

Quadrotor Obstacle Avoidance Based on Integral Backstepping

¹Lianchuan Zhang, ¹Young Baik Kim and ²Kil To Chong

^{1,2}Department of Electronics Engineering, Chonbuk National University, Jeonju, South Korea

¹KEPCO Engineering and Construction Company, Ltd, Daejeon, South Korea

Article history

Received: 12-01-2016

Revised: 23-06-2016

Accepted: 24-06-2016

Corresponding Author:

Kil To Chong

Department of Electronics
Engineering, Chonbuk National
University, Jeonju, South Korea
Email: kitchong@jbnu.ac.kr

Abstract: This paper addressed obstacle avoidance problem of a quadrotor in outdoor environment. Path planning was finished by employing classical Dijkstra algorithm and the controller of the quadrotor adopted integral backstepping method. In order to get a smooth trajectory with time optimal and acceleration constraints properties, quantic polynomials were utilized to generate trajectory file. The scheme is particularly practical and useful because of its small computation burden and feasibility. Simulations have been done to verify the performance of the whole system. As a result, the quadrotor successfully tracked the path without any collision with obstacles except that there are some small error in the final part of position tracking performance ($\pm 1m$) and some small delay in the attitude tracking about 0.3 ms. In conclusion, the proposed method showed acceptable performance for quadrotor obstacle avoidance.

Keywords: Quadrotor, Obstacle Avoidance, Dijkstra, Integral Backstepping

Introduction

Quadrotors, as a kind of Unmanned Aerial Vehicles (UAVs), have being an attractive research area in recent years. In terms of dynamics modeling and controller design, many fantastic works have been published and demonstrate robust control performance (Geng *et al.*, 2013). Since fixed wing UAVs payload is large enough, it is possible for them to install kinds of equipment such as powerful communication facilities and high resolution stereo vision system, which give them an advantage over communicating with military satellites to successfully avoiding collision with other UAVs or aircrafts. Comparing to fixed wing UAVs, obstacle avoidance control of small UAVs such as quadrotors has been a very challenging problem because of its limited payload, which make it unrealistic to choose high powerful processor with more relative weight to implement complicated algorithms and to mount accurate sensors such stereo vision equipment. However, no matter where quadrotors are applied in indoor or outdoor environment, collision avoidance with obstacles is one of basic functions to finish complicated tasks. Hence it needs to pay attention to obstacle avoidance controller design and choosing hardware.

Recent approaches for path planning has many different directions. In terms of usage of sensors for obstacle avoidance, one main trend is based on stereo vision system by using image processing technics, which obtained impressive and robust performance

especially in indoor environment. However, its main drawback is that image processing implementation is offline, which means that it is not feasible for long distance surveillance in outdoor environment. Another research direction is based on some other different kind sensors instead of camera such as a laser range finder. Although such sensor can give online and real time information to microprocessor and they also can be applied into both indoor and outdoor environment, its experimental performance is not as good as such systems based on stereo vision system. When it comes to the algorithm processing raw map information sent by sensors, it has two main directions: Extending classical method successfully applied on ground mobile robots and proposing some other new approaches. Simultaneously Localization and Mapping (SLAM) is one of representatives of classical method of ground mobile robots obstacle avoidance (Grzonka *et al.*, 2012) and some other methods also obtain good results such as combining D* lite and Probabilistic Roadmap (Hrabar, 2008), Mixed Integer Linear Program (MILP) (Richards *et al.*, 2002), dynamically feasible interpolation (Dever *et al.*, 2006) and using motion primitives (Bottasso *et al.*, 2008).

This paper firstly derived the dynamics model for quadrotors based on Newton-Euler equations. In terms of control of the quadrotor, integral backstepping controller was adopted. The main part of this paper is path planning based on Dijkstra algorithm. By using this simple method, it can substantially alleviate the

computation burden problem, which make it possible to run on off shelf products. A discrete original path consisting of hundreds of points was firstly generated by using Dijkstra algorithm. However, it is not optimal. Hence, this paper proposed one measure to improve this original path: Condensing the original path by checking if it can directly reach the next point without collision, which can abridge the original path to make it time optimal. In order to taking velocity and acceleration constraints into account and meet desired velocity inputs for integral backstepping controller, quantic polynomials were chosen to generate smooth 3D trajectory. Similar forest environment was created under Simulink environment to evaluate the performance of the path planning method and integral backstepping controller. Simulation results showed that this new method exhibit nice performance for quadrotor obstacle avoidance. In comparison with other methods, the main contribution of this paper is incorporating Dijkstra algorithm and integral backstepping controller to generate a relatively time optimal 3D trajectory with substantially decreasing computation burden of microprocessors and considering dynamics constraints.

The dynamics modeling is described in section II. Section III gives an overview about integral backstepping controller. Simulation is presented in section IV. The results and discussion, presented in section V, is a clear illustration of performance of the whole system.

Quadrotor Dynamics Modeling

Quadrotor dynamics model is usually obtained by two different approaches: Euler-Lagrange and Newton-Euler equations and these two approaches can get same motion equations. There are many nonlinear factors of complete dynamics of a quadrotor such as free-stream velocity, blade flapping and gyroscopic effect. It would be very complicated even not feasible for the purpose of control if one dynamics model considers all nonlinear effect factors (Habib *et al.*, 2014). Therefore, this chapter builds a simplified dynamics model which retains main features and ignores some nonlinear effect factors such as free-stream velocity and blade flapping, which are easily observed in aggressive motions of large quadrotors (Garcia *et al.*, 2006; Kendoul *et al.*, 2009; 2010). In this chapter, the dynamics model will be derived based on Newton-Euler equations under the following assumptions:

- The quadrotor structure is supposed be symmetrical and rigid
- The center of geometry and the body fixed frame origin are assumed to coincide

- The propellers are supposed rigid
- Thrust and drug are proportional to the square of propeller's speed

In this study, the dynamics model will be derived based on Newton-Euler equations. Based on Newton-Euler formalism, the dynamics of a rigid body in body frame can be described as:

$$\begin{bmatrix} mI_3 & 0 \\ 0 & J \end{bmatrix} \begin{bmatrix} \dot{V} \\ \dot{\Omega} \end{bmatrix} + \begin{bmatrix} \Omega \times mV \\ \Omega \times J\Omega \end{bmatrix} = \begin{bmatrix} F_{ext} \\ \tau \end{bmatrix} \quad (1)$$

where, I is the identity matrix; $V = (u, v, w)$ and $\Omega = (p, q, r)$ are, respectively, the linear and angular velocities in the body-fixed frame; F_{ext} and τ are the total external force and torque, respectively; and J is the moment of inertia which is given by:

$$J = \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix} \quad (2)$$

Based on Euler angles parameterization, one can use a rotation matrix R to express translational dynamics in inertial frame, where R is defined as follows:

$$R = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix} \quad (3)$$

where $\eta = (\phi, \theta, \psi)$ denotes three Euler angles roll, pitch and yaw, respectively and s and c are abbreviations for \sin and \cos function. By considering this transformation, the translational dynamics of inertial frame are computed as follows:

$$m\dot{V} = RF_B - \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} \quad (4)$$

where $V = (\dot{x}, \dot{y}, \dot{z})$ the rotorcraft velocity in inertial frame and g is the gravitational acceleration and F_B is the total force excluding the gravity force.

The original Newton-Euler equations are derived based on rigid body which does not consider internal dynamics. Since electric motor's gyroscopic effect is obvious, this paper adds gyroscopic effect term to Newton-Euler equations. The new dynamics equations for rotational motion can be expressed as:

$$J\dot{\Omega} + \Omega \times J\Omega + \Omega \times [0 \ 0 \ J_r \Omega_r]^T = \tau \quad (5)$$

where J_r is motor's moment of inertia and Ω_r is residual angular speed of four motors (w_1, w_2, w_3, w_4), which can be expressed as:

$$\Omega_r = w_1 - w_2 + w_3 - w_4 \quad (6)$$

To transform attitude dynamics in body-fixed frame into inertia frame, we need the kinematic relation between Ω and $\dot{\eta}$:

$$\dot{\eta} = R_r \Omega \quad (7)$$

where the Euler matrix R_r is given by:

$$R_r = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi)\sec(\theta) & \cos(\phi)\sec(\theta) \end{bmatrix} \quad (8)$$

We can make assumption around hover state and small angles where $\psi \approx 0, \theta \approx 0$. Based on that assumption, this transformation matrix can be simplified to an identity matrix, which means that actually no changes are made on rotational dynamics. Depending on this approximation, the rotational dynamics of inertial frame can be calculated as follows:

$$J\ddot{\eta} + \dot{\eta} \times J\dot{\eta} + \Omega \times [0 \ 0 \ J_r \Omega_r]^T = \tau \quad (9)$$

A quadrotor is an under-actuated system with 6 degree of freedom and four control inputs, which are the total thrust U_1 and the torques (U_2, U_3, U_4). Hence the force and torque vectors in Equation 4 and 5 can be expressed as $F_B = [0 \ 0 \ U_1]$ and $\tau = [U_2 \ U_3 \ U_4]^T$, respectively. Under the assumption that thrusts are proportional to the square of propeller's speed, the relationship between control inputs (U_1, U_2, U_3, U_4) and rotors' speed (w_1, w_2, w_3, w_4) can be given by:

$$\begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} K_f & K_f & K_f & K_f \\ 0 & -K_f & 0 & K_f \\ -K_f & 0 & K_f & 0 \\ K_m & -K_m & K_m & -K_m \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix} \quad (10)$$

where K_f and K_m are the aerodynamic force and moment constants respectively. Recalling Equation 4 and 8, the nonlinear model of a quadrotor can be expressed in the

following form, where l denotes the distance between rotors' center and the center of mass:

$$\begin{cases} \ddot{\phi} = \dot{\theta}\dot{\psi} \left(\frac{I_y - I_z}{I_x} \right) - \frac{J_r}{I_x} \dot{\theta}\dot{\Omega} + \frac{l}{I_x} U_2 \\ \ddot{\theta} = \dot{\phi}\dot{\psi} \left(\frac{I_x - I_z}{I_y} \right) + \frac{J_r}{I_y} \dot{\phi}\dot{\Omega} + \frac{l}{I_y} U_3 \\ \ddot{\psi} = \dot{\psi}\dot{\theta} \left(\frac{I_x - I_y}{I_z} \right) + \frac{l}{I_z} U_4 \\ \ddot{z} = -g + (\cos\phi \cos\theta) \frac{1}{m} U_1 \\ \ddot{x} (\cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi) \frac{1}{m} U_1 \\ \ddot{y} (\cos\phi \sin\theta \sin\psi + \sin\phi \sin\psi) \frac{1}{m} U_1 \end{cases} \quad (11)$$

The dynamics model derived by this section is suitable for mini-quadrotor with small propellers and low flying speed. For a more accurate dynamics for large quadrotors model which considers blade flapping, big angles of attack, one can refer to (Hoffmann *et al.*, 2007) and paper (Bouabdallah, 2007).

Integral Backstepping Controller Design

Backstepping is a novel and practical approach that provides a recursive method for stabilizing the origin of a system in strict-feedback form. By constructing Lyapunov function for the closed-loop systems, backstepping controller guarantees system stability. However, integral backstepping has more robustness to some disturbances owing to integral action, which cancels steady error. In other words, integral backstepping is the combination between PID and backstepping. Integral backstepping controller was derived in Samir Bouabdallah's thesis (Bouabdallah, 2007). According to its results, integral backstepping controller is robust to external disturbances and can stabilize the quadrotor. The following equations are rotational (Equation 12) and translational (Equation 14) control laws of integral backstepping controller:

$$\begin{cases} U_2 = \frac{1}{b_1} \left[(1 - c_1^2 + \lambda_1) e_1 + (c_1 = c_2) \right. \\ \left. e_2 - c_1 \lambda_1 x_1 + \ddot{\phi}_d - x_4 x_6 a_1 - x_4 a_2 \Omega \right] \\ U_3 = \frac{1}{b_2} \left[(1 - c_3^2 + \lambda_2) e_3 + (c_3 = c_4) \right. \\ \left. e_4 - c_3 \lambda_2 x_2 + \ddot{\theta}_d - \dot{\phi}\dot{\psi} a_3 + \dot{\phi} a_4 \Omega \right] \\ U_4 = \frac{1}{b_3} \left[(1 - c_5^2 + \lambda_3) e_5 + (c_5 = c_6) \right. \\ \left. e_6 - c_5 \lambda_3 x_3 \right] \end{cases} \quad (12)$$

$$\begin{cases} e_1 = \phi_{ref} - \phi \\ e_2 = c_1 e_1 + \dot{\phi}_{ref} + \lambda_1 x_1 - \dot{\phi} \\ e_3 = \theta_{ref} - \theta \\ e_4 = c_3 e_3 + \dot{\theta}_{ref} + \lambda_2 x_2 \\ e_5 = \psi_{ref} - \psi \\ e_6 = c_5 e_5 + \dot{\psi}_{ref} + \lambda_3 x_3 - \dot{\psi} \end{cases} \quad (13)$$

$$\begin{cases} U_1 = \frac{m}{\cos \phi \cos \theta} \begin{bmatrix} g + (1 - c_7^2 + \lambda_4) e_7 \\ + (c_7 + c_8) e_8 - c_7 \lambda_4 x_4 \end{bmatrix} \\ U_x = \frac{m}{U_1} \begin{bmatrix} (1 - c_9^2 + \lambda_5) e_9 + (c_9 + c_{10}) e_{10} \\ - c_9 \lambda_5 x_5 \end{bmatrix} \\ U_y = -\frac{m}{U_1} \begin{bmatrix} (1 - c_{11}^2 + \lambda_6) e_{11} \\ + (c_{11} + c_{12}) e_{12} - c_{11} \lambda_6 x_6 \end{bmatrix} \end{cases} \quad (14)$$

$$\begin{cases} e_7 = z_{ref} - z \\ e_8 = c_7 e_7 + \dot{z}_{ref} + \lambda_4 x_4 - \dot{z} \\ e_9 = x_{ref} - x \\ e_{10} = c_9 e_9 + \dot{x}_{ref} + \lambda_5 x_5 - \dot{x} \\ e_{11} = y_{ref} - y \\ e_{12} = c_{11} e_{11} + \dot{y}_{ref} + \lambda_6 x_6 - \dot{y} \end{cases} \quad (15)$$

Solving Under-Actuated Problem

Quadrotor is an under-actuated system which can produce 6 outputs ($\phi, \theta, \psi, x, y, z$) with 4 inputs (U_1, U_2, U_3, U_4). From the translational motion equations, we can see that desired angles (ϕ_d, θ_d, ψ_d) are the outputs of position system. Considering ψ_d is given by an operator. One can use the following motion Equation 16 to solve under-actuated problem and get Equation 17:

$$\begin{cases} U_z = -g + (\cos \phi \cos \theta) \frac{1}{m} U_1 \\ U_x = (\cos \phi \sin \theta \cos \psi + \sin \phi \sin \psi) \frac{1}{m} U_1 \\ U_y = (\cos \phi \sin \theta \sin \psi - \sin \phi \sin \psi) \frac{1}{m} U_1 \end{cases} \quad (16)$$

$$\begin{cases} U_1 = m \sqrt{(U_x^2 + U_y^2 + (g + U_z)^2)} \\ \phi_d = \sin^{-1} \left(\frac{U_x \sin \psi_d - U_y \cos \psi_d}{U_1} \right) \\ \theta_d = \tan^{-1} \left(\frac{U_x \cos \psi_d + U_y \sin \psi_d}{g + U_z} \right) \end{cases} \quad (17)$$

With Equation 17, the under-actuated problem is solved and then the control system advocated for the overall system is schematized in Fig. 1. As the Fig. 1 shown, the two controllers, position controller and rotation controller, are connected by nonlinear Equation 17.

Obstacles Avoidance Scheme

Obstacle avoidance involves detecting obstacle, avoiding them and finally getting optimal path. In this part, Dijkstra algorithm was applied to finish path planning. Because it has following advantages: Simple, fast and low computation burden. Based on Dijkstra algorithm, path planning was finished in several steps:

- Getting obstacle and map boundary information to finish mapping, as Fig. 2 shown
- Utilizing Dijkstra algorithm to get original discrete path, as Fig. 2 shown
- Exploiting quantic polynomials to generate trajectory file with acceleration and average velocity constraints

In terms of Dijkstra algorithm, it firstly obtains original discrete path consisting of points. After mapping, the whole procedure can be divided into three steps:

- Picking the unvisited vertex with the lowest-distance
- Calculating the distance through it to each unvisited neighbor and updating neighbor's distance if smaller
- Marking visited when done with neighbors

This original discrete path consisted of hundreds of points. However, it is not optimal. In order to make it more time optimal, this chapter condensed the discrete path through checking if the quadrotor can directly arrive the next point without collision with obstacle. After finishing this step, it obtained a condensed path composed of much less points, which were marked by five blue stars, as shown in Fig. 3. Comparing to the original Dijkstra path's length (56.00 m), this condensed path's length is 38.46 m, which is shorter about 31% than the original one. This condensed path are also treated as the reference path for simulation experiments. Another advantage of this condensed path is that it is convenient to generate smooth trajectory, because sometimes it can fail to directly generate smooth trajectory if the distance between two adjacent points is too small.

Since integral backstepping controller needs desired velocity inputs, quantic polynomials were chosen to generate smooth 3D trajectory with acceleration constraints. Formulating the desired trajectory as a 5 order polynomial offers two advantages. Firstly, constraints on velocities and accelerations are easily incorporated into the reference path.

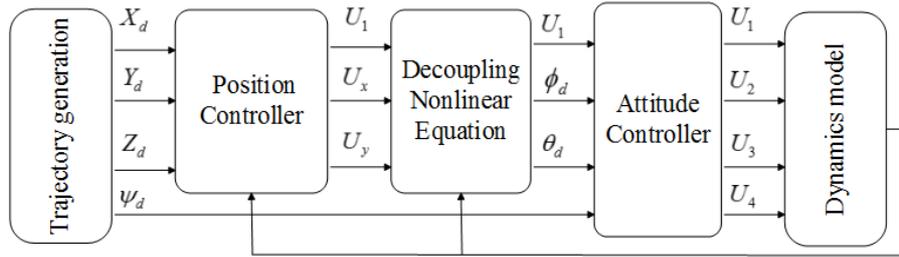


Fig. 1. Structure of control system

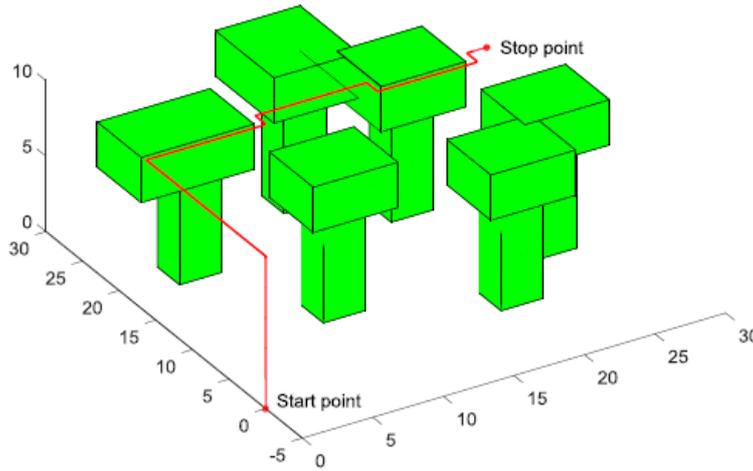


Fig. 2. Defining boundaries of obstacle and the map

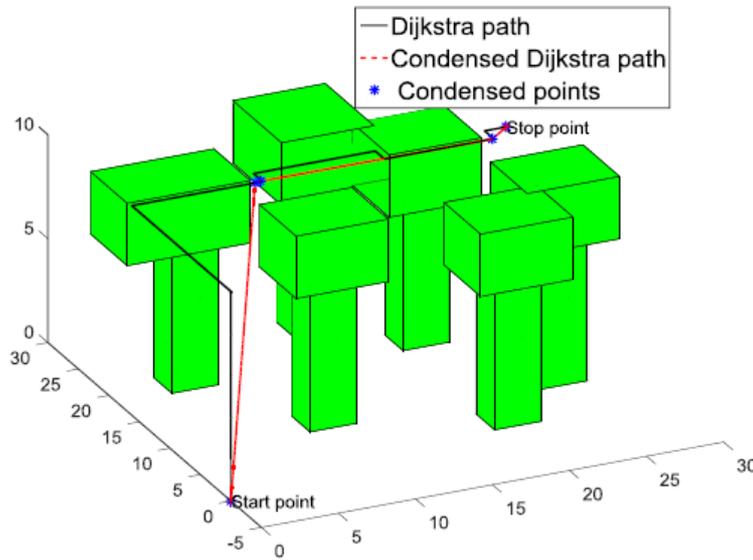


Fig. 3. Dijkstra path and condensed path

Secondly, providing boundary conditions to ensure the continuity of the first four derivatives of the reference path, which means that those inputs to quadrotor will be smooth. The reference trajectory between two points can be defined as:

$$r(t) = a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \quad (18)$$

It is obvious that initial and final velocities and accelerations are zero. So the following equation can be obtained:

$$\begin{bmatrix} dr \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} t_f^3 & t_f^4 & t_f^5 \\ 3 & 4t_f & 5t_f^2 \\ 6 & 4t_f & 20t_f^2 \end{bmatrix} \begin{bmatrix} a_3 \\ a_4 \\ a_5 \end{bmatrix} \quad (19)$$

With:

$$dr = r_f - r_i \quad (20)$$

Solving for coefficients:

$$\begin{bmatrix} a_3 \\ a_4 \\ a_5 \end{bmatrix} = \begin{bmatrix} t_f^3 & t_f^4 & t_f^5 \\ 3 & 4t_f & 5t_f^2 \\ 6 & 4t_f & 20t_f^2 \end{bmatrix}^{-1} \begin{bmatrix} dr \\ 0 \\ 0 \end{bmatrix} \quad (21)$$

Thus the desired trajectory between two points is given by:

$$r(t) = x_i + a_3t^3 + a_4t^4 + a_5t^5 \quad (22)$$

After getting Equation 22, it is easy to get parameters for the desired trajectory if traveling time between two points is known.

Here is the method that how we use the above equation with acceleration constraints to generate smooth desired trajectory:

- Defining average speed V_{avg} and maximum acceleration a_{max}
- Calculating traveling time t_f by using average speed and distance between two points
- Using Equation 22 to get parameters of the desired trajectory
- Checking maximum acceleration to see if it exceeds a_{max} . If not, those parameters are acceptable; if it does, it reduces the average speed to 80% of the

former one and iterates until parameters of desired trajectory are acceptable

By taking constraints on actuators of a quadrotor into account, this chapter added acceleration and average speed constraints to trajectory generation. Finally, a 3D trajectory was generated by exploiting quintic polynomials to guarantee the trajectory staying on the reference path.

Simulation and Results

In order to evaluate the performance of path planning algorithm and integral backstepping controller, this paper, based on Simulink environment, built outdoor simulate forest environment for quadrotor simulation tests, as shown in Fig. 2. Firstly, Dijkstra algorithm finished path planning tasks and then trajectory was generated by using Equation 18 to 22. The generated trajectory could give desired position and velocity inputs to integral backstepping controller. By exploiting Simulink Response Optimization toolbox, this paper obtained all the parameters for integral backstepping controller.

As Fig. 4 shown, the quadrotor was command to fly from starting point (0, 0, 0) to final points (25, 18, 10) with no collision with trees and it succeeded to finish this task, which the desire yaw angle was given by the pilot. Figure 5 and 6 presented responses of position and attitude, respectively. In terms of position response, it tracked the reference trajectory successfully in most of the time except that there are some small error ($\pm 1m$) in the end. In comparison, attitude response has some obvious time delays about 0.3 ms but the yaw repose showed nice tracking performance. In conclusion, the whole system including path planning and integral backstepping controller works well and finishes the task successfully.

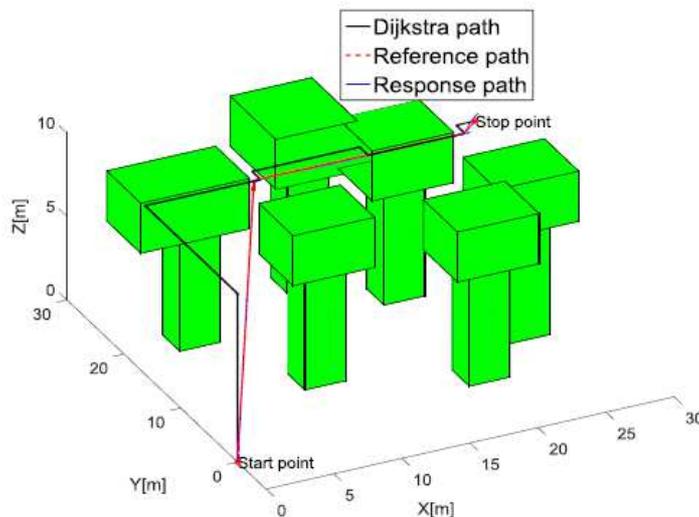


Fig. 4. Obstacle avoidance simulation

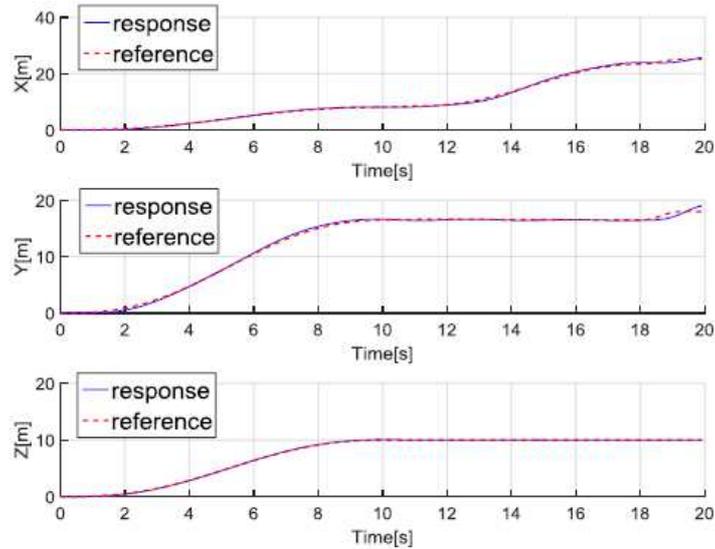


Fig. 5. Position response of obstacle avoidance simulation

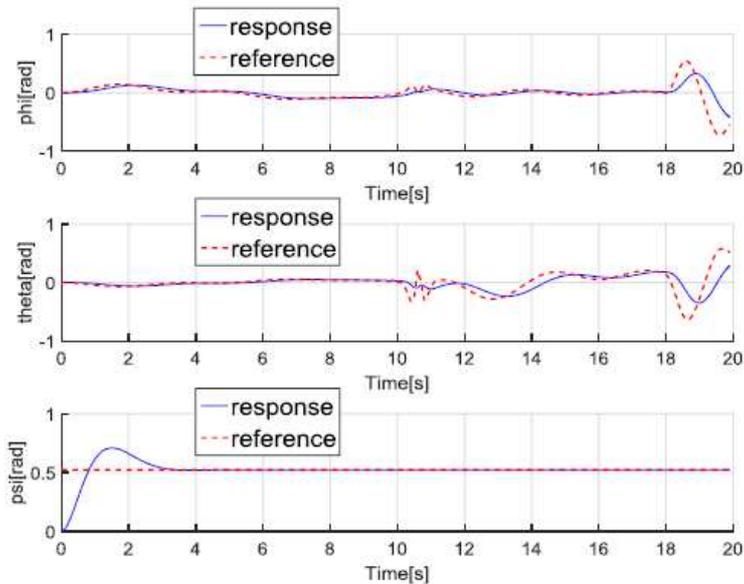


Fig. 6. Attitude response of obstacle avoidance simulation

Conclusion

This paper presented one obstacle avoidance method for a quadrotor. A nonlinear dynamics model was studied firstly. Integral backstepping controller was proposed to tackle the stable problem of the quadrotor. Dijkstra algorithm was applied to obtain discrete path. In order to make the path optimal and feasible, this paper condensed original discrete path into several points and added accelerations and velocities constrains to the smooth trajectory, which was generated using quantic polynomials. Simulation results show that the

quadrotor successfully complete the task with very little tacking error.

Many breakthroughs of path planning about quadrotors has been made, especially some algorithm shows impressive performance with stereo vision system. Under supervision of video system, one quadrotor can finish very complicated and difficult tasks such as holding one cup of wine or several quadrotors cooperate with each other to do some big and sophisticated tasks such as building something. With the advances of technology, there functions can be achieved in cluttered and long range environments.

Acknowledgement

The research is supported by BK21 PLUS and by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (NRF-2013R1A2A2A01068127).

Author's Contributions

Lianchuan Zhang: This author finished integral back stepping controller design, applied Dijkstra algorithm and finished simulation experiments.

Young Baik Kim: Finished quad rotor dynamics modeling and joined to tune integral backstepping controller parameters.

Kil To Chong: Gave lots of suggestions on developing this papers and finished most of work for publication.

Ethics

This article is original and contains unpublished material. The corresponding author confirms that all of the other authors have read and approved the manuscript and no ethical issues involved.

References

- Bottasso, C.L., D. Leonello and B. Savini, 2008. Path planning for autonomous vehicles by trajectory smoothing using motion primitives. *IEEE Trans. Control Syst. Technol.*, 16: 1152-1168. DOI: 10.1109/TCST.2008.917870
- Bouabdallah, S., 2007. Design and control of quadrotors with application to autonomous flying. Ecole Polytechnique Federale de Lausanne.
- Dever, C., B. Mettler, E. Feron, J. Popovic and M. Mcconley, 2006. Nonlinear trajectory generation for autonomous vehicles via parameterized maneuver classes. *J. Guidance Control Dynam.*, 29: 289-302.
- Garcia, P.C., R. Lozano and A.E. Dzul, 2006. *Modelling and Control of Mini-Flying Machines*. 1st Edn., Springer Science and Business Media, London, ISBN-10: 1846281792, pp: 252.
- Geng, Q., H. Shuai and Q. Hu, 2013. Obstacle avoidance approaches for quadrotor UAV based on backstepping technique. *Proceedings of the 25th Chinese Control and Decision Conference*, May 25-27, IEEE Xplore Press, Guiyang, pp: 3613-3617. DOI: 10.1109/CCDC.2013.6561575
- Grzonka, S., G. Grisetti and W. Burgard, 2012. A fully autonomous indoor quadrotor. *IEEE Trans. Robot.*, 28: 90-100. DOI: 10.1109/TRO.2011.2162999
- Habib, M.K., W.G.A. Abdelaal and M.S. Saad, 2014. Dynamic modeling and control of a quadrotor using linear and nonlinear approaches. American University in Cairo.
- Hoffmann, G.M., H. Huang, S. Waslander and C. Tomlin, 2007. Quadrotor helicopter flight dynamics and control: Theory and experiment. *Proceedings of the AIAA Guidance, Navigation and Control Conference Exhibit*, (CCE' 07), Hilton Head, South Carolina, pp: 1-20. DOI: 10.2514/6.2007-6461
- Hrabar, S., 2008. 3D path planning and stereo-based obstacle avoidance for rotorcraft UAVs. *Proceedings of the International Conference on Intelligent Robots and Systems*, Sept. 22-26, IEEE Xplore Press, Nice, pp: 807-814. DOI: 10.1109/IROS.2008.4650775
- Kendoul, F., Y. Zhenyu and K. Nonami, 2009. Embedded autopilot for accurate waypoint navigation and trajectory tracking: Application to miniature rotorcraft UAVs. *Proceedings of the IEEE International Conference on Robotics and Automation*, May 12-17, IEEE Xplore Press, Kobe, pp: 2884-2890. DOI: 10.1109/ROBOT.2009.5152549
- Kendoul, F., Z. Yu and K. Nonami, 2010. Guidance and nonlinear control system for autonomous flight of minirotorcraft unmanned aerial vehicles. *J. Field Robot.*, 27: 311-334. DOI: 10.1002/rob.20327
- Richards, A., T. Schouwenaars, J.P. How and E. Feron, 2002. Spacecraft trajectory planning with avoidance constraints using mixed-integer linear programming. *J. Guidance Control Dynam.*, 25: 755-764. DOI: 10.2514/2.4943