Following Trajectories on a Quadrotor using a Linear Feedback Model

Guan Sun
School of Engineering and Applied Science
University of Pennsylvania
Email: guansun@seas.upenn.edu

Nitin J. Sanket
School of Engineering and Applied Science
University of Pennsylvania
Email: nitinsan@seas.upenn.edu

Caio Cesar R. Mucchiani
School of Engineering and Applied Science
University of Pennsylvania
Email: caio@seas.upenn.edu

***Abstract*—A linear control model for quadrotor trajectory following is presented in this report. Smooth path stitching between waypoints is accomplished by the use of cubic spline. The linearized approximation of a non-linear model is used for a proportional-derivative based control. Position control is carried out on the position data obtained from VICON motion capture system.**

r	Desired position vector
r_{act}	Current position vector
r_{cmd}	Commanded acceleration
ψ	Desired yaw
ψ_{act}	Current yaw
e	Error
g	Acceleration due to gravity (9.81 m/s)
m	Mass of the quadrotor
K_p	Proportional gain
K_d	Derivative gain
F_{tot}	Total force exerted by the motors

I. Introduction

Control of quadrotors has gained momentum in the recent years [1] due to the cost effectiveness of manufacture of these systems. A lot of work is still ongoing in this area to design robust control systems under varied conditions.
The general control strategies can be divided into two main sub-categories [2], namely, (a) Linear Control and (b) Non-Linear Control. The later strategy is more computation intensive and robust and can handle almost any 3D pose. However, for our purposes a linear control model is sufficient as the trajectories are smooth and do not involve extreme maneuvers [3].

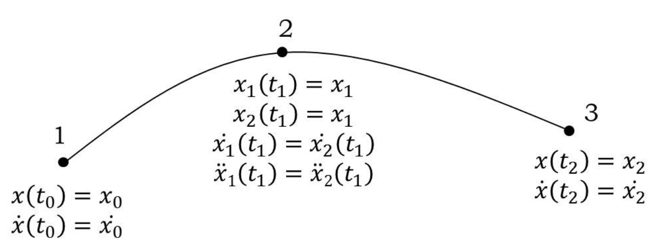

Fig. 1. Cubic Spline via 3 points.

II. Problem Definition and Solutions

The problem statement was to develop and implement algorithms for path planning (using A^* search) [4], collision avoidance, path shortening and collision avoidance (Collectively called trajectory generation). The final phase of the project was to test out the trajectory generation and control algorithms [5] on a KMel Nano+ quadrotor [6]. The problems in implementing the aforementioned is that a quadrotor inherently has non-linear dynamics and we are trying to fit a linear model to it. Also the path planning has to be fast enough for real time control of the quadrotor. Some of the solutions we came up with for these problems were:
(a) Make the path smooth and continuous without aggressive maneuvers so that a linear control model would suffice.
(b) Path planning was performed using a cubic spline trajectory generator.

III. Linear Control Model

To linearize the non-linear dynamics of a quadrotor the following assumptions are made:

1) The quadrotor is near the "hover" position, i.e., banking angles are small.
2) The actual motor velocities ω_i are equal to the commanded motor velocities ω_{des}.
3) The quadrotor is symmetric with respect to X and Y axes.

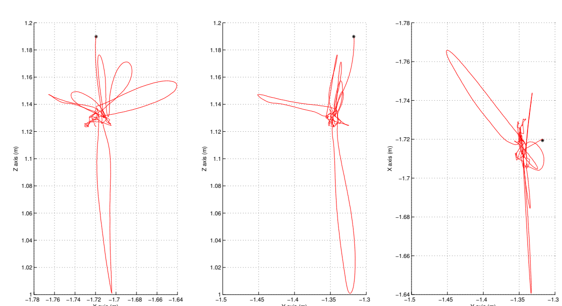

Fig. 2. Data Log for Hover.

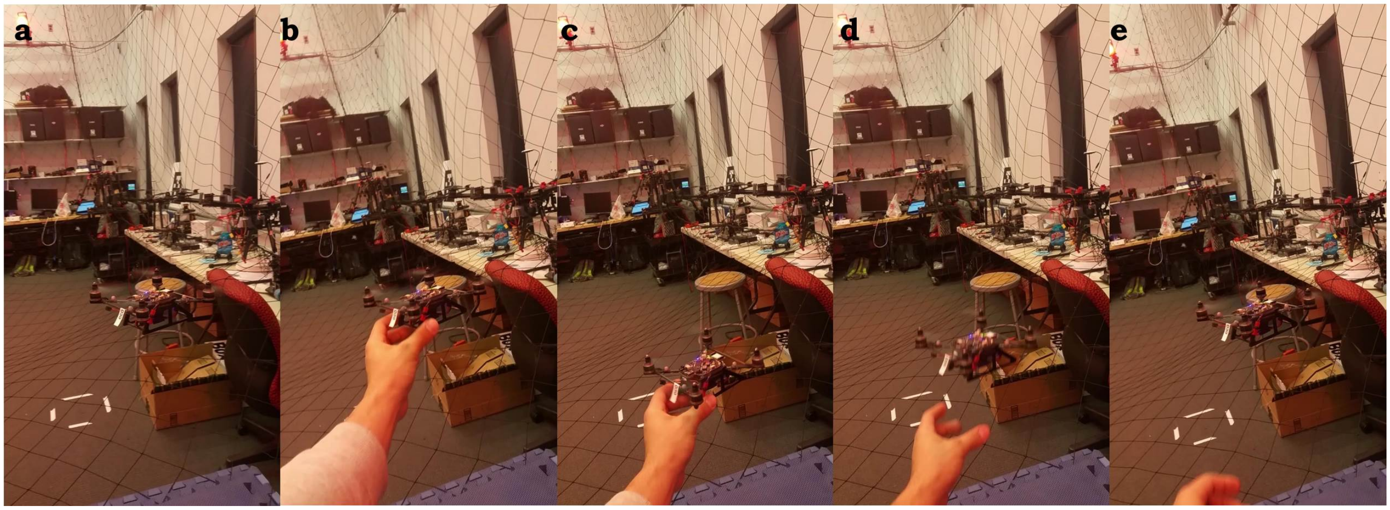

Fig. 3. Rejection of distrubances in hover state: (a) Quadrotor in hover state, (b) & (c) Sequence showing disturbance being applied in -Z direction, (d) Quadrotor released, and (e) Quadrotor goes back to desired hover state.

A. Controller Design

For near-hover state, we get the position error as,

$$e_i = (r_{i,act} - r_i)$$

The required condition for this error to exponentially go to zero is,

$$(\ddot{r}_{i,act} - \ddot{r}_{i,cmd}) + K_{d,i}\left(\dot{r}_{i,act} - \dot{r}_i\right) + K_{p,i}\left(r_{i,act} - r_i\right) = 0$$

Hence the commanded acceleration is,

$$\ddot{r}_{i,cmd} = \ddot{r}_{i,act} + K_{d,i}\left(\dot{r}_{i,act} - \dot{r}_i\right) + K_{p,i}\left(r_{i,act} - r_i\right)$$

Since the bank angle changes are small, we get,

$$\Delta\theta = \theta - \theta_0 \approx \theta,\ \Delta\phi = \phi - \phi_0 \approx \phi$$

$$\begin{aligned}
\ddot{r}_{1,des} &= g\left(\theta_{des}\cos\psi_{act} + \phi_{des}\sin\psi_{act}\right) \\
\ddot{r}_{2,des} &= g\left(\theta_{des}\sin\psi_{act} - \phi_{des}\sin\psi_{act}\right) \\
\ddot{r}_{3,des} &= \tfrac{1}{m}F_{tot} - g
\end{aligned}$$

Rearranging the above equations we get,

$$F_{tot} = mg + m\ddot{r}_{3,des}$$

Also the desired roll and pitch angles are given by,

$$\begin{aligned}
\phi_{des} &= \tfrac{1}{g}\left(\ddot{r}_{1,des}\sin\psi_{act} - \ddot{r}_{2,des}\cos\psi_{act}\right) \\
\theta_{des} &= \tfrac{1}{g}\left(\ddot{r}_{1,des}\cos\psi_{act} + \ddot{r}_{2,des}\sin\psi_{act}\right)
\end{aligned}$$

As the heading yaw direction does not matter to us, we set it to zero, i.e.,

$$\psi_{des} = 0$$

The position PD controller gains used by us were $K_p = [10, 10, 15]$ and $K_d = [5, 5, 7]$ (represented as [X, Y, Z] axes gains).

IV. Cubic Spline Trajectory Via Waypoints

To minimize the square functional for a minimum acceleration trajectory, we obtain the minimum order of the equation to be 3 from Euler-Lagrange equation. A simple example of how this trajectory is formed for 3 waypoints is illustrated in Fig. 1.

Let the spline between points 1 and 2 be represented as

$$f(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$

and the spline between points 2 and 3 be

$$f(t) = b_0 + b_1 t + b_2 t^2 + b_3 t^3$$

Now, we have to solve the following matrix equation to get the coefficients,

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 1 & t_0 & {t_0}^2 & {t_0}^3 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2t_0 & 3{t_0}^2 & 0 & 0 & 0 & 0 \\ 1 & t_1 & {t_1}^2 & {t_1}^3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & t_1 & {t_1}^2 & {t_1}^3 \\ 0 & 1 & 2t_1 & 3{t_1}^2 & 0 & -1 & -2t_1 & -3{t_1}^2 \\ 0 & 0 & 2 & 6t_1 & 0 & 0 & -2 & -6t_1 \\ 0 & 0 & 0 & 0 & 1 & t_2 & {t_2}^2 & {t_2}^3 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2t_2 & 3{t_2}^2 \end{bmatrix}^{-1} \begin{bmatrix} x_0 \\ \dot{x}_0 \\ x_1 \\ x_1 \\ 0 \\ 0 \\ x_2 \\ \dot{x}_2 \end{bmatrix}$$

The same concept was used for n waypoints (waypoint counting starts from 0), where we get $4 \times n$ coefficients and conditions and the square matrix is of size $4n \times 4n$.

V. Implementation and Results

The KMel Nano+ quadrotor [7] has the communication architecture shown in Fig. 4.

As seen in Fig. 4, there are two MATLAB instances running at any moment in time. They are named High Level Matlab (HLM) and Low Level Matlab (LLM). LLM runs the control loop of the quadrotor and also runs the trajectory generator during system initialization, whereas HLM sends the action command(s) and waypoint information to the LLM.

We conducted 3 experiments to quantify the performance of our algorithms,

1) *Hover:* To test the hover controller, here we command the quadrotor to stay at a fixed position in space wherein we apply disturbance and check if the quadrotor returns to its fixed position. This experiment was helpful in fine tuning the gains of the quadrotor.
2) *One Waypoint:* This experiment was used to test if

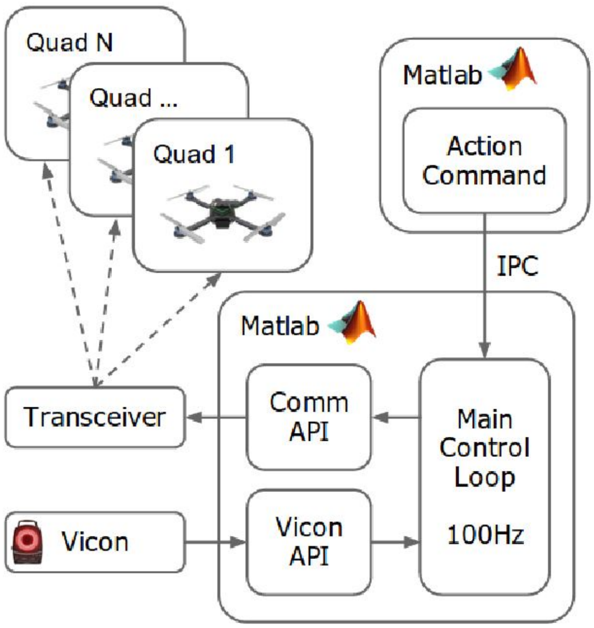

Fig. 4. Communication Architecture of KMel Nano+ quadrotor (Image adopted from [8]).

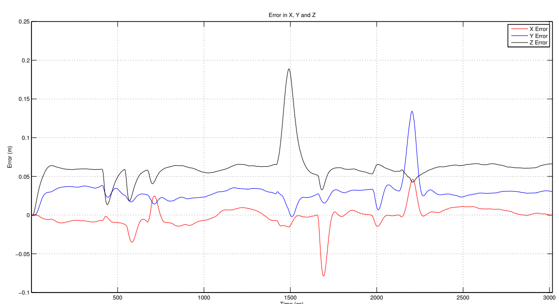

Fig. 5. Error for Hover.

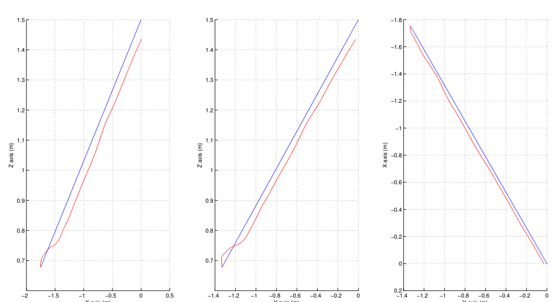

Fig. 6. Data Log for Single Waypoint.

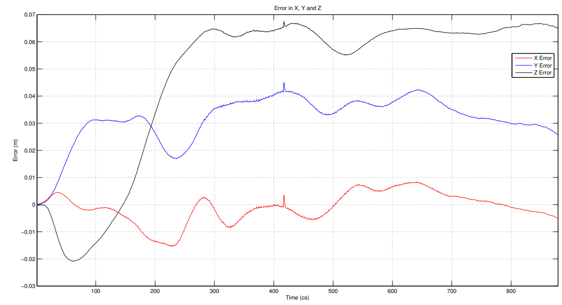

Fig. 7. Error for Single Waypoint.

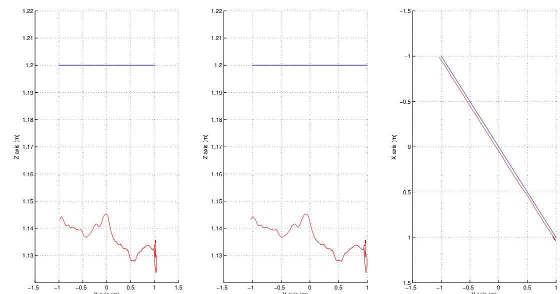

Fig. 8. Desired and Actual Trajectories for Waypoints 1.

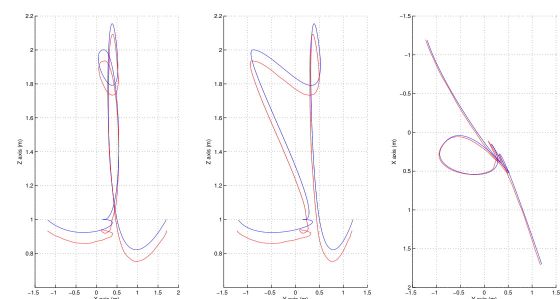

Fig. 9. Desired and Actual Trajectories for Waypoints 2.

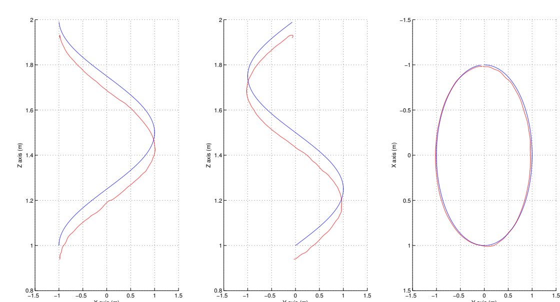

Fig. 10. Desired and Actual Trajectories for Waypoints 3.

the quadrotor can go to a commanded waypoint from hover.

3) *Multi Waypoints:* This experiment was used to test if the quadrotor can go through a set of commanded waypoints from hover. This test was intended to test our cubic spline trajectory generator and the controller.

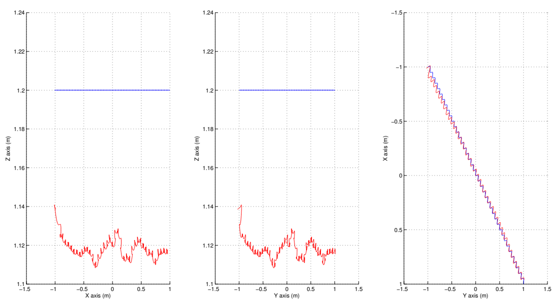

Fig. 11. Desired and Actual Trajectories for Waypoints 4.

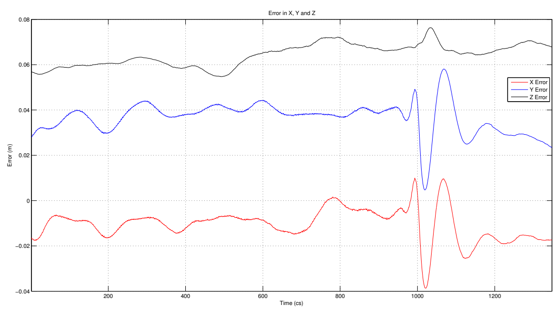

Fig. 12. Error for Waypoints 1.

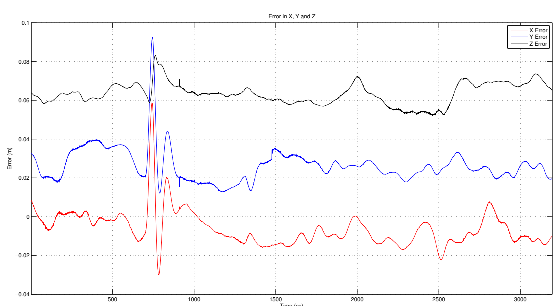

Fig. 13. Error for Waypoints 2.

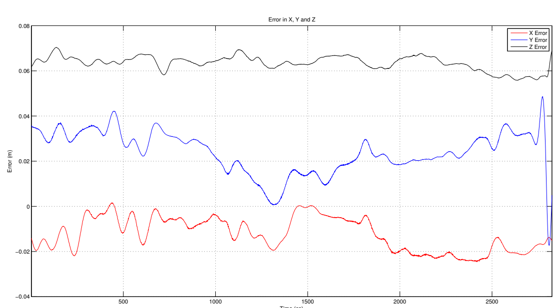

Fig. 14. Error for Waypoints 3.

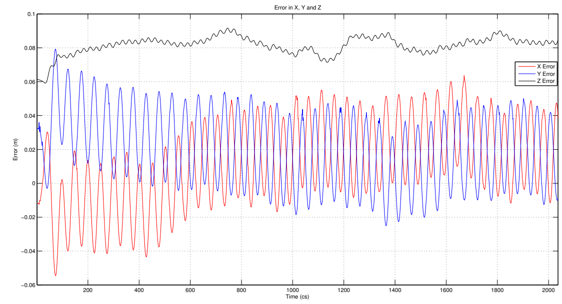

Fig. 15. Error for Waypoints 4.

The previously described algorithms were implemented on a KMel Nano+ quadrotor and the data was logged.

A. *Hover Experiment*

The quadrotor was commanded to go to the position indicated by the black star in the Fig. 2. The quadrotor was disturbed to see if it went back to the original position. A sequence of images showing disturbances applied to the quadrotor in hover state are shown in Fig. 3. The error plot ($e = r - r_{act}$) is shown in Fig. 5. Observing closely we see that there is always a small offset in Z direction, this maybe due to the value of K_p in Z direction being smaller than necessary or thrust output from propellers not being uniform.

B. *One Waypoint Experiment*

The quadrotor was commanded to go to the position [0 0 1.5] from its hover position. Here, the blue line corresponds to the desired trajectory and the red line corresponds to the actual trajectory. The recoded data is shown in Fig. 6. The error plot is shown in Fig. 7.

C. *Multi Waypoints Experiment*

The quadrotor was commanded to execute 4 different trajectories. The corresponding plots are shown in Figs. 8, 9, 10 and 11. The error for each of the trajectories is shown in Figs. 12, 13, 14 and 15.

VI. Conclusion

The linear control model for the quadrotor performed as expected for all trajectories, considering their smoothness and lack of extreme maneuvers. The use of a cubic spline for trajectory generation had an effective outcome for all single and milt-waypoint tests, and for the hover test, the disturbances manually applied to the quadrotor were useful calibration tools for the controller gains. Finally, the steady state errors for all navigation modes in the X, Y and Z directions were smaller than 0.07m indicating the linearization of the system dynamics did not affect the results considerably, as expected.

Acknowledgment

The authors would like to thank Dr. Vijay Kumar and all the Teaching Assistants of MEAM 620 course for all the help in accomplishing this project.

References

[1] Vijay Kumar, and Nathan Michael, *Opportunities and challenges with autonomous micro aerial vehicles*, The International Journal of Robotics Research, pp. 1279–1291, 2012.

[2] Samir Bouabdallah, *Design and control of quadrotors with application to autonomous flying*, Diss. cole Polytechnique federale de Lausanne, 2007.

[3] MEAM 620 instructors at Univeristy of Pennsylvania, *Dynamic Modeling, Control and Simulation of an Autonomous Quadrotor*, MEAM 620: Project 1 Phase 2, 2015.

[4] MEAM 620 team at Univeristy of Pennsylvania, *Path Planning (in 3D!)*, MEAM 620: Project 1 Phase 1, 2015.

[5] MEAM 620 instructors at Univeristy of Pennsylvania, *Trajectory Generation and Control of a Quadrotor*, MEAM 620: Project 1 Phase 3, 2015.

[6] MEAM 620 instructors at Univeristy of Pennsylvania, *MEAM 620 Project 1 Phase 4*, MEAM 620: Project 1 Phase 4, 2015.

[7] KMel Robotics, http://kmelrobotics.com/.

[8] MEAM 620 instructors at Univeristy of Pennsylvania, *MEAM 620 Lab Manual*, MEAM 620 Lab Manual, 2015.