The Desktop Quad
with an LQR setpoint controller

Parker Lusk

April 21, 2017

Abstract

The Desktop Quad is an effort to allow an inexpensive quadrotor to be used as a case study in controls, vision processing, and autopilot design scenarios. It features an 8cm x 8cm quadrotor with an upward-facing camera for localization. The (really) micro air vehicle (MAV) is tethered using silicone wire for communications and power, allowing indefinite flight. This report describes the progress of the Desktop Quad achieving fully autonomous flight.

1 Introduction

My project was to add an LQR setpoint controller to a multirotor system.

2 System Modeling

We begin by modeling the Desktop Quad multirotor and deriving the equations of motion. The 12 state variables that will be used to describe the equations of motions are

$$x = \begin{bmatrix} p_n & p_e & p_d & \dot{p}_n & \dot{p}_e & \dot{p}_d & \phi & \theta & \psi & p & q & r \end{bmatrix}^T$$

where all the states are inertial coordinate frame quantities except for the angular rates p, q, and r which are in the body frame.

2.1 Kinematics and Dynamics

In order to analyze, simulate, and control the Desktop Quad, a 6DOF dynamic model must be derived. This is done using kinematics and dynamics. The goal is to find the evolution equations for each of the quantities we care about, as found in the state vector above.

$$\begin{bmatrix} \ddot{p}_n \\ \ddot{p}_e \\ \ddot{p}_d \end{bmatrix} = R_b^i(\phi, \theta, \psi) \begin{bmatrix} 0 \\ 0 \\ -T \end{bmatrix} \frac{1}{m} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{1}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & s_\psi t_\theta & c_\phi t_\theta \\ 0 & c_\phi & -s_\phi \\ 0 & \frac{s_\phi}{c_\theta} & \frac{c_\phi}{c_\theta} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{2}$$

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \Gamma_1 pq - \Gamma_2 qr \\ \Gamma_5 pr - \Gamma_6 (p^2 - r^2) \\ \Gamma_7 pq - \Gamma_1 qr \end{bmatrix} + \begin{bmatrix} \Gamma_3 l + \Gamma_4 n \\ \frac{1}{J_y} m \\ \Gamma_4 l + \Gamma_8 n \end{bmatrix} \tag{3}$$

2.2 Forces and Moments

3 Control Architecture

The Desktop Quad uses the ROSflight [4] stack for autonomous flight and thus has an on-board autopilot for attitude control. This simplifies the control problem and eliminates the need to control the moments l, m, and n. The LQR controller developed here will close the attitude loop using a position setpoint controller with the simplified state vector

$$x = \begin{bmatrix} p_n & p_e & p_d & \dot{p}_n & \dot{p}_e & \dot{p}_d & \psi \end{bmatrix}^T$$

with input to the ROSflight autopilot

$$\nu = \begin{bmatrix} T & \phi & \theta & r \end{bmatrix}^T .$$

Because of how we have defined our state, we will need to create a nonlinear map between the control u produced by our controller and the input ν to the autopilot. We can use the kinematic and dynamic relationships to define the map

$$\boldsymbol{u} = \begin{bmatrix} u_{p_{3x1}} \\ u_{\psi_{1x1}} \end{bmatrix}, \tag{4}$$

where

$$u_p = R_b^i(\phi, \theta, \psi) \begin{bmatrix} 0 \\ 0 \\ -T \end{bmatrix} \frac{1}{m} \tag{5}$$

and

$$u_\psi = q\frac{sin\phi}{cos\theta} + r\frac{cos\phi}{cos\theta}. \tag{6}$$

Inverting these relationships will give us a mapping for how u affects v, which can be feed into autopilot. The state-space system can be represented as

$$\dot{x} = Ax + Bu + bg \tag{7}$$

with

$$A = \begin{bmatrix} 0_{3x3} & I_{3x3} & 0_{3x1} \\ 0_{3x3} & 0_{3x3} & 0_{3x1} \\ 0_{1x3} & 0_{1x3} & 0_{1x1} \end{bmatrix}, \quad B = \begin{bmatrix} 0_{3x3} & 0_{3x1} \\ I_{3x3} & 0_{3x1} \\ 0_{1x3} & 1 \end{bmatrix}, \quad b = [0 \ \ 0 \ \ 0 \ \ 0 \ \ 0 \ \ 1 \ \ 0]^T \tag{8}$$

4 Building the System

5 Visual Localization

6 Simulation

7 Control Architecture

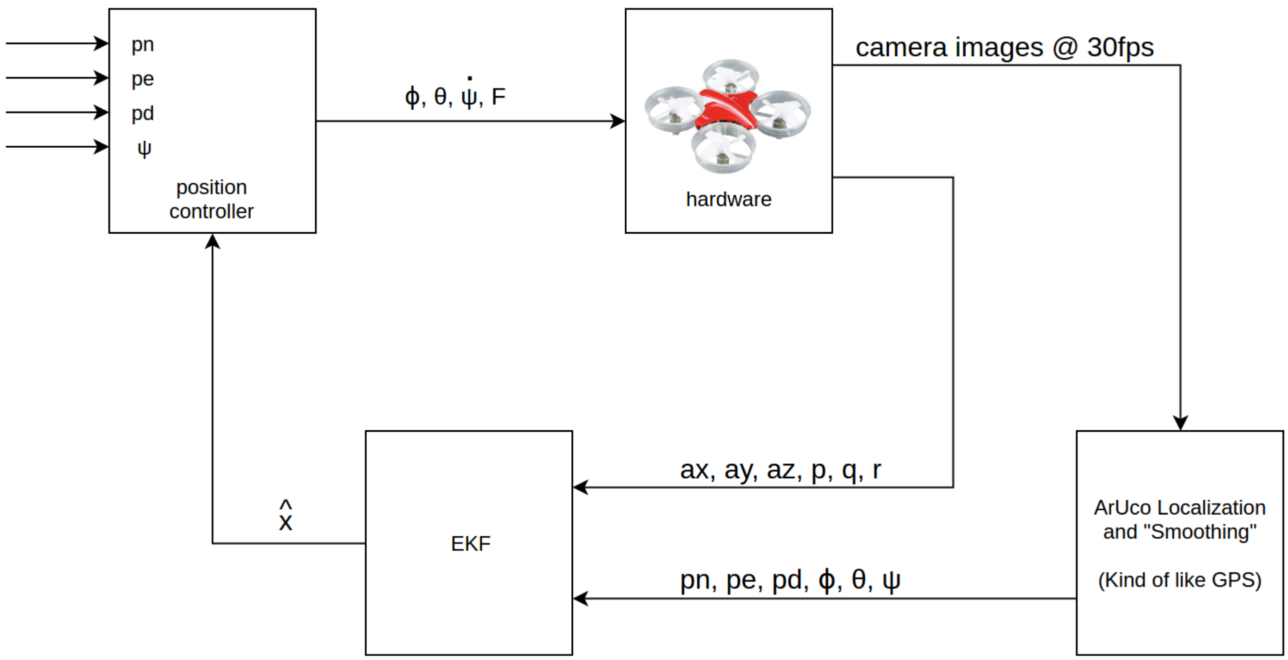

Figure 1: Control architecture of the Desktop Quad system.

References

[1] R. W. Beard and T. W. McLain, "Small unmanned aircraft," 2011.

[2] R. W. Beard and T. W. Mclain, "Introduction to Feedback Control using Design Studies," 2016.

[3] J. Ferrin, R. Leishman, R. Beard, and T. McLain, "Differential flatness based control of a rotorcraft for aggressive maneuvers," IEEE Int. Conf. Intell. Robot. Syst., pp. 2688–2693, 2011.

[4] J. Jackson, G. Ellingson, and T. McLain, "ROSflight: A lightweight, inexpensive MAV research and development tool," 2016 Int. Conf. Unmanned Aircr. Syst. ICUAS 2016, pp. 758–762, 2016.

[5] E. R. Albasiouny, A. Sarhan, and T. Medhat, “Mean-shift-FAST algorithm to handle motion-blur with tracking fiducial markers,” Proc. - 2015 10th Int. Conf. Comput. Eng. Syst. ICCES 2015, no. December, pp. 286–292, 2016.

[6] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, “Automatic generation and detection of highly reliable fiducial markers under occlusion,” Pattern Recognit., vol. 47, no. 6, pp. 2280–2292, 2014.

[7] E. Olson, “AprilTag: A robust and flexible visual fiducial system,” Proc. - IEEE Int. Conf. Robot. Autom., pp. 3400–3407, 2011.

[8] M. G. Prasad, S. Chandran, and M. S. Brown, “A motion blur resilient fiducial for quadcopter imaging,” Proc. - 2015 IEEE Winter Conf. Appl. Comput. Vision, WACV 2015, pp. 254–261, 2015.