J Intell Robot Syst (2013) 70:1–12
DOI 10.1007/s10846-012-9789-z

Control and Navigation Framework for Quadrotor Helicopters

Amr Nagaty · Sajad Saeedi · Carl Thibault · Mae Seto · Howard Li

Received: 20 September 2012 / Accepted: 20 September 2012 / Published online: 14 October 2012

Abstract This paper presents the development of a nonlinear quadrotor simulation framework together with a nonlinear controller. The quadrotor stabilization and navigation problems are tackled using a nested loops control architecture. A nonlinear Backstepping controller is implemented for the inner stabilization loop. It asymptotically tracks reference attitude, altitude and heading trajectories. The outer loop controller generates the reference trajectories for the inner loop controller to reach the desired waypoint. To ensure boundedness of the reference trajectories, a PD controller with a saturation function is used for the outer loop. Due to the complexity involved in controller development and testing, a simulation framework has been developed. It is based on the Gazebo 3D robotics simulator and the Open Dynamics Engine (ODE) library. The framework can effectively facilitate the development and validation of controllers. It has been released and is available at Gazebo quadrotor simulator (2012).

Keywords Quadrotor · Nonlinear control · Backstepping · Navigation · Simulation framework · Gazebo · Open dynamics engine

A. Nagaty (✉) · S. Saeedi · C. Thibault · H. Li
COllaboration Based Robotics and Automation (COBRA) Laboratory, Department of Electrical and Computer Engineering, University of New Brunswick, Fredericton, New Brunswick, Canada
e-mail: amr.nagaty@unb.ca
URL:http://www.unb.ca/cobra

S. Saeedi
e-mail: sajad.saeedi.g@unb.ca

C. Thibault
e-mail: carl.t@unb.ca

H. Li
e-mail: howard@unb.ca

M. Seto
Department of Mechanical Engineering, University of New Brunswick, Fredericton, New Brunswick, Canada
e-mail: mz715250@dal.ca

1 Introduction

Unmanned aerial vehicles (UAVs) have become increasingly popular for military and commercial applications. The market demand for UAVs arises from low manufacturing and operational costs as compared to their manned counterparts. UAVs are mostly used for surveillance, inspection and data acquisition. Their potential applications include border patrol, search and rescue, wildfire monitoring, traffic monitoring and land surveys. Most of the previously mentioned applications require hovering and vertical takeoff and landing (VTOL) capabilities. Generally, fixed-wing aircrafts are unable to perform VTOL and suffer from maneuverability constraints. Conventional helicopters are capable of hovering and VTOL but are dynamically and structurally complex, expensive and hard to control [2]. Quadrotor helicopters are becoming more favorable than conventional helicopters as they are mechanically simpler and easier to control. Still, quadrotor control is a challenging problem because of the inherent system nonlinearities and cross couplings due to the gyroscopic moments and underactuation [3].

Different control methods have been recently applied to tackle the quadrotor's stability problem. PID and LQ control methods have been reported to successfully stabilize the quadrotor's attitude around hover position in the presence of minor disturbances [3]. Later, the same authors applied backstepping and sliding mode nonlinear control methods and reported improved stability in the presence of relatively high perturbations [4]. In [5], feedback linearization controller was combined with linear H∞ controller to robustify the control law. Integral sliding mode and reinforcement learning methods were applied in [6] for altitude control. The attractive cascaded-systems structure of the quadrotor's model suggests the application of the Backstepping approach [7]. It has been used several times in literature for quadrotor stabilization. In [8], the quadrotor system was divided into three interconnected subsystems: under-actuated, fully actuated and propeller subsystems. Backstepping algorithm was applied recursively until the whole system was stabilized. In [9], a hybrid Backstepping technique and the Fernet-Serret Theory were applied to improve the disturbance rejection capability. An integral term in the tracking error was incorporated to eliminate steady state error.

The complexity correlated to the design of quadrotor controllers sets up the necessity for a reliable simulation framework [10]. A well designed and tuned simulation can effectively reduce the developing and testing time and cost for controllers. The major requirements in a simulation framework are:

- Accurate mathematical model of quadrotors
- Ease of control system development and testing
- Good quality rendering
- Simulated set of sensors

Due to the overhead involved in quadrotor simulator development, most researchers tend to only rely on numerical simulations without visualization. On the other hand, commercial solutions are generally expensive such as the RotorLib helicopter simulator developed by RTDynamics [11]. Some open source solutions are available, however they lack some of the previous requirements. JSBSim is an open source nonlinear flight dynamics simulator that lacks rendering [12]. FlightGear is an open flight simulator framework, based on JSBSim, with a sophisticated visualizer [13]. However, it lacks simulated sensors feedback. For the UAV's branch of robotics, open source simulator development is lagging. While in the case of ground robots, more complete frameworks have been developed and thoroughly tested. In particular, we focus on the Player / Stage / Gazebo framework [14]. It is the most complete simulation framework with respect to the previously mentioned requirements. It is capable of simulating robots, sensors and objects in a three-dimensional world. In addition, it generates realistic sensor feedback that can be used for testing not only controllers but autonomous behaviors as well. It is based on accurate simulation of rigid body dynamics using the Open Dynamics Engine (ODE) library [15].

In this paper, we tackle the stabilization and navigation problems of a quadrotor. A nonlinear model is derived and used for controller design. Motivated by the structure of the model, a nested loops control architecture is used. For the inner loop, a Backstepping tracking controller is designed for attitude, altitude and heading. For the outer loop, a PD controller with a saturation function is implemented. It achieves waypoint navigation while ensuring the boundedness of the reference trajectories. We build upon an existing simulator to develop a simulation framework that meets the previous requirements [16]. It is based on the Gazebo 3D robotics simulator, the ODE library and the nonlinear quadrotor model.

This paper is organized as follows: The quadrotor kinematics and dynamics models are derived in Section 2. In Section 3, we present the state space model and the navigation and stabilization controllers. The stability of the proposed controller is investigated in Section 4. The developed simulation framework is introduced in Section 5. Simulation results are presented in Section 6 to show the effectiveness of the proposed controller. Finally, the paper is concluded in Section 7.

2 Quadrotor Model

A quadrotor aircraft is actuated by four rotors. A cross formed by two arms holds the four rotors as shown in Fig. 1. Two rotors at the ends of one arm rotate in the clockwise direction, while the other pair of rotors rotates in the opposite direction to cancel the yawing moment. Control forces and moments are generated by varying the speed of the rotors (Ω_1, Ω_2, Ω_3, Ω_4). Table 1 shows how to generate positive control forces and moments by varying rotor speeds. In Table 1, a (+) symbol indicates that increasing the corresponding rotor speed generates a positive force / moment. While, a (−) symbol indicates that decreasing the corresponding rotor speed generates a positive force / moment.

In this section, the quadrotor model used for the controller design is described. The kinematics and dynamics models are derived separately in order to design the controller.

2.1 Kinematics Model

To transform a vector from the body frame (x, y, and z) to the inertial frame (N, E, and D) shown in Fig. 1, the following transformation matrix is used [17]

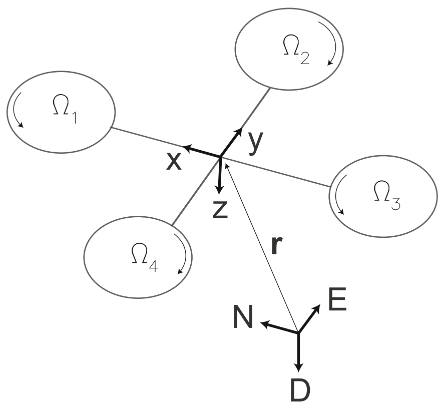

Fig. 1 Quadrotor reference frames

Table 1 Variation of rotor speeds to generate control forces and moments

Force / moment	Ω_1	Ω_2	Ω_3	Ω_4
Roll moment		−		+
Pitch moment	+		−	
Yaw moment	+	−	+	−
Vertical thrust	+	+	+	+

$$\mathbf{R} = \begin{bmatrix} c\psi\, c\theta & c\psi\, s\phi\, s\theta - c\phi\, s\psi & s\phi\, s\psi + c\phi\, c\psi\, s\theta \\ c\theta\, s\psi & c\phi\, c\psi + s\phi\, s\psi\, s\theta & c\phi\, s\psi\, s\theta - c\psi\, s\phi \\ -s\theta & c\theta\, s\phi & c\phi\, c\theta \end{bmatrix} \tag{1}$$

The order of rotation used is yaw (ψ) followed by pitch (θ) followed by roll (ϕ) around the z, y and x axes respectively. Euler rates $\dot{\eta} = \left[\, \dot{\phi}\ \dot{\theta}\ \dot{\psi} \,\right]^T$ and angular body rates $\omega = [\, p\ q\ r \,]^T$ are related by [18]

$$\omega = \mathbf{R}_r \dot{\eta} \tag{2}$$

where

$$\mathbf{R}_r = \begin{bmatrix} 1 & 0 & -\sin(\theta) \\ 0 & \cos(\phi) & \sin(\phi)\cos(\theta) \\ 0 & -\sin(\phi) & \cos(\phi)\cos(\theta) \end{bmatrix} \tag{3}$$

Around hover position, we assume the following condition [17–19]

$$\mathbf{R}_r \approx \mathbf{I}_{3\times 3} \tag{4}$$

where **I** is the identity matrix.

2.2 Dynamics Model

The dynamics model consists of the rotational and translational motions. The rotational motion is fully actuated, while the translational motion is underactuated [8, 9]. The rotational equations of motion are derived in the body frame using the Newton-Euler method [19]

$$\mathbf{J}\dot{\omega} + \omega \times \mathbf{J}\omega + \omega \times \left[\, 0\ 0\ J_r \Omega_r \,\right]^T = \mathbf{M}_B \tag{5}$$

where **J** is the quadrotor's diagonal inertia matrix. The last term on the left hand side of Eq. 5 represents the gyroscopic moment due to the rotors' inertia J_r and relative speed $\Omega_r = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4$. The aerodynamic forces and moments produced by the ith rotor are directly proportional to the square of the rotor's speed

$$\begin{aligned} F_i &= k_F \Omega_i^2 \\ M_i &= k_M \Omega_i^2 \end{aligned} \tag{6}$$

where k_F and k_M are the aerodynamic force and moment constants respectively. The moments acting on the quadrotor in the body frame are given by

$$\mathbf{M}_B = \begin{bmatrix} l \cdot k_F \left(-\Omega_2^2 + \Omega_4^2\right) \\ l \cdot k_F \left(\Omega_1^2 - \Omega_3^2\right) \\ k_M \left(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2\right) \end{bmatrix} \tag{7}$$

where l is the moment arm, the distance from the axis of rotation of the rotors to the center of the quadrotor. The translational equations of motion are derived in a North-East-Down navigation frame using Newton's second law

$$m\ddot{\mathbf{r}} = \left[0 \; 0 \; mg \right]^T + \mathbf{R}\mathbf{F}_B \tag{8}$$

where $\mathbf{r} = \left[x \; y \; z \right]^T$ is the quadrotor's position in the navigation frame, m is the quadrotor's mass and g is the acceleration due to gravity. The non-gravitational forces acting on the quadrotor in the body frame are given by

$$\mathbf{F}_B = \begin{bmatrix} 0 \\ 0 \\ -k_F \left(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2\right) \end{bmatrix} \tag{9}$$

3 Controller Design

Because of the quadrotor dynamics, a nested loops control strategy is appropriate [8]. From Eqs. 5 and 8, it can be seen that the rotational motion is independent of the translational motion, while the opposite is not true. Thus, an inner control loop can be designed to ensure asymptotic tracking of desired attitude, altitude and heading. While, an outer control loop can be designed for quadrotor navigation, as shown in Fig. 2. In this section, the quadrotor's state space model is presented for controller design. A PD controller is designed for the outer loop, while a Backstepping controller is designed for the inner loop.

3.1 State Space Model

The state vector is defined as [4]

$$\mathbf{X} = \left[\phi \; \dot{\phi} \; \theta \; \dot{\theta} \; \psi \; \dot{\psi} \; z \; \dot{z} \; x \; \dot{x} \; y \; \dot{y} \right] \tag{10}$$

The control input vector is defined as

$$\mathbf{U} = \left[U_1 \; U_2 \; U_3 \; U_4 \right] \tag{11}$$

where

$$\begin{aligned} U_1 &= k_F \left(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2\right) \\ U_2 &= k_F \left(-\Omega_2^2 + \Omega_4^2\right) \\ U_3 &= k_F \left(\Omega_1^2 - \Omega_3^2\right) \\ U_4 &= k_M \left(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2\right) \end{aligned} \tag{12}$$

The state space model is given by

$$\dot{\mathbf{X}} = \mathbf{f}\left(\mathbf{X}, \mathbf{U}\right) \tag{13}$$

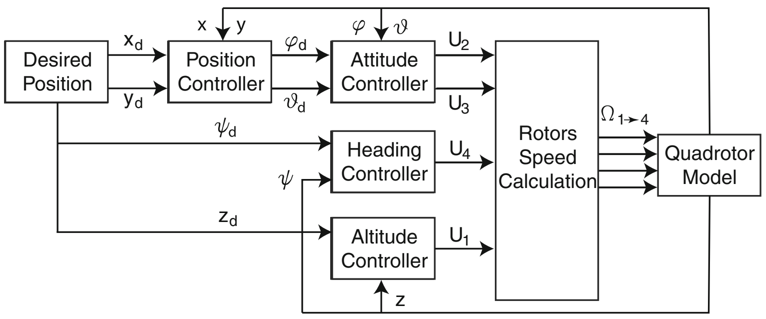

Fig. 2 Control architecture

where

$$\mathbf{f}(\mathbf{X},\mathbf{U})=\begin{bmatrix} x_2 \\ x_4x_6a_1+x_4\Omega_r a_2+b_1U_2 \\ x_4 \\ x_2x_6a_3+x_2\Omega_r a_4+b_2U_3 \\ x_6 \\ x_2x_4a_5+b_3U_4 \\ x_8 \\ g-\frac{U_1}{m}\cos x_1\cos x_3 \\ x_{10} \\ -\frac{U_1}{m}(\sin x_1\sin x_5+\cos x_1\sin x_3\cos x_5) \\ x_{12} \\ \frac{U_1}{m}(\sin x_1\cos x_5-\cos x_1\sin x_3\sin x_5) \end{bmatrix} \tag{14}$$

$$\begin{aligned} a_1&=\frac{(I_{yy}-I_{zz})}{I_{xx}} \\ a_2&=\frac{J_r}{I_{xx}} \\ a_3&=\frac{(I_{zz}-I_{xx})}{I_{yy}} \\ a_4&=\frac{J_r}{I_{yy}} \\ a_5&=\frac{(I_{xx}-I_{yy})}{I_{zz}} \\ b_1&=\frac{l}{I_{xx}} \\ b_2&=\frac{l}{I_{yy}} \\ b_3&=\frac{1}{I_{zz}} \end{aligned} \tag{15}$$

3.2 Outer Control Loop

For position control, a PD controller with a saturation function is implemented to generate the reference roll ϕ_d and pitch θ_d. The reference angles are tracked by the inner loop controller [19]. Based on the desired waypoint, the position controller calculates the desired accelerations $\ddot{x}_d$ and $\ddot{y}_d$

$$\begin{aligned} \ddot{x}_d&=k_p\,(x_{\text{ref}}-x)+k_d\,(\dot{x}_{\text{ref}}-\dot{x}) \\ \ddot{y}_d&=k_p\,(y_{\text{ref}}-y)+k_d\,(\dot{y}_{\text{ref}}-\dot{y}) \end{aligned} \tag{16}$$

where (x_d, y_d) is the desired waypoint, $(\dot{x}_d, \dot{y}_d)$ is the desired velocity and k_p and k_d are the proportional and derivative controller gains respectively. The reference roll and pitch angles can be solved for using the desired accelerations

$$\begin{aligned} -\frac{U_1}{m}\,[\sin\phi_d\sin\psi+\cos\phi_d\sin\theta_d\cos\psi]&=\ddot{x}_d \\ \frac{U_1}{m}\,[\sin\phi_d\cos\psi-\cos\phi_d\sin\theta_d\sin\psi]&=\ddot{y}_d \end{aligned} \tag{17}$$

Using the small angle assumption around the hover position, the previous equations can be simplified

$$\begin{aligned} -\frac{U_1}{m}\,[\phi_d\sin\psi+\theta_d\cos\psi]&=\ddot{x}_d \\ \frac{U_1}{m}\,[\phi_d\cos\psi-\theta_d\sin\psi]&=\ddot{y}_d \end{aligned} \tag{18}$$

$$\begin{bmatrix} -\sin\psi & -\cos\psi \\ \cos\psi & -\sin\psi \end{bmatrix}\begin{bmatrix} \phi_d \\ \theta_d \end{bmatrix}=\frac{m}{U_1}\begin{bmatrix} \ddot{x}_d \\ \ddot{y}_d \end{bmatrix} \tag{19}$$

which has a closed form solution for ϕ_d and θ_d. A saturation function is needed to ensure that the reference roll and pitch angles are within specified limits

$$\begin{aligned} \phi_d&=sat(\phi_d) \\ \theta_d&=sat(\theta_d) \end{aligned} \tag{20}$$

where

$$sat(v)=\begin{cases} v & \|v\|\le v_{\max} \\ sign(v)v_{\max} & \|v\|>v_{\max} \end{cases} \tag{21}$$

such that

$$sign(v)=\begin{cases} -1 & v<0 \\ 1 & v\ge 0 \end{cases} \tag{22}$$

3.3 Inner Control Loop

The attitude, heading and altitude controllers (see Fig. 2) are derived using the Backstepping approach. The idea behind Backstepping is to design a controller recursively by considering some of the state variables as virtual controls. Then, intermediate stabilizing control laws are designed for the virtual controls [20]. At the last step, the actual control input is used to stabilize the whole system. The design procedure is systematic. Therefore, we only present the design procedure for roll control. Pitch, yaw and altitude controllers can be derived similarly. For clarity of presentation, the nonlinear system under investigation is extracted from the state space model derived in Eq. 14.

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = x_4 x_6 a_1 + x_4 \Omega_r a_2 + b_1 U_2 \tag{23}$$

which is in the strict feedback form

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = f(\mathbf{X}) + b_1 U_2 \tag{24}$$

The backstepping procedure transforms the closed loop system to the following coordinates

$$z_1 = x_1 - x_{1d}$$
$$z_2 = x_2 - \alpha_1 \tag{25}$$

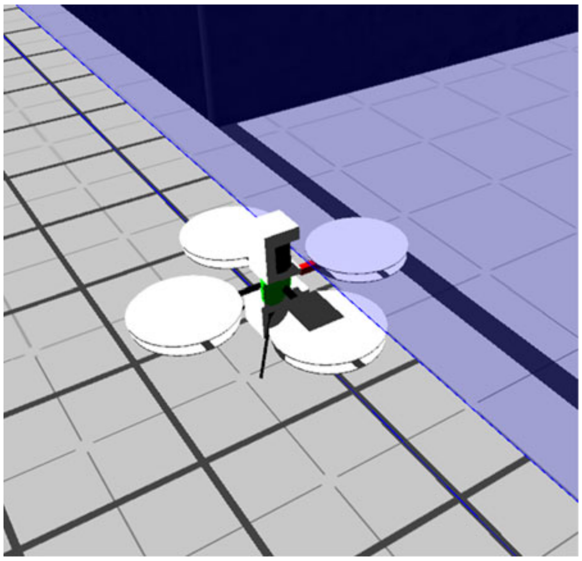
Fig. 4 Simulation environment

where α_1 is the virtual control input and $x_1 - x_{1d}$ is the roll tracking error. We introduce the following Lyapunov function

$$V_1 = \frac{1}{2} z_1^2 \tag{26}$$

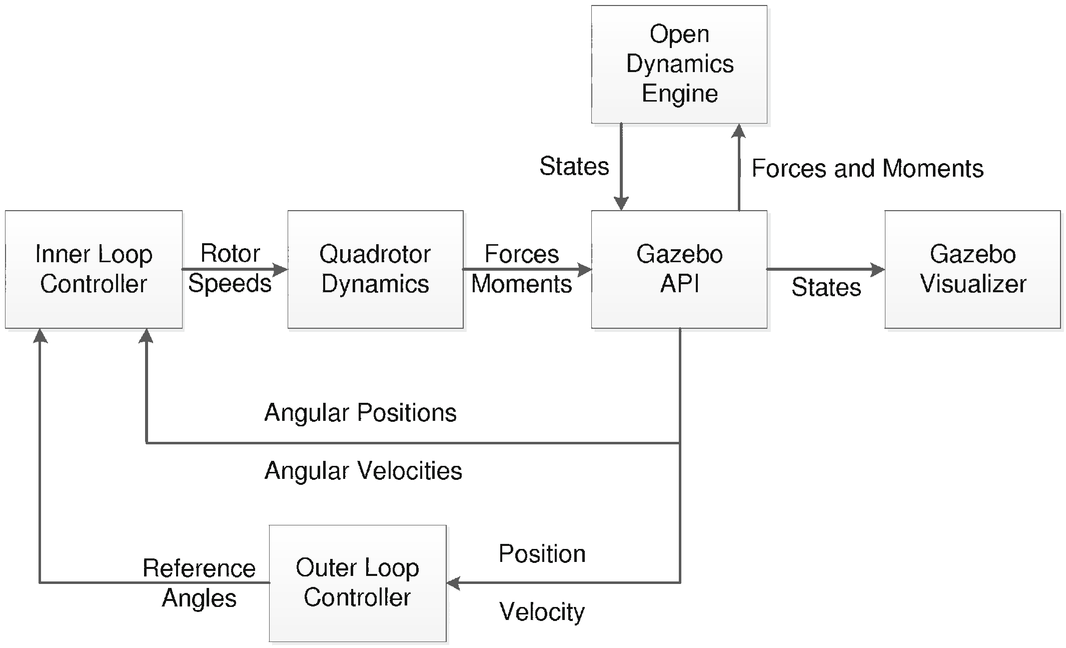

Fig. 3 Simulator framework

whose derivative along the system trajectories is given by

$$\begin{aligned}\dot{V}_1 &= z_1\dot{z}_1 \\ &= z_1\left(z_2 + \alpha_1 - \dot{x}_{1d}\right) \qquad (27)\end{aligned}$$

The virtual control input can be chosen as

$$\alpha_1 = -c_1 z_1 + \dot{x}_{1d} \qquad (28)$$

such that

$$\dot{V}_1 = -c_1 z_1^2 + z_1 z_2 \qquad (29)$$

We augment the previous Lyapunov function in the following way

$$V_2 = \frac{1}{2}z_1^2 + \frac{1}{2}z_2^2 \qquad (30)$$

The derivative along system trajectories is given by

$$\begin{aligned}\dot{V}_2 &= z_1\dot{z}_1 + z_2\dot{z}_2 \\ &= -c_1 z_1^2 + z_1 z_2 + z_2\left(\dot{x}_2 - \dot{\alpha}_1\right) \\ &= -c_1 z_1^2 + z_2\left(z_1 + f(\mathbf{X}) + b_1 U_2 - \dot{\alpha}_1\right) \qquad (31)\end{aligned}$$

Now, the actual control input is at our disposal for the overall roll control

$$U_2 = \frac{1}{b_1}\left(-c_2 z_2 - z_1 - f(\mathbf{X}) + \dot{\alpha}_1\right) \qquad (32)$$

such that

$$\dot{V}_2 = -c_1 z_1^2 - c_2 z_2^2 \qquad (33)$$

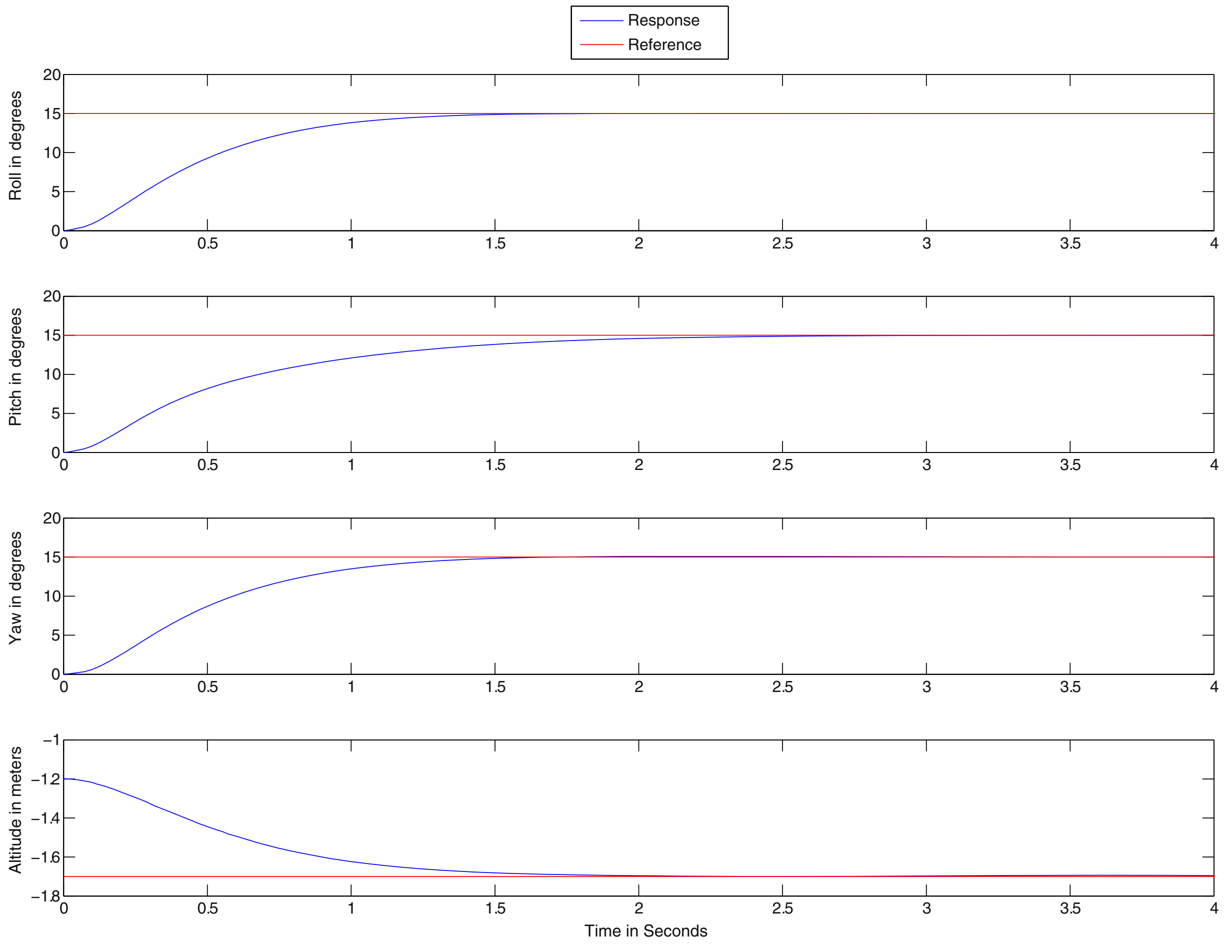

Fig. 5 Simulation results for closed inner loop response

4 Stability Analysis

In this section, the stability and performance of the proposed inner loop controller are analyzed using Lyapunov's stability theorem [7, 20]. The closed loop system is analyzed in the transformed coordinate system

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = \begin{bmatrix} -c_1 & 1 \\ -1 & -c_2 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \tag{34}$$

The closed loop system is an autonomous system in the error coordinates. Stability can be analyzed using LaSalle invariance theorem. Recall Eq. 30, it is clear that the Lyapunov function is positive definite, decresent and radially unbounded. Recall Eq. 33, the Lie derivative of the Lyapunov function is negative definite. According to Lyapunov's second method, the closed loop system is Globally Asymptotically Stable (GAS). Also, the system trajectories z_1 and z_2 are bounded. From Eq. 28, the boundedness of the virtual input α_1 can be deduced. From Eq. 32, the boundedness of the actual control input can be deduced. Moreover, the Lyapunov function has a minimum dissipation rate

$$\dot{V}_2(\mathbf{Z}) \leq -\sigma V_2(\mathbf{Z}) \tag{35}$$

where $c_1 \geq \frac{1}{2}$, $c_2 \geq \frac{1}{2}$ and $\mathbf{Z} = \begin{bmatrix} z_1 & z_2 \end{bmatrix}$. Therefore, the closed loop system is Globally Exponentially Stable (GES) with bounded control input as long as Eq. 4 holds. Next, a bound on the transient tracking error is derived in terms of the controller design parameters. From Eq. 33, the dissipation rate of the Lyapunov function satisfies

$$\dot{V}_2(\mathbf{Z}) \leq -c_1 z_1^2 \tag{36}$$

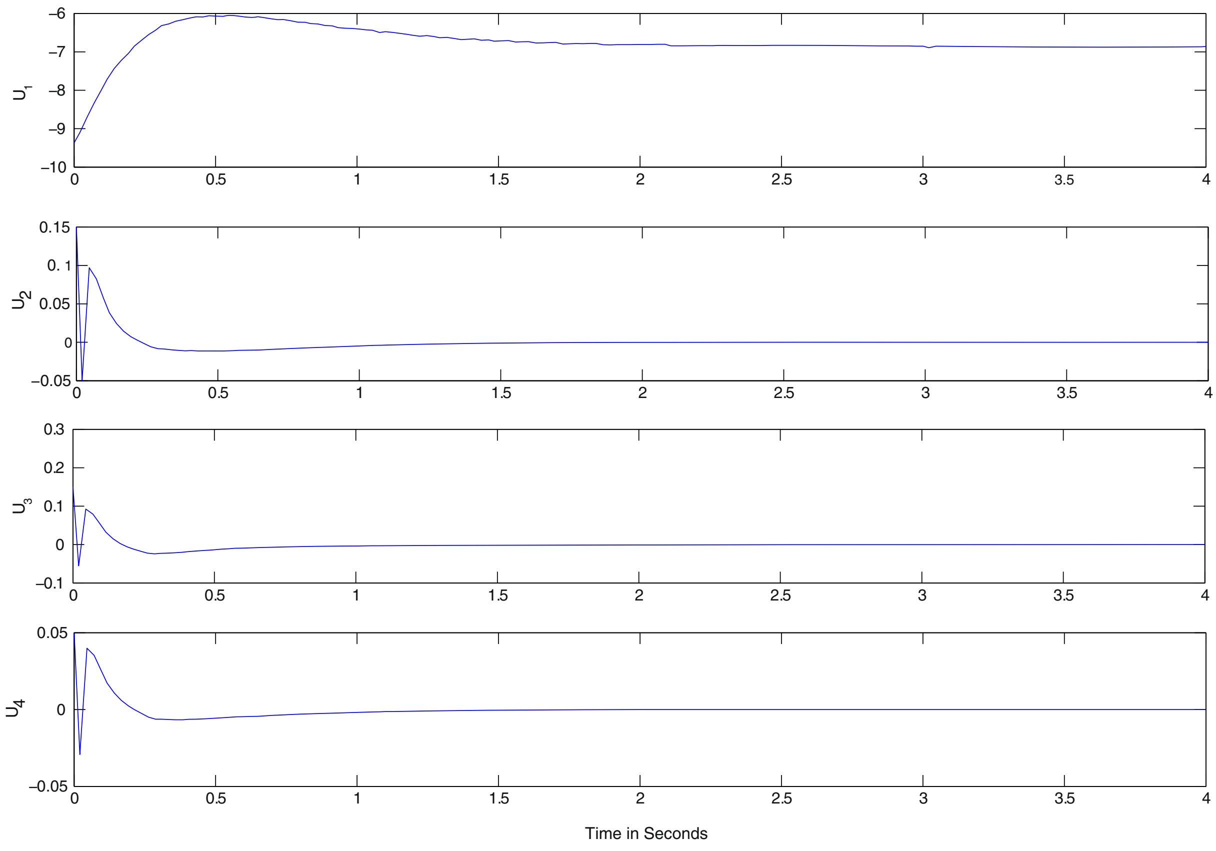

Fig. 6 Simulation results for actuation inputs

The L_2 norm of the transient tracking error is given by

$$\begin{aligned}||z_1||_2^2 &= \int_0^\infty |z_1(\tau)|^2\, d\tau \\ &\leq -\frac{1}{c_1}\int_0^\infty \dot{V}_2((\mathbf{Z}(\tau))\, d\tau \\ &\leq \frac{1}{c_1}[V_2(\mathbf{Z}(0)) - V_2(\mathbf{Z}(\infty))] \\ &\leq \frac{1}{c_1}V_2(\mathbf{Z}(0))\end{aligned} \tag{37}$$

5 Simulation Framework

A simulation framework for testing the quadrotor's stabilization and navigation controllers is developed. It is based on the open-source 3D robotics simulator Gazebo [14] and the ODE library [15]. The quadrotor dynamics model derived earlier is implemented in the Gazebo simulator to calculate the forces and moments acting on the vehicle. Using the Gazebo API interface, the derived forces and moments are input to the ODE. It solves the body's equations of motion and sends the vehicle's states back to Gazebo for visualization. The stabilization and navigation controllers are implemented directly in Gazebo such that they can use the vehicle's states for feedback and send the control commands to the dynamics module. The simulator framework is shown in Fig. 3. Gazebo offers plugins for different sensor packages that can be mounted on the quadrotor to provide sensor feedback for higher levels of autonomy. For example, the quadrotor can be equipped with onboard camera and laser sensors to provide feedback for visual servoing and simultaneous localization and mapping. Figure 4 shows a screen shot of the simulation environment where the quadrotor is equipped with additional sensors, camera and laser.

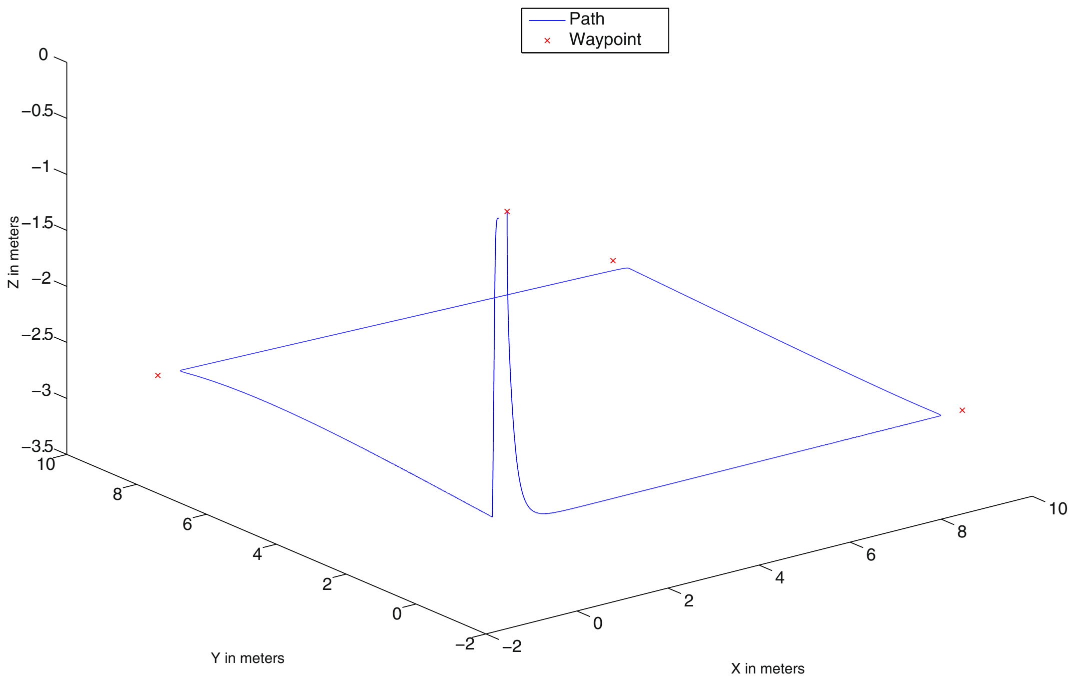

Fig. 7 Simulation results for waypoint following

6 Simulation Results

In this section, simulation results for the inner and outer loop controllers are presented.

6.1 Inner Loop Results

Figure 5 shows the closed loop response of the roll, pitch, yaw and altitude to a step input. The steady state is reached within approximately 2 s using bounded control inputs as shown in Fig. 6. The Backstepping controller asymptotically regulates the roll, pitch, yaw and altitude despite of the cross couplings between them and the inherent nonlinearities.

6.2 Outer Loop Results

Figure 7 shows the waypoint navigation performance of the proposed controller; it shows take-off, waypoint following and landing. Due to our coordinate system convention defined in Section 2, the altitude is negative in the upward direction. As shown in Fig. 7, the quadrotor is given four waypoints to follow at a desired altitude of 3 m. The quadrotor doesn't have to exactly reach the waypoint to advance to the next one, rather a threshold distance of 0.5 m is defined. The quadrotor first ascends to the desired altitude and starts following the programmed waypoints and finally descends to the original altitude. Figure 8 shows the reference roll and pitch angles generated by the position controller to reach the desired waypoints. Initially, the quadrotor is directly facing the first waypoint. The position controller generates negative reference pitch angles and regulates the roll angle to zero such that the quadrotor moves towards the first waypoint. Notice that the position controller doesn't maintain a negative pitch angle as it will cause the quadrotor's linear

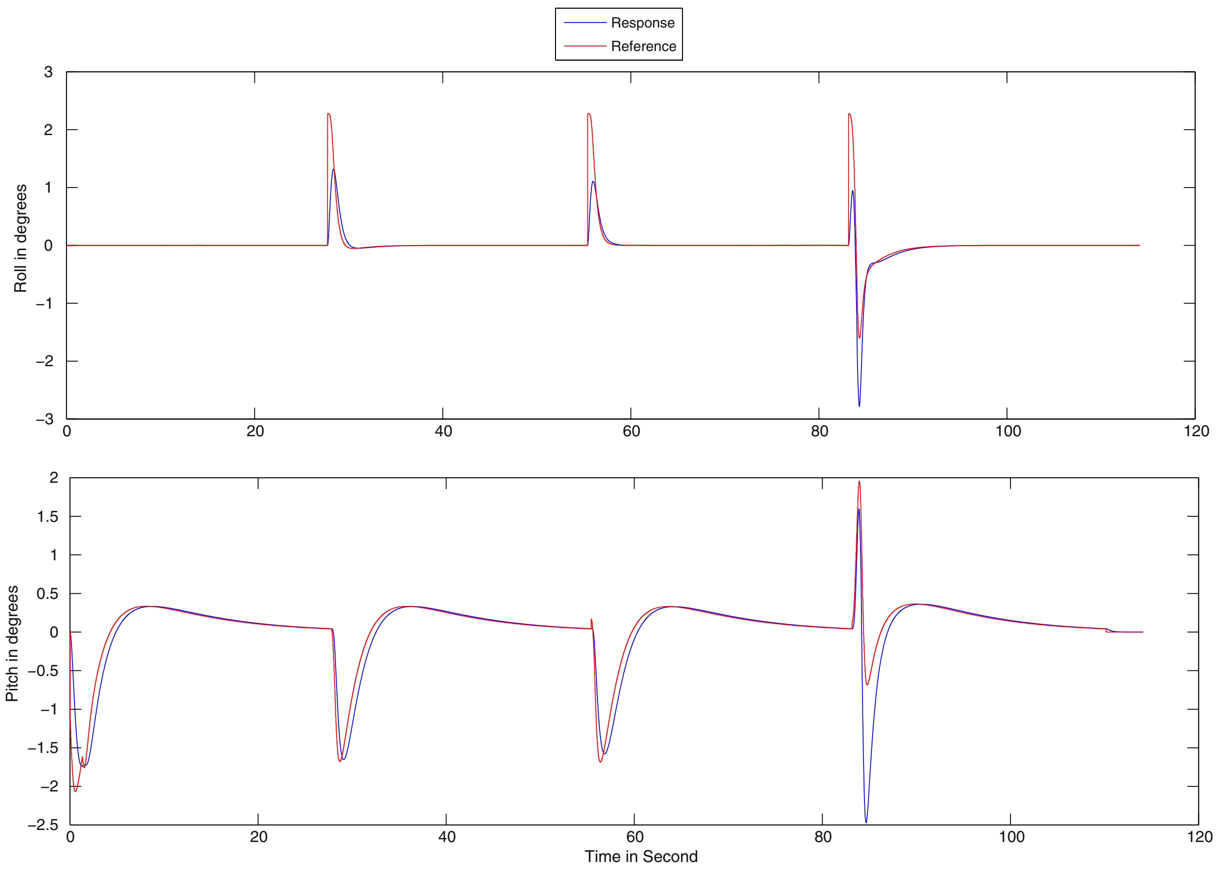

Fig. 8 Simulation results for position controller

acceleration to increase rapidly and overshoot the desired waypoint. Rather, it produces a large initial acceleration and decelerates the quadrotor gradually untill it reaches the desired waypoint at a low speed. At around 30 s, the quadrotor is within 0.5 m from the first waypoint and starts advancing to the second one. As the quadrotor performs the 90° turn, the position controller generates reference roll angles to control the quadrotor's position as it pitches forward during the turn. At around 110 s, the quadrotor reaches its final waypoint and attitude is stabilized to zero.

7 Conclusion

In this paper, a simulation framework for quadrotor stabilization and navigation is presented. A nonlinear quadrotor model is presented and used for controller design and simulator development. Based on the structure of the developed model, a nested loops control architecture is adopted for stabilization and navigation. The inner loop control system is designed using the nonlinear Backstepping method to asymptotically track reference attitude, altitude and heading trajectories. Stability of the controller is analyzed in terms of asymptotic tracking, transient tracking error and boundedness of control inputs. For position control, a PD controller with a saturation function is implemented to ensure boundedness of the generated reference trajectories. A simulation framework is developed to facilitate controller development and testing. The framework is based on a nonlinear quadrotor model implemented in the Gazebo robotics simulator using the ODE library. The framework provides rendering capabilities and access to a set of simulated sensors that can be used for testing autonomous behaviors. Simulation results demonstrate the effectiveness of the proposed controller and simulator. Future work includes further improvement in the developed simulation framework and integration with the well-known middleware, Robot Operating System (ROS) [21]. The developed simulation framework has been released and is available at [1].

Acknowledgements The authors would like to acknowledge the following funding agencies: Natural Sciences and Engineering Research Council of Canada, Canada Foundation for Innovation, Defence Research and Development Canada, and New Brunswick Innovation Foundation.

References

1. Gazebo quadrotor simulator. Available online http://sourceforge.net/projects/gazebo-quad-sim. Accessed 10 Mar 2012
2. Hoffmann, G., Waslander, S., Vitus, M., Huang, H., Gillula, J., Pradeep, V., Tomlin, C.: Stanford testbed of autonomous rotorcraft for multi-agent control. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009. IROS 2009, pp. 404–405 (2009)
3. Bouabdallah, S., Noth, A., Siegwart, R.: Pid vs lq control techniques applied to an indoor micro quadrotor. In: 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2004, (IROS 2004), Proceedings, vol. 3, pp. 2451–2456. 2 Sept – Oct (2004)
4. Bouabdallah, S., Siegwart, R.: Backstepping and sliding-mode techniques applied to an indoor micro quadrotor. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, Robotics and Automation 2005. ICRA 2005, pp. 2247–2252 (2005)
5. Mokhtari, A., Benallegue, A., Daachi, B.: Robust feedback linearization and gh infin; controller for a quadrotor unmanned aerial vehicle. In: 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005, (IROS 2005), pp. 1198–1203 (2005)
6. Waslander, S., Hoffmann, G., Jang, J.S., Tomlin, C.: Multi-agent quadrotor testbed control design: integral sliding mode vs. reinforcement learning. In: 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005, (IROS 2005), pp. 3712–3717 (2005)
7. Khalil, H.: Nonlinear Systems. Prentice Hall (2002)
8. Madani, T., Benallegue, A.: Backstepping control for a quadrotor helicopter. In: 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3255–3260 (2006)
9. Colorado, J., Barrientos, A., Martinez, A., Lafaverges, B., Valente, J.: Mini-quadrotor attitude control based on hybrid backstepping amp; frenet-serret theory. In: 2010 IEEE International Conference on Robotics and Automation (ICRA), pp. 1617–1622 (2010)
10. Mancini, A., Cesetti, A., Iual, A., Frontoni, E., Zingaretti, P., Longhi, S.: A framework for simulation and testing of uavs in cooperative scenarios. J. Intell. Robot. Syst. **54**, 307–329 (2009)
11. Rotorlib: Available online http://www.rtdynamics.com. Accessed 10 Mar 2012
12. Jsbsim: Available online http://jsbsim.sourceforge.net. Accessed 10 Mar 2012

13. Flightgear: Available online http://www.flightgear.org. Accessed 10 Mar 2012
14. Gazebo 3d Robotics Simulator: Available online http://playerstage.sourceforge.net/gazebo/gazebo.html. Accessed 10 Mar 2012
15. Open Dynamics Engine Library: Available online http://www.ode.org/. Accessed 10 Mar 2012
16. Gazebo Quad Sim: Available online http://sourceforge.net/projects/gazeboquadrotor. Accessed 10 Mar 2012
17. Lee, D., Jin Kim, H., Sastry, S.: Feedback linearization vs. adaptive sliding mode control for a quadrotor helicopter. Int. J. Control Autom. Syst. **7**, 419–428 (2009)
18. Bouabdallah, S.: Design and control of quadrotors with application to autonomous flying. Ph.D. dissertation, Lausanne (2007)
19. Michael, N., Mellinger, D., Lindsey, Q., Kumar, V.: The grasp multiple micro-uav testbed. IEEE Robot. Autom. Mag. **17**(3), 56–65 (2010)
20. Krstić, M., Kanellakopoulos, I., Kokotović, P.: Nonlinear and adaptive control design, ser. Adaptive and learning systems for signal processing, communications, and control. Wiley (1995)
21. Robot Operating System: Available online http://www.ros.org/. Accessed 10 Mar 2012