Comparing and Developing PID and Sliding Mode Controllers for Quadrotor

Ceren Cömert and Coşku Kasnakoğlu
Electrical and Electronics Engineering, TOBB University of Economics and Technology, Ankara, Turkey
Email: {ccomert, kasnakoglu}@etu.edu.tr

Abstract—This paper provides information about controller design for quadrotors which became an essential research topic. Sliding mode control (SMC) provides better solutions to problems that can be caused by model discrepancies. Firstly, sliding mode controllers are designed for altitude, roll, pitch and yaw angle controls. One common problem of sliding mode control, chattering is eliminated. Then, proportional integral derivative (PID) controller is developed in order to compare and discuss simulation results. Coefficients are tuned by iteration method in both PID and SMC designs. It can be interpreted that the controllers of both designs are capable of tracking the desired values.

Index Terms—unmanned aerial vehicle, quadcopter, sliding mode control, quadrotor, proportional integral derivative, nonlinear control

I. INTRODUCTION

Unmanned aerial vehicles became an essential research topic due to their vast application areas. Search and rescue, transportation, fire monitoring, military and defense, crop spraying and are some of the areas that they can be used. Because of their maneuverability, quadrotor are often used for controller designs [1]-[4].

Quadrotors are unmanned aerial vehicles whose four rotors can be controlled separately. This situation gives the capability to move rotational and translational in three dimensional space called six degrees of freedom (6DOF).

Fig. 1 demonstrates quadrotors’ movements and angles which occur between earth and body fixed frames [5]-[9].

It is possible to make transition between earth frame and body frame.

Equation (1) through “(4)” show the transformation between two referential frames [10], [11].

\[ \begin{bmatrix}
    x \\
    y \\
    z \\
\end{bmatrix} = R(\psi)R(\theta)R(\phi) \begin{bmatrix}
    u \\
    v \\
    w \\
\end{bmatrix} \]

(1)

\[ R(\psi) = \begin{bmatrix}
    \cos(\psi) & -\sin(\psi) & 0 \\
    \sin(\psi) & \cos(\psi) & 0 \\
    0 & 0 & 1 \\
\end{bmatrix} \]

(2)

\[ R(\theta) = \begin{bmatrix}
    \cos(\theta) & 0 & \sin(\theta) \\
    0 & 1 & 0 \\
    -\sin(\theta) & 0 & \cos(\theta) \\
\end{bmatrix} \]

(3)

\[ R(\phi) = \begin{bmatrix}
    1 & 0 & 0 \\
    0 & \cos(\phi) & -\sin(\phi) \\
    0 & \sin(\phi) & \cos(\phi) \\
\end{bmatrix} \]

(4)

A. Force and Torque Formulas

Equation (5) shows inertia matrix of the quadrotor which will be used for attitude controller design. Inertia matrix is used to determine the torque needed for a desired angular acceleration about a rotational axis [12].

\[ J = \begin{bmatrix}
    J_x & 0 & 0 \\
    0 & J_y & 0 \\
    0 & 0 & J_z \\
\end{bmatrix} \]

(5)

Using Newton-Euler law, the force and torque formulas can be defined in “(6)” and “(7)”.

In these equations F represents force, \( \tau \) represents torque and m represents mass. u, v and w are velocities, while p, q, and r stand for angular rates[5], [13], [14]

\[ \begin{bmatrix}
    F_x \\
    F_y \\
    F_z \\
\end{bmatrix} = m \begin{bmatrix}
    u \\
    v \\
    w \\
\end{bmatrix} + m \begin{bmatrix}
    qw - rv \\
    ru - pw \\
    pv - qu \\
\end{bmatrix} \]

(6)

\[ \begin{bmatrix}
    \tau_x \\
    \tau_y \\
    \tau_z \\
\end{bmatrix} = \begin{bmatrix}
    pq(l_y - l_z) \\
    pr(l_z - l_x) \\
    pq(l_x - l_y) \\
\end{bmatrix} \]

(7)

B. Transformation Matrix

Equation (8) and “(9)” show the transformation matrix [15], [16].

\[ \begin{bmatrix}
    \phi \\
    \theta \\
    \psi \\
\end{bmatrix} = \begin{bmatrix}
    1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\
    0 & \cos(\phi) & -\sin(\phi) \\
    0 & \sin(\phi) / \cos(\theta) & \cos(\phi) / \cos(\theta) \\
\end{bmatrix} \begin{bmatrix}
    p \\
    q \\
    r \\
\end{bmatrix} \]

(8)
\[
\begin{bmatrix}
p \\
q \\
r
\end{bmatrix} = \begin{bmatrix} 1 & 0 & -\sin(\theta) \\
0 & \cos(\phi) & \cos(\theta)\sin(\phi) \\
0 & -\sin(\phi) & \cos(\theta)\cos(\phi) \end{bmatrix} \begin{bmatrix} \phi \\
\theta \\
\psi \end{bmatrix}
\] (9)

\[s = \dot{e} + \lambda e \] (13)

\[\dot{s} = \ddot{e} + \lambda \ddot{e} = \ddot{x} - \ddot{x}_d = f(x) + u - x_d + \lambda e \] (14)

Sliding condition is defined in “(15)”. If a controller is designed to achieve \(s\) zero and \(f(x)\) predicted as \(\Gamma(x)\) then the new controller input is given in “(16)”, [24], [25].

\[\frac{1}{2} \frac{d}{dt}(s^2) \leq -\eta |s| \] (15)

\[u = -\hat{f}(x) + x_d \cos(\phi) - \lambda \ddot{e} - k_1 s - k_2 \text{sign}(s) \] (16)

Equation (17) defines sign function.

\[\text{sign}(s) = \begin{cases} 
-1 & s < 0 \\
1 & s > 0 
\end{cases} \] (17)

Derivative of the sliding surface is obtained as in “(18)”, [24], [26], [27].

\[\dot{s} = -k_1 s - k_2 \text{sign}(s) \] (18)

A. Altitude Controller

Equation (19) represent quadrotor dynamics for altitude. By equating “(14)” and “(18)” and substituting \(\ddot{z}\) from “(19)”, the input for the altitude controller is calculated as in “(20)” [24], [28], [29].

\[\ddot{z} = g - U_1 \cos(\phi)\cos(\theta)/m \] (19)

\[U_1 = \frac{m}{\cos(\phi)\cos(\theta)} k_1 s + k_2 \text{sign}(s) + \lambda(\dot{z}_{\text{ref}} - \dot{z}) + g - z_{\text{ref}} \] (20)

The MATLAB Simulink model designed for the altitude control, is shown in Fig. 3.

Table I shows sliding mode coefficients for altitude control.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(k_1)</td>
<td>1</td>
</tr>
<tr>
<td>(k_2)</td>
<td>12</td>
</tr>
<tr>
<td>(\lambda)</td>
<td>20</td>
</tr>
</tbody>
</table>
\section*{B. Attitude Controller}

Dynamic model equations for roll, pitch and yaw are given in the \textit{"(21)" through "(23)"} respectively. Controller inputs are obtained by substituting $\ddot{\phi}$, $\ddot{\theta}$ and $\ddot{\psi}$ from these three equations.

\begin{align}
\ddot{\phi} &= \frac{d}{dx} U_2 - \frac{J_m}{J_x} \dot{\theta} + \frac{J_y - J_z}{J_x} \dot{\psi} \\
\ddot{\theta} &= \frac{d}{dy} U_3 - \frac{J_m}{J_y} \dot{\phi} + \frac{J_z - J_x}{J_y} \dot{\psi} \\
\ddot{\psi} &= \frac{d}{dz} U_4 + \frac{J_x - J_y}{J_z} \dot{\phi} 
\end{align}

Sliding mode controller inputs for roll angle (phi), pitch angle (theta) and yaw angle (psi) are given in the \textit{"(24)"}, \textit{"(25)"} and \textit{"(26)"} respectively \cite{28}, \cite{30}, \cite{31}.

\begin{align}
U_2 &= \frac{J_x}{L} [k_1 + k_2 \text{sign}(s) + \lambda (\dot{\phi}_{\text{ref}} - \dot{\phi}) + \phi_{\text{ref}}^\prime] + \frac{J_m}{J_x} \dot{\theta} + \frac{J_y - J_z}{J_x} \dot{\psi} \\
U_3 &= \frac{J_y}{L} [k_1 + k_2 \text{sign}(s) + \lambda (\dot{\theta}_{\text{ref}} - \dot{\theta}) + \theta_{\text{ref}}^\prime] + \frac{J_m}{J_y} \dot{\phi} + \frac{J_z - J_x}{J_y} \dot{\psi} \\
U_4 &= \frac{J_z}{L} [k_1 + k_2 \text{sign}(s) + \lambda (\dot{\psi}_{\text{ref}} - \dot{\psi}) + \psi_{\text{ref}}^\prime] - \frac{J_x - J_y}{J_z} \dot{\phi} 
\end{align}

\section*{C. Chattering Reduction}

Chattering problem often occurs in sliding mode designs. One reason for chattering to occur is fast dynamics which are neglected by model causing fast switching of sliding mode controllers \cite{32}, \cite{33}. Chattering effect is demonstrated in Fig. 4.

A common way to solve this problem is to introduce a boundary layer around the surface \cite{34}. This method provides a smooth continuous approach to the discontinuous sign function by a continuous function \cite{25}. The most commonly used functions for chattering elimination are saturation \cite{34}, \cite{35} and sigmoid \cite{23} functions. Saturation and sigmoid functions are given \textit{"(27)"} and \textit{"(28)"} respectively.

\begin{align}
sat\left(\frac{s}{\varphi}\right) &= \begin{cases} 
\frac{s}{\varphi} & \text{if } s \leq 1 \\
\text{sign}\left(\frac{s}{\varphi}\right) & \text{if } s > 1
\end{cases} \\
sigmoid(s) &= \frac{s}{|s| + \varepsilon}
\end{align}

$\varepsilon$ is positive constant and $\varphi$ represents boundary stickness.

In order to reduce chattering sign function in the altitude control model is changed with the sigmoid function, while the ones in the attitude control models (phi, theta and psi models) are changed with the saturation function.

Because of overshoot increase in phi and theta, an integrator is added to the sliding surface as illustrated in the \textit{"(29)"} and \textit{"(30)"} for the roll and pitch angle control \cite{24}, \cite{35}.

\begin{align}
s &= (d/dt + \lambda) n^{-1} \int e \, dt \\
s &= \dot{e} + 2\lambda e + \lambda^2 \int e \, dt
\end{align}

Simulink model used for roll, pitch and angle control is shown in Fig. 5.

\begin{table}[ht]
\centering
\caption{SMC COEFFICIENTS FOR ATTITUDE CONTROL}
\begin{tabular}{|c|c|c|c|}
\hline
\textbf{Name} & \textbf{Roll} & \textbf{Pitch} & \textbf{Yaw} \\
\hline
$k_1$ & 6 & 6 & 0.1 \\
$k_2$ & 18.5 & 18.6 & 90 \\
$\lambda$ & 0.27 & 0.27 & 2.5 \\
\hline
\end{tabular}
\end{table}

Figure 6. SMC yaw control simulink model.
Simulink model used for yaw angle control is shown in Fig. 6.

III. PID CONTROL

A. Altitude Control

Equation (29) represents PID altitude controller input.

\[ U_1 = K_p(z_{ref} - z) + K_d(z_{ref} - z) + K_i \int (z_{ref} - z) \quad (29) \]

MATLAB Simulink model designed for the altitude control is shown in Fig. 7.

![Figure 7. PID altitude controller Simulink model.](image)

Table III demonstrates PID coefficients for altitude control.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kp</td>
<td>2.6</td>
</tr>
<tr>
<td>Ki</td>
<td>0.5</td>
</tr>
<tr>
<td>Kd</td>
<td>3.7</td>
</tr>
</tbody>
</table>

Fig. 8 shows simulation results for altitude for both C and PID controllers.

![Figure 8. Simulation results for altitude model a) PID, b) SMC](image)

B. Attitude Controller

The proportional-integral-derivative controller inputs for roll angle (\(\phi\)), pitch angle (\(\theta\)) and yaw angle (\(\psi\)) are given in the “(30)” through “(32)” [36].

\[
U_2 = K_p(\phi_{ref} - \phi) + K_d(\dot{\phi}_{ref} - \dot{\phi}) + K_i \int (\phi_{ref} - \phi) \quad (30)
\]

\[
U_3 = K_p(\theta_{ref} - \theta) + K_d(\dot{\theta}_{ref} - \dot{\theta}) + K_i \int (\theta_{ref} - \theta) \quad (31)
\]

\[
U_4 = K_p(\psi_{ref} - \psi) + K_d(\dot{\psi}_{ref} - \dot{\psi}) + K_i \int (\psi_{ref} - \psi) \quad (32)
\]

Simulink model designed for roll angle, pitch angle and yaw angle PID control is shown in Fig. 9.

![Figure 9. PID altitude controller Simulink model.](image)

PID coefficients for attitude control are given in the Table IV.

<table>
<thead>
<tr>
<th>Name</th>
<th>Angle</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kp</td>
<td>Roll</td>
<td>3</td>
</tr>
<tr>
<td>Ki</td>
<td>Pitch</td>
<td>0.1</td>
</tr>
<tr>
<td>Kd</td>
<td>Yaw</td>
<td>0.6</td>
</tr>
</tbody>
</table>

IV. SIMULATION RESULTS FOR ATTITUDE

![Figure 10. Simulation results for roll angle a) PID, b) SMC](image)
and discontinuity is suppressed by using either saturation function or sigmoid function. After chattering is eliminated coefficients are readjusted and an integrator is added to the sliding surface for the roll and pitch angle controllers. The designed sliding mode controllers are capable of following altitude and attitude commands.

In order to reduce steady state error, angular rates (p, q and r) are used as a feedback in PID controller design. As a result, PID controllers are capable of tracking the desired values.

Simulation results reveals that the sliding mode controllers can track the desired values with faster response and less oscillation than the PID controllers.

Future research includes testing the theoretical simulation results on a quadcopter in order to obtain experimental results.

REFERENCES


T. Luukkonen, “Modelling and control of a quadcopter, independent research project in applied mathematics,” Aalto University, Espoo, 2011.


Ceren Cümcü is currently a master student at the Electrical & Electronics Engineering Department, TOBB University of Economics & Technology, Ankara, Turkey. She received the BSc degree in Electrical & Electronics Engineering in 2014 from the TOBB University of Economics & Technology, Ankara, Turkey. Unmanned aerial vehicles and control systems are her current research areas.

Cosku Kasnakoglu obtained B.S. degrees from the Department of Electrical and Electronics Engineering and the Department of Computer Engineering at the Middle East Technical University (METU), Ankara, Turkey in 2000. He obtained his M.S. and Ph.D. degrees from the Department of Electrical and Computer Engineering at the Ohio State University (OSU), Columbus, Ohio, USA in 2003 and 2007. He is currently an associate professor in the Department of Electrical and Electronics Engineering at TOBB University of Economics and Technology in Ankara, Turkey. Dr. Kasnakoglu's current research interests include nonlinear control, flow control, unmanned air vehicles, dynamical modeling, adaptive control and linear parameter varying systems.