}{\cos(x_1)\cos(x_3)}[x_{3d} + (c_2^2 - 1)z_3 - a_1x_4x_6 - (c_1 + c_2)z_2] \\
    U_4 &= \frac{1}{\cos(x_1)\cos(x_3)}[x_{5d} + (c_2^2 - 1)z_5 - a_3x_2x_4 - (c_5 + c_6)z_6]
\end{align*}$$

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 the position subsystem as following:

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

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:

$$\begin{align*}
    \dot{x}_7 &= x_8 \\
    \dot{x}_8 &= \frac{1}{m}(\cos(x_3)\cos(x_1)U_1 - mg + D_z)
\end{align*}$$

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 $\tilde{D}_z = D_z - \hat{D}_z$. 

13
For a given desired altitude $z_d = x_{7d}$, define the altitude error variable as $z_7 = x_7 - x_{7d}$, speed error variable $z_8 = x_8 - a_7$, where $a_7$ is a virtual control variable.

**Step 1**: design for $z_7$ subsystem.

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

Take $x_8$ as the control variable, if $x_8 = x_{7d} - c_7 z_7$, then $z_7$ subsystem is asymptotically stable. Since $x_8$ is a transient control variable, we can only express $x_8$ by $a_7$. Thus, design:

$$a_7 = x_{7d} - c_7 z_7 (c_7 > 0)$$

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

$$V_7 = z_7 \dot{z}_7 = z_7 (x_8 - x_{7d})$$

$$= z_7 (z_8 + a_7 - x_{7d})$$

$$= z_7 z_8 - c_7 z_7^2$$

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.

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

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

$$V_8 = z_8 \dot{z}_8 + \dot{V}_7 = z_8 (x_8 - x_{7d} + c_7 z_7) - c_7 z_7^2 + z_7 z_8$$

$$= z_8 [x_8 - x_{7d} + c_7 (z_8 - c_7 z_7)] - c_7 z_7^2 + z_7 z_8$$

$$= z_8 \left[ \frac{1}{m} (\cos(x_3)\cos(x_1) U_1 - mg + D_z) - x_{7d} + c_7 (z_8 - c_7 z_7) \right] - c_7 z_7^2 + z_7 z_8$$

Now, the system input:

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

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

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

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

$$\dot{V}_8 = z_8 \dot{z}_8 + \dot{V}_7 - \frac{1}{m \lambda} \hat{D}_z \dot{\hat{D}}_z$$

$$= -c_7 z_7^2 - c_8 z_8^2 + \frac{1}{m} z_8 (D_z - \dot{\hat{D}}_z) - \frac{1}{m \lambda} \hat{D}_z \dot{\hat{D}}_z$$

$$= -c_7 z_7^2 - c_8 z_8^2 + \frac{\hat{D}_z}{m} (z_8 - \frac{1}{\lambda} \dot{\hat{D}}_z)$$

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 $z_7, z_8, \hat{D}_z$ are asymptotically stable.
Same reasoning, we could apply adaptive backstepping method on $D_x$ and $D_y$:

$$\begin{align*}
  u_x &= \frac{m}{T} \left[ x_9d + (c_9^2 - 1)z_9 - (c_9 + c_{10})z_{10} - \frac{D_x}{m} \right] \\
  u_y &= \frac{m}{T} \left[ x_{11d} + (c_{11}^2 - 1)z_{11} - (c_{11} + c_{12})z_{12} - \frac{D_y}{m} \right]
\end{align*}$$

where

$$\begin{align*}
  \dot{D}_x &= \lambda_2 z_{10} \quad (\lambda_2 > 0) \\
  \dot{D}_z &= \lambda_3 z_{12} \quad (\lambda_3 > 0)
\end{align*}$$

Then, put (10),(11),(12) into (8) and (9), we could solve for the desired rolling angle and pitch angle adaptive to unknown disturbances.

**Reference**


