Backstepping Control on SE(3) of a Micro Quadrotor for Stable Trajectory Tracking

Hyeonbeom Lee, Suseong Kim, Tyler Ryan, H.Jin Kim*
Department of Mechanical and Aerospace Engineering
Seoul National University
Seoul
korener33@snu.ac.kr, mer911tb@gmail.com, ryantr@gmail.com, hjinkim@snu.ac.kr

Abstract—This paper presents a backstepping controller based on SE(3) to track the desired trajectory for a quadrotor unmanned aerial vehicle. The controller consists of two parts: 1) a position control part and 2) an attitude control part. The position controller is used to track the desired Cartesian coordinates using position and velocity errors, while the attitude controller uses the rotation matrix error and body angular velocity error to stabilize attitude dynamics expressed on SO(3). Simulation results illustrate the more stable tracking performance of the proposed controller in noisy environments in comparison with geometric controller. Experimental results on a micro quadrotor show the satisfactory performance of the proposed controller.

Index Terms—Quadrotor, UAVs, backstepping, trajectory tracking, SE(3).

I. INTRODUCTION

Quadrotors, which are highly maneuverable and can be made small[1], have become a popular research platform for small UAVs (Unmanned Aerial Vehicles). In order to perform a variety of missions such as surveillance for natural disaster or surveillance for military, trajectory tracking is an essential capability for quadrotor UAVs.

Several solutions to enable trajectory tracking of a quadrotor were proposed [2], [3], [4]. Linear controllers were shown in [2], [4]. A quadrotor followed a desired trajectory with a PID or LQR controller. Tracking performance for a planned sequence of waypoints was shown [2]. A nonlinear controller based on a linearized model was proposed [3] and experimental results were shown. Since these controllers are based on linearized models or linear controllers, problems occur when the nonlinearity is significant.

In order to overcome the nonlinearities and under-actuation of quadrotors, a feedback linearization approach was proposed for quadrotors [5], [6]. Feedback linearization was compared with an adaptive sliding mode controller [5] and showed good results in noisy environments. A linearized H-∞ controller which was robust to uncertainties in parameters and disturbances was presented [6]. However, these controllers involved high-order derivative terms, which is less reliable even when sensor noise is applied.

Backstepping control was applied to the quadrotor [7], [8], [9]. Backstepping and sliding mode controllers were compared [7]. A tracking controller [8], [9] is derived using Lagrangian formulation of quadrotor dynamics based on the backstepping approach. However, it is not easy to implements in real experiments. Also the Euler angle representation has a fundamental restriction due to singularities.

Geometric control law which relies on the geometry of mechanical systems has been previously applied to fully actuated systems or satellites[10]. Geometric controller, which means geometric tracking control for a quadrotor in this paper, was derived based on special Euclidean group SE(3) and proposed [11] to avoid singularities and complexity which can be happened in other control laws. Experimental results of tracking performance under a geometric controller were presented [12], [13]. However, this controller cannot ensure the performance with noisy sensors because the controller is similar to PD control type. A robust geometric tracking controller was presented [14], [15]. [14] presented the attitude controller with an anti-windup integrator technique which is used for robust control. A robust adaptive geometric tracking controller on SO(3) and SE(3) was presented [15] and showed good tracking performance without exact knowledge of inertia matrix. However, there have been no experimental results for trajectory tracking with a robust geometric controller and robust control is not simple for implementation in real experiments.

In this paper, we construct a nonlinear controller based on the backstepping approach for position control. Since geomet-
ric control defined by a rotation error matrix can avoid the singularities that occur when using Euler angles, the attitude controller is designed based on the geometric control approach. The controller which is based on simple integral solution can be easily adapted to situations when noise is applied to the states as well as the inertia. This can make a quadrotor track the desired trajectory more stable in comparison with the states as well as the inertia. This can make a quadrotor track the desired trajectory in Fig. 3 is presented in this section. The quadrotor follows the desired trajectory. Section VI contains concluding remarks.

II. QUADROTOR DYNAMIC MODEL

In this section, the quadrotor dynamic model is described. Each rotor produces lift force \( f_i (i= 1, 2, 3, 4) \) in Fig. 2 and moment. Motors 1 and 3 rotate counter-clockwise while motors 2 and 4 rotate clockwise to cancel the drag moment. The equation of motion with mass \( m \) and inertia tensor \( J \) of the quadrotor can be written as

\[
\begin{align*}
\dot{m} \ddot{X} &= mg e_3 - f R e_3 \\
\dot{R} &= R \hat{\Omega} \\
J \ddot{\Omega} &= -\Omega \times J \Omega + M,
\end{align*}
\]

where \( X \) is the quadrotor position in the inertial frame, \( \Omega \) is the angular velocity in the body frame, \( e_3 = [0; 0; 1]^T \) and \( R \) is the coordinate transformation matrix from the body frame to the inertial frame. \( \hat{\cdot} \) is the hat map which takes an element of \( \mathbb{R}^3 \) to \( SO(3) \).

Each rotor of the quadrotor has an angular velocity \( \omega_i \) and produces a force \( f_i \) and drag moment \( \tau_i \).

\[
f_i = k_f \omega_i^2, \quad \tau_i = k_m \omega_i^2
\]

III. CONTROL DESIGN

The backstepping control on \( SE(3) \) structure as presented in Fig. 3 is presented in this section. The quadrotor follows the desired trajectory \( X_d \) and the desired direction of \( x \)-axis of the body-fixed frame \( [b_1 \ b_2 \ b_3] \), i.e. \( b_1_d = [\cos(\psi_d) \ \sin(\psi_d) \ 0]^T \).

A. Position Controller

Input for the position controller is derived from (1). The way of obtaining desired force is similar to (3). We define the position tracking error \( e_1 \) and sliding mode error \( r_p \) as

\[
e_p := (x_d - x, y_d - y, z_d - z)^T := X_d - X \\
r_p := \dot{e}_p + \Lambda_1 e_p,
\]

where \( \Lambda_1 \) is a 3×3 positive diagonal matrix, which is a design parameter.

To reduce sliding surface error, we now focus on position error dynamics as

\[
\begin{align*}
\dot{r}_p &= m \ddot{e}_p + m \Lambda_1 \dot{e}_p \\
&= m \ddot{X}_d - m \ddot{X} + m \Lambda_1 (r_p - \Lambda_1 e_p) \\
&= m \ddot{X}_d + m \Lambda_1 r_p - m \Lambda_1^2 e_p + F_d - m \omega_3 e_3,
\end{align*}
\]

where \( F_d \) is a desired force vector in \( \mathbb{R}^{3?1} \), which corresponds to \( fRe_3 \).

We get the error dynamics (3) containing the ideal force term from (7) and (1) to keep \( \|r_1\| \) small. To stabilize position
tracking error dynamics, we define the desired force as

\[ F_d = -m \ddot{X}_d + m \Lambda_1^2 e_p + m g e_3 - k_{p1} r_p - k_{i1} \int_0^{t_{	ext{now}}} r_p dt, \]  

(9)

where \( k_{p1} \) and \( k_{i1} \) are 3 \times 3 diagonal control gain matrices.

In general, integral terms for accumulation of position error can make the system unstable in the tracking problem because once a current trajectory is not identical to the desired trajectory, the integral error for positions keeps increasing. However, integral of the sliding surface is more effective because the sliding surface is almost zero when properly controlled.

If the position control gain matrices are positive definite, i.e. \( k_{p1} > 0 \) and \( k_{i1} > 0 \), and \((k_{p1} - m \Lambda_1)^2 > 0\) is satisfied, then the closed-loop position error dynamics becomes

\[ m \ddot{r}_p = -(k_{p1} - m \Lambda_1) r_p - k_{i1} \int_0^{t_{	ext{now}}} r_p dt. \]  

(10)

If the tracking error and sliding surface are assumed to be UUB(Uniformly Ultimately Bounded), the stability proof for error dynamics including the integral of sliding surface can be shown by defining Lyapunov candidate function as \( V = \frac{1}{2} r_1^T P_1 r_1 + \frac{1}{2} [f(r_1^T dt) P_2 [r_1^T dt]. \) We can easily get asymptotic stability by the derivative of Lyapunov candidate function is smaller than zero.

The next step is to obtain the desired direction of the body-fixed \( z \) axis, \( \dot{b}_{3d} \). From the desired force, we get the \( \dot{b}_{3d} \) vector is obtained as \( \dot{b}_{3d} = \frac{F_d}{\| F_d \|} \). \( \dot{b}_{3d} \).

Knowing \( b_{1d} \), the desired rotation matrix is written as \( R_d = [ b_{2d} \times b_{3d} b_{2d} b_{3d} ] \), \( R_d \).

where \( b_{2d} = (b_{3d} \times b_{1d}) / \| b_{3d} \times b_{1d} \| \). In our experiment, the desired rotation matrix is calculated in the attitude control loop as shown in Fig. 6

B. Attitude Controller

For attitude control, we use attitude tracking error \( e_R \) and angular velocity tracking error \( e_\Omega \):

\[ e_R = \frac{1}{2} (R - \bar{R}^T)^\vee, \]  

\[ e_\Omega = \Omega - \bar{R}^T \dot{\Omega}_d, \]  

(13)

where \( \bar{R} = R_d^T R \) and \( ^\vee \) is the vee map which takes an element of \( so(3) \) to \( \mathbb{R}^3 \). The reason for defining attitude error as \( e_R \) and \( e_\Omega \) is shown \( [11] \).

The geometric controller uses the same error definition, but uses the backstepping control approach for improved robustness. We define the sliding mode error \( r_a \) as

\[ r_a = \dot{e}_R + \Lambda_2 e_R. \]  

(15)

The stability in attitude dynamics is proven through attitude error dynamics. However, before obtaining the attitude error dynamics, we need to know \( \dot{e}_R \) and \( \dot{e}_\Omega \).

\[ \dot{e}_R = \frac{1}{2} (\dot{e}_\Omega \bar{R}^T + \bar{R} \dot{e}_\Omega)^\vee = \frac{1}{2} (tr[\bar{R}^T]I - \bar{R}^T)e_\Omega = :D e_\Omega \]  

(16)

\[ \dot{e}_\Omega = \frac{d}{dt}(D e_\Omega) = \dot{D} e_\Omega + D \dot{e}_\Omega \]  

(17)

\[ \dot{e}_\Omega = \dot{\Omega} + (\dot{\bar{R}}^T \dot{\Omega}_d - \bar{R}^T \dot{\dot{\Omega}}_d). \]  

(18)

In \( [16] \), the property \( \hat{x} A + A^T \hat{x} = (tr(A) I - A)x \) has been used. The matrix \( D \) defined in \( [16] \) is invertible when the rotation angle between \( R \) and \( R_d \) is less than \( 180^\circ \). To obtain \( e_\Omega \) in \( [18] \) from \( [14] \), the property \( \dot{\Omega}_d \Omega_d^T = \Omega_d^T \Omega_d = 0 \) has been used. We obtain attitude error dynamics from \( [16] \), \( [17] \), \( [18] \) and \( [15] \) as

\[ J \dot{R}_2 = J \dot{e}_R + J \Lambda_2 \dot{e}_\Omega \]  

(19)

\[ = J \dot{D} e_\Omega + J \dot{D} e_\Omega + J \Lambda_2 (r_2 - \Lambda_2 r_2) \]  

\[ = J \dot{D} e_\Omega + J D (\hat{\bar{R}}^T \dot{\Omega}_d - \bar{R}^T \dot{\dot{\Omega}}_d) + J \Lambda_2 r_2 - J \Lambda_2^2 e_R \]  

(20)

\[ J \dot{D} e_\Omega + J D (\hat{\bar{R}}^T \dot{\Omega}_d - \bar{R}^T \dot{\dot{\Omega}}_d) + J \Lambda_2 r_2 - J \Lambda_2^2 e_R, \]  

(21)

where \( \Lambda_2 \) is a \( 3 \times 3 \) design parameter matrix, which is diagonal with positive entries. From \( [16] \), \( [18] \) and \( [19] \), we get nonlinear error dynamics as shown in \( [20] \). \( [21] \) is obtained using attitude dynamics \( [2] \).

To cancel the nonlinear attitude term and reduce sliding surface error in the same manner as the position controller, we define desired momentum as

\[ M = \Omega \times J \Omega - J D^{-1} J^{-1}(k_{p2} r_a + k_{i2} \int_0^{t_{\text{now}}} r_a dt) \]  

\[ - J (\bar{\Omega} \bar{R}^T \dot{\Omega}_d - \bar{R}^T \dot{\dot{\Omega}}_d) - J D^{-1} \dot{D} e_\Omega + J D^{-1} \Lambda_2^2 e_R, \]  

(22)

where \( k_{p2} \) and \( k_{i2} \) are \( 3 \times 3 \) diagonal control gain matrices.

The desired momentum shown in \( [22] \) is complex for implementation in real experiments. To simplify \( [22] \), we assumed that the angular error between \( R \) and \( R_d \) is very small, so \( D \) can be approximated as identity matrix. \( [22] \) can be rewritten as

\[ M_{\text{modified}} = \Omega \times J \Omega - (k_{p2} r_a + k_{i2} \int_0^{t_{\text{now}}} r_a dt) \]  

\[ - J (\bar{\Omega} \bar{R}^T \dot{\Omega}_d - \bar{R}^T \dot{\dot{\Omega}}_d) - J \Lambda_2^2 e_R, \]  

(23)

If desired angular velocity is almost zero, then \( M \) is simply obtained as PID control law and expressed as

\[ M_{\text{simple}} = - k_{p2} r_a - k_{i2} \int_0^{t_{\text{now}}} r_a dt + J \Lambda_2^2 e_R. \]  

(24)

The controller achieves good tracking performance shown in Sec. \( [\dot{V}] \) and Sec. \( [\dot{W}] \) even for the micro quadrotor, despite the simplicity of this controller. If attitude control gain matrices are positive definite, i.e. \( k_{p2} > 0, k_{i2} > 0 \) and \( (k_{p2} - J \Lambda_2) > 0 \),
is satisfied, then the closed-loop attitude error dynamics appear as

\[ J \dot{\mathbf{r}}_a = -(k_2 - J \Lambda_2) \mathbf{r}_a - k_i \int_0^{\tau_{now}} r_a dt. \] (25)

Stability can be proved in same manner as position control. Experimental results showed that the controller achieved good tracking performance, despite the simplicity of this controller.

IV. SIMULATION RESULTS

Prior toe experiments, simulation results in a noisy environment are performed to illustrate the difference between the proposed controller and the simple geometric controller which is shown in [11], [12].

A. Quadrotor Model

The quadrotor parameters used are given in Table I. The properties here are the same as that used in the experiment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mass</td>
<td>0.3 kg</td>
<td>Ixx</td>
<td>0.00079 kgm^2</td>
</tr>
<tr>
<td>b</td>
<td>9.791e-9 N/rpm^2</td>
<td>Iyy</td>
<td>0.00079 kgm^2</td>
</tr>
<tr>
<td>d</td>
<td>2.399e-10 N/rpm^2</td>
<td>Izz</td>
<td>0.00148 kgm^2</td>
</tr>
<tr>
<td>Arm length</td>
<td>0.114m</td>
<td>Max. rpm</td>
<td>13000 rpm</td>
</tr>
</tbody>
</table>

The rotational inertia are obtained through SolidWorks program and the thrust coefficient b and drag coefficient d are estimated from flight data. In this simulation, we also considered simple motor dynamics as

\[ \dot{\mathbf{f}} = p(f_d - \mathbf{f}), \] (26)

where \( f_d \) is the desired thrust force. The rate p is assumed to be 80 s^{-1} for increasing thrust, and 60 s^{-1} for decreasing (see [18]).

The simulation is performed under Matlab Simulink. Quadrotor dynamics block as shown in [5] is constructed by a Matlab S-function based on (1) and (2) (See [19]).

B. Simulation Results

The red line is the desired trajectory which is shown as Fig. 4a shows the trajectory following performance in 3D frame. Fig. 4b shows the x,y,z displacement and velocity. The red line is the desired trajectory which is shown as

\[ X_d = \begin{bmatrix} 0.7 \cos(0.3 \pi t) & 0.7 \sin(0.3 \pi t) & 0.55 \end{bmatrix}^T. \] (27)

The desired yaw is set to be 0°, i.e. \( b_{1,d}=[1 0 0]^T \).

The blue solid line is the trajectory using the proposed controller, and the green dotted line is the trajectory using the simple geometric controller. We assume that white Gaussian noise is applied to the position and Euler angles with standard deviation as 0.01 m and 0.01 rad. This means that maximum 6cm error in position and 4 degree in attitude error is applied. The translational and angular velocities are also affected by position and Euler angle noise.

Fig. 4: Simulation results about comparing trajectory controller (Blue: the proposed controller, green: conventional geometric controller, red: the desired value).

Fig. 5: Attitude error. (Left column: actual and desired Euler angle under the proposed controller, right column: under the conventional geometric controller)

Fig. 5a shows the Euler angle error in the inertial frame and Fig. 5b shows the history of the attitude errors, \( e_R \) and \( e_\Omega \). \( e_R(n) \) is the n-th element of the vector \( e_R \), and \( e_\Omega(n) \) is the n-th element of the vector \( e_\Omega \) in Fig 5b. We show attitude error from 16 s to 20 s. Cyan and magenta dotted lines in Fig. 5a are the desired Euler angles obtained from each controller. In attitude control, there is no direct information available about the Euler angles because the controller uses the rotation error matrix and body angular rate error. However, the desired Euler angles are obtained from the desired rotation matrix in [12] using the inverse mapping equation as shown in (28).

\[
\begin{bmatrix}
\phi \\
\theta \\
\psi
\end{bmatrix}
= \begin{bmatrix}
\arctan 2(R_{32}, R_{33}) \\
- \arcsin(R_{13}) \\
\arctan 2(R_{21}, R_{11})
\end{bmatrix}
\] (28)

In the above equation, the values \( R_{**} \) are given by the
The proposed controller

\[ R = \begin{bmatrix} c\psi c\theta & c\psi s\theta s\phi - s\psi c\phi & c\psi s\theta c\phi + s\psi s\phi \\ s\psi c\theta & s\psi s\theta s\phi + c\psi c\phi & s\psi s\theta c\phi - c\psi s\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix}, \]  

\( (29) \)

where \( s\phi = \sin(\phi), \ c\phi = \cos(\phi), \ s\theta = \sin(\theta), \ c\theta = \cos(\theta), \ s\psi = \sin(\psi) \) and \( c\psi = \cos(\psi) \).

The main reason for the performance difference between the two controllers is that because of lack of robustness of the position controller to noise, the desired Euler angles are incorrectly calculated. Therefore, the magenta and black lines are different. Attitude control performance under two controllers is also different as shown in Table II. 

**Attitude control using the geometric controller did not show satisfactory Euler angle tracking performance, whereas our controller showed good tracking performance.**

The main reason for the performance difference between the two controllers is that because of lack of robustness of the position controller to noise, the desired Euler angles are incorrectly calculated. Therefore, the magenta and black lines are different. Attitude control performance under two controllers is also different as shown in Table II. 

**TABLE II: Norm of trajectory tracking error**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>The proposed controller</th>
<th>Geometric controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>error of x</td>
<td>1.5997</td>
<td>6.0645</td>
</tr>
<tr>
<td>error of y</td>
<td>1.8525</td>
<td>8.1023</td>
</tr>
<tr>
<td>error of z</td>
<td>1.7474</td>
<td>5.8466</td>
</tr>
<tr>
<td>error of ( \phi )</td>
<td>71.7355</td>
<td>103.5056</td>
</tr>
<tr>
<td>error of ( \theta )</td>
<td>62.1029</td>
<td>57.6395</td>
</tr>
<tr>
<td>error of ( \psi )</td>
<td>22.7765</td>
<td>26.6749</td>
</tr>
</tbody>
</table>

Simulation results show that our solution gives more accurate results than a simple geometric controller against sensor noise.

**V. EXPERIMENTAL RESULTS**

In this section, the trajectory tracking experiment is described to show the tracking performance using the Vicon motion capture system.

**A. Quadrotor Hardware**

The quadrotor used was developed by [16] including author. Four 2200 kv BLDC motors with \( 5 \times 3 \) propellers are used for thrust of the quadrotor. Total weight is 300 grams with battery. Flight time is about 9 minutes with a 2S 1000 mAh li-po battery. The main processor is an 8-bit microprocessor based on arduino. This quadrotor includes a semiconductor optical flow sensor [16], but we not use the autonomous hovering function in our experiments.

**B. Experiment Setup**

We performed the trajectory flight test as shown in Fig. 6. Position and attitude control calculations are performed at the base station at 45 Hz control frequency. From the desired trajectory and measured state variables, the desired force \( f \) is sent to control command calculation part. Attitude control is performed from the desired yaw vector \( b_{1d} \) and desired thrust vector \( b_{3d} \). Desired moment \( M \) is also sent to the control command loop. The control command is obtained from the desired thrust \( f \), desired momentum \( M \) and motor scaling matrix in \( \mathbf{R} \). After calculating the control command, control signal is sent to the quadrotor. The base station computer sends this control signal to the quadrotor, then the onboard micro processor sends PWM signals to each motor.

The flight scenario is as follows: the quadrotor takes off to a predefined height, 0.55 m, within 20 s. Then the quadrotor is commanded to hover, and to follow the desired trajectory for the next 80 s. If the quadrotor finishes following the trajectory, it lands at that position.

**C. Experiment Results**

Fig. 7a shows the 3D trajectory of the quadrotor for 100 s. In the first phase, the quadrotor is made to fly to a point \( X_d = [0.6 \ 0 \ 0.55] \). After arriving at the point, the quadrotor is made to follow the trajectory given in (27). The desired trajectory is a small circle with radius 0.7 m. This desired trajectory, represented by the red line in Fig. 7, is the same as in the simulation in (27). The blue line under the proposed controller is the actual trajectory of the quadrotor. Fig. 7a shows x, y and z displacements and velocity of the quadrotor.

Fig. 8a shows roll, pitch and yaw angle error in the experi-
Our future work is to perform tacking experiments in high angle condition and outdoor environments.

VI. CONCLUSIONS

This paper presented a geometric-based backstepping control algorithm for trajectory tracking for a quadrotor. The proposed controller consists of two parts: position control and attitude control. The position controller is used to track the desired trajectory in the Cartesian coordinate using position and velocity error, whereas the attitude controller using rotation matrix error and body angular velocity error tracks the desired angle obtained from the position controller and desired yaw angle. Simulation results showed that our solution gives a good trajectory tracking performance against sensor noise, whereas other geometric controller proposed in the previous works does not. The experimental results show satisfactory tracking performance in following a small circular trajectory. Our future work is to perform tacking experiments in high angle condition and outdoor environments.

ACKNOWLEDGEMENT

This work was supported by a Defense Research Grant, funded by the Agency for Defense Development, under the contract UD120013JD. This work was also supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MEST)(2013013911).

REFERENCES