GEOMETRIC POSITION CONTROL OF AN UAV

Rishab Shah

5/2/2017

Boston University

Department of Electrical and Computer Engineering

Technical Report No. ECE-

BOSTON UNIVERSITY

GEOMETRIC POSITION CONTROL OF AN UAV

Rishab Shah

Boston University
Department of Electrical and Computer Engineering
8 Saint Mary's Street
Boston, MA 02215
www.bu.edu/ece

5/2/2017

Technical Report No. ECE-

Summary

Umanned aerial vehicles (UAV) are increasingly being used in military and civilian domians, including surveillance operations, weather observation and disaster relief coordination. Unfortunately, in most structures which are often struck by disasters are broken down, which causes the search and rescue operations to be very dangerous under such situations. This allows for using these UAVs to maneuver tight spaces and perform data reconnaissance and provide valuable information to the rescue team about the environment.

This project aims to provide a new-age position tracking control of a quadrotor UAV that is potentially able to perform aggressive acrobatic manipulation of its movement. The control input for this system uses an underactuated control with the four rotor thrusts as the input that control the six translational and rotational degrees of freedom, and achieve exponential tracking of the four outputs for all angles. The analysis of the system presents a globally defined model of the rigid body dynamics. A non-linear position tracking controller is developed on the special Euclidean group SE(3) and the properties are shown to be global. These kind of controllers that are based on non-linear manifolds are called geometric controllers that exploit the geometric properties of the space and the rigid body. The geometric controller has been implemented as a simulation on MATLAB which shows global exponential stability of the position tracking by performing a simple maneuver that illustrates the versatility of the controller.

The UAV used in this project is the Parrot Rolling Spider Mini Drone which has an open source firmware and can be used to customize and run any kinds of controllers with additional capabilities of incorporating the camera into the control of the UAV. An open source toobox from MIT is used here to design the controller. The toolbox already consists of a well-established connectivity software with ROS and MATLAB which are used to provide the control program to the UAV and simpler controllers such as the PID control, full state feedback control and LQR control have been tested successfully with verifiable results.

Contents

List of Figures

1 Introduction

Consider the partially collapsed building in figure 1. Sending rescue personnel into the building to search for survivors puts them in grave danger. Without knowing what awaits them inside the building, it is very difficult to make good decisions about where it is safe to venture and where to look for survivors. If instead, the building could be searched by a robot, the risks taken by the rescue workers would be greatly diminished. Indeed, there are many situations where it is dangerous and difficult for humans to acquire sensing information and where robots could be of use. While the utility of robots performing such sensing tasks may be obvious, creating the robots is certainly not. Operating within a partially collapsed building, or other similar environments requires a robot to be able to traverse cluttered, obstacle strewn terrain. Over the years, researchers have tackled these problems and designed a number of ground robot systems capable of traversing through rough terrain. Despite these advances, it is still an active area of reserach. Many researchers have therefore proposed the use of Unmanned Aerial Vehicles (UAVs) as an alternative robotic platform for rescue tasks and a host of other applications, with recent advances in the planning, control and design of the quadrotor UAVs.

Figure 1: A partially collapsed building after an earthquake. J.K. Nakata, U.S. Geological Survey

A quadrotor UAV consists of two pairs of counter-rotating rotors and propellers which are positioned at the ends of its square body frame. It can perform vertical take-off and landing [2] without any complex mechanical links that are commonly present in a helicopter [1]. Due to the reduction of mechanical complexity and wear, it is expected that well-designed quadrotors will prove to be inherently more robust and reliable. But, in practical applications, it is best to maximize their dynamic performance and aerodynamic capabilities. Linear control systems such as proportional-derivative controllers or linear quadratic controllers are widely used to improve the stability properties of an equilibrium.

The non-linear controllers that are used, such as the sliding mode control or the backstepping approach, use Euler angles in their implementation, they exhibit singularities when representing complex maneuvers of the UAV which restricts the tracking of non-trivial trajectories.

1.1 Geometric Control

According to Velimir Jurdjevic, "Geometric control theory provides the calculus of variations new perspectives that both unify its classic theory and outline new horizons toward which its theory extends. It forms a theoretical foundation for extensions of the maximum principle to optimal problems on arbitrary differentiable manifolds and contains important results concerning the topological and differential properties of the reachable sets." [3] Also, geometric control theory is a subject that is deeply involved with the structural properties of the Lie algebras and their differentiable manifolds. A Lie group is a smooth manifold which also carries a group structure whose product and inversion operations are smooth as maps of manifolds.

We assume that the states of the system can be placed into a one-to-one correspondence with the points of an abstract n-dimensional manifold. The manifold models the state space, while the dynamical system models the dynamics of the physical system. Physical quantities such as angular velocity, etc., provide local coordinates on the manifold. These local coordinates can be used to locally depict the dynamic system on the manifold as a set of differential equations defined on an open subset of $\mathbb{R}^n$, even though the manifold itself might be globally quite different from the open subset $\mathbb{R}^n$. While local coordinates are opportune for analyzing the local properties of a dynamic system, questions of a global nature require global analysis for adequate results.

In this project, I implement a geometric controller for a quadrotor UAV. The dynamics of a quadrotor UAV can be expressed globally on the configuration manifold of the special Euclidean group SE(3). SE(3) is defined as a non-commutative product of 3D rotations and 3D translations, SE(3):=SO(3)$\triangleright$ $\mathbb{R}^3$. This geometric controller, contrary to other similar approaches, uses an underactuated control input with the four motor thrusts to stabilize six rotational and six translational degrees of freedom. This method completely avoids any singularities, i.e., a point at which a given mathematical object is not defined or not "well-behaved", for example infinite or not differentiable, that arise when using local coordinates.

2 Quadrotor model

Consider a quadrotor model as depicted in Figure 2. This is a system of four identical rotors located at the ends of a square frame, which generate a torque and thrust normal to the plane of this square.

Let's choose an inertial reference frame $\{\vec{i}_1, \vec{i}_2, \vec{i}_3\}$ and a fixed body frame $\{\vec{b}_1, \vec{b}_2, \vec{b}_3\}$.The origin of the fixed body frame is located at the center of mass of this quadrotor.

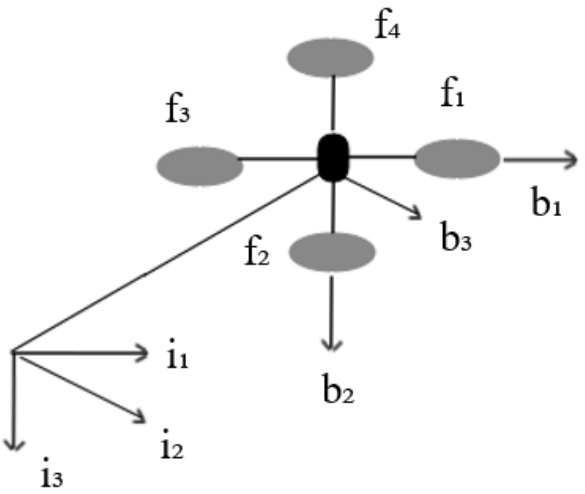

Figure 2: Quadrotor model

$m \in \mathbb{R}$ the total mass
$J \in \mathbb{R}^{3x3}$ the inertia with respect to fixed body frame
$R \in SO(3)$ rotation matrix from body frame to inertial frame
$x \in \mathbb{R}^3$ the location of the center of mass in the inertial frame
$f_i \in \mathbb{R}$ the thrust generated by the i-th propeller along $-\vec{b}_3$ axis
$f \in \mathbb{R}$ total thrust
The configuration of this UAV is defined by the location of the center of mass and the attitude with respect to the inertial frame.Therefore, the manifold in question is the special Euclidean group SE(3).

2.1 Assumptions

We assume that the torque generated by each propeller is directly proportional to its thrust. We also assume that the first and the third propellers rotate clockwise and the second and the fourth propellers rotate counter-clockwise, this represents a positive thrust. Additionally, the thrust of each propeller is assumed to be directly controlled, and the direction of thrust is normal to the plane. For the simulation of the quadrotor, we assume a point mass system and ignore the friction forces acting on the system.

3 Geometric controller

Lee et. al. [4] defines the equations of motion on the manifold as

$$\dot{x} = v \tag{1}$$

$$m\dot{v} = mge_3 - fRe_3 \tag{2}$$

$$\dot{R} = R\hat{\omega} \tag{3}$$

$$J\dot{\omega} + \omega \times J\omega = M \tag{4}$$

To develop the controllers, a cost function

$$\Psi(R, R_d) = \frac{1}{2} trace[\mathbf{I} - R_d^T R] \tag{5}$$

was chosen such that appropriate error functions could be found for the attitude, position and velocity. But, the choice of the error function in equation 5 resulted in a position controller that was only exponentially stable for initial atitude error less than 90°. We choose a cost function such that it has exponential convergence everywhere.

$$\Phi(R_1, R_2) = \frac{1}{2} dist^2(R_1, R2) \tag{6}$$

The resulting attitude error e_R then has the form,

$$e_R = \nabla(\Phi(R_1, R_2)) = dist(R_1, R_2) \tag{7}$$

By choosing the cost function in the form of equation 7, we will also improve the exponential stability region of the position controller as well. We start by using a rigid point mass in $\mathbb{R}^3$ which has exponential stability, the state equation for this system is given by

$$\begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & \mathbf{I} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix} + \frac{1}{m} \begin{bmatrix} 0 \\ \mathbf{I} \end{bmatrix} \mathbf{F} \tag{8}$$

$$\mathbf{F} = [k_d \nabla\rho(x_d - x) + k_v(v_d - v)]m \tag{9}$$

where k_d and k_v are strictly positive scalar gains, and $\nabla\rho$ can be any convex cost function. In this example, it is given by

$$\rho(x) = 2\left(\sqrt{1 + \frac{\|x\|^2}{2}} - 1\right) \tag{10}$$

Then we use the fact that the transient of our system is fast enough that we can model our UAV as a point mass, thus we can infer that the geometric controller is exponenetially stable.

4 Practical Tests

The implementation for this project was divided into two phases that were worked out simultaneously. These were, working on the Parrot Rolling Spider drone to try various simpler controllers such as the PID control, Full state feedback and the LQR control, while working the geometric controller and its simulation itself in parallel. The Parrot Rolling Spider is used to perform tests to control the dynamic system. It has the following specifications

Mass	68g
Motors	33g thrust/motor
IMU	6-axis-accelerometer-gyroscope
Altitude sensors	Sonar , Pressure sensor
Vision	Downward facing camera, 160x120
Misc.	Bluetooth Low Energy 4.0
Battery	7-8 min flight time

The reason for using this particular drone was that it is easy to carry/fly, it has safety protective wheels that makes it reliable for testing purposes because it is more prone to crashes and because of the bluetooth connection, it is easy to quickly program and download flight data immediately after the experiment. Also, the firmware is opensource and available to anyone to customize which is one of the reasons why this model is so popular among educators and students.

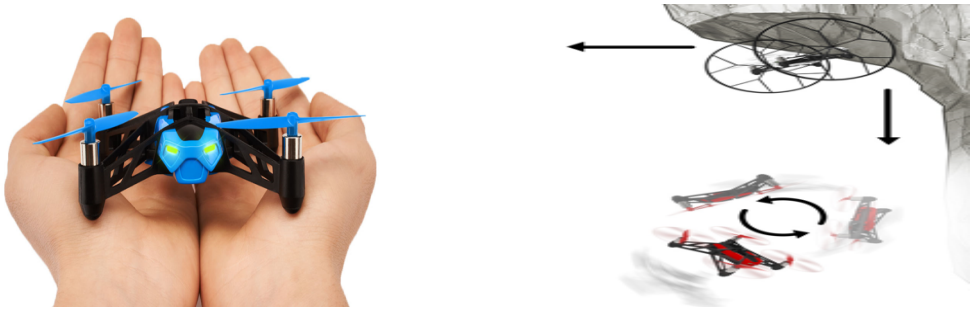

Figure 3: The Parrot Rolling Spider
http://global.parrot.com/au/products/rolling-spider/

The workflow of the Parrot's custom firmware is as follows. At each control loop stage, Parrot code calls an externally compiled library. The user can then write the control code in C and compile the code into a library, which will be called by the Parrot custom firmware at each control loop stage. Parrot also provides the compiler toolchain and the custom firmware which opens up the three interfaces to insert custom controls code, image processing code and optical flow handling. This process can

easily be made use of to write controllers for the UAV.

4.1 How the MIT toolbox works

The toolbox [5] written by MIT professor Sertac Karaman and Fabian Riether. A MATLAB/ Simulink toolbox was constructed that can simulate control systems and auto-generate the control loop C code from Simulink block diagrams. They also constructed tools that easily compile the auto-generated code, upload the compiled code to the drone via Bluetooth, start/execute an experiment via Bluetooth and download the experiment data via Bluetooth. There is only a signle flashing of the custom firmware required, when using it for the the first time, where I use the USB the cable to flash the custom firmware, after which all the interaction is done via Bluetooth.

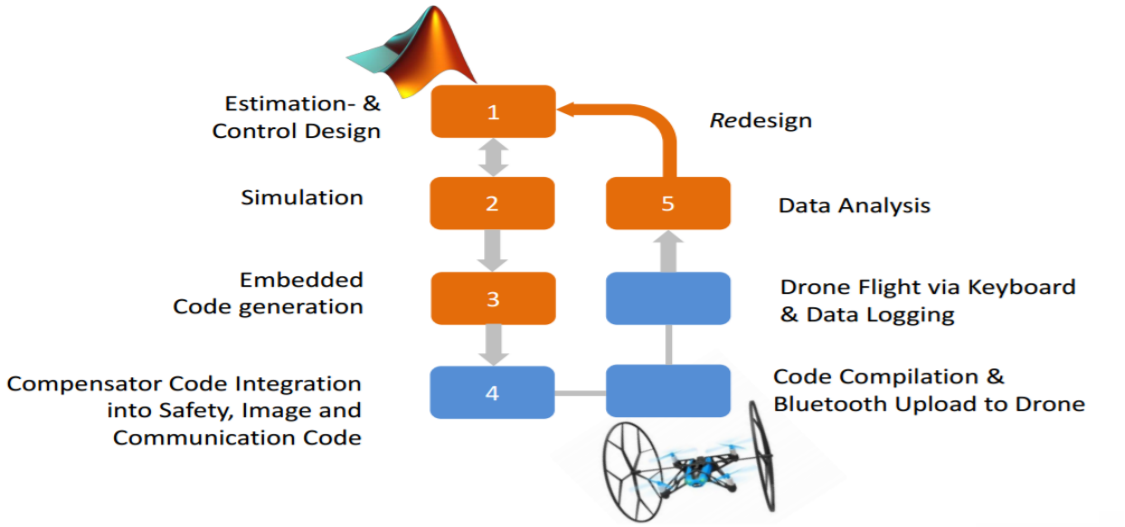

Figure 4: Workflow of the MIT Toolbox, https://github.com/Parrot-Developers /RollingSpiderEdu/blob/master/MIT_MatlabToolbox/media/GettingStarted.pdf

4.2 Controllers

These systems use the optical flow for their pose and state estimtaion in addition to using the Sonar and Pressure sensors, all of which are very noisy. The toolbox has its standard output set such that the drone hovers upto a height of 1.1m and just stays there until the end of its flight time. The flight time is set to 20 seconds to reduce battery consumption.

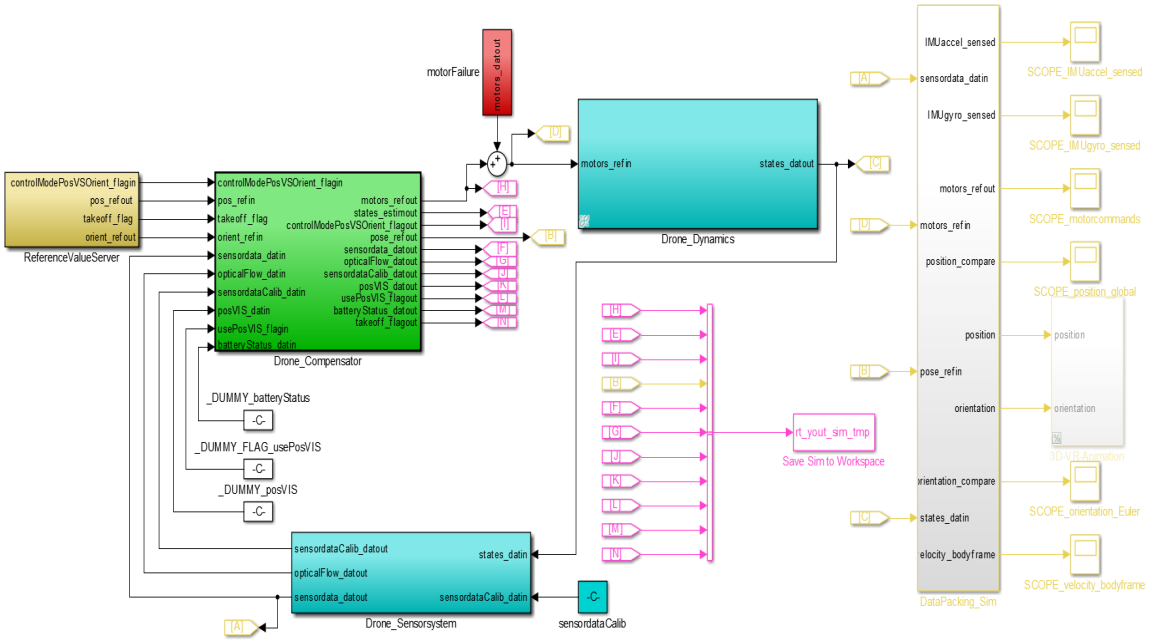

Figure 5: Simulink block, https://github.com/Parrot-Developers /RollingSpiderEdu/blob/master/MIT_MatlabToolbox/media/GettingStarted.pdf

4.2.1 PID control

A PID controller is a linear controller, that calculates an error value continuously, as the difference between a desired point and a measured process varible and applies a correction based on proportional, integral and derivative terms. The proportional gain controls the present values of the error. The integral gain accounts for the past values of the error and the derivative term accounts for the possible future trends of the error, based on its current rate of change. The problem is that this does not provide optimal control. The PID controller was designed with gains as shown in the Simulink block for the controller and results that were obtained were clearly observed by the graph plots of the states, and various sensor mesaurements.
The first graph on the top left shows the sensed changes in the IMU sensors, i.e, the Sonar and the Pressure sensors, the next graph shows the commands to the motor that are in turn converted to motor thrusts. Finally, the last graph shows the global position of the UAV where it is clearly seen of the PID dynamics, which are the overshoot and stabilization in the position in the Z-direction. The graph is such that increase in the Z-direction is seen as a negative change.

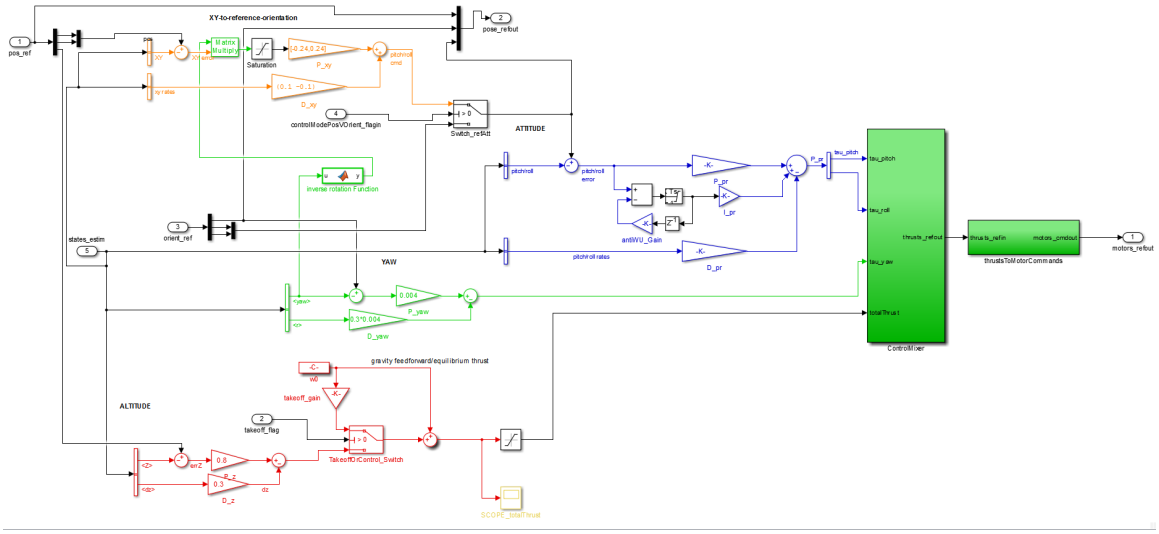

Figure 6: PID controller

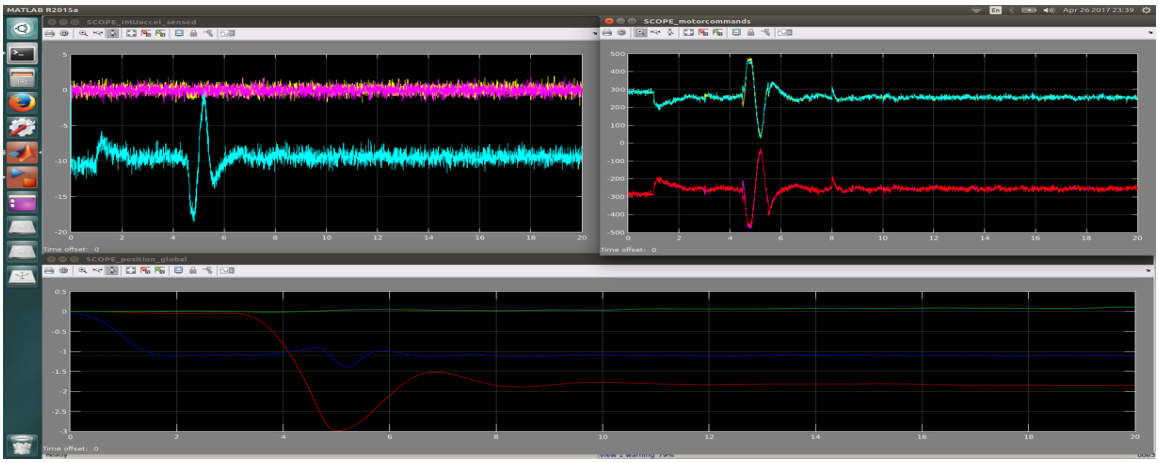

Figure 7: PID controller

4.2.2 Full State Feeback

Full state feedback, or pole placement, is a method employed in feedback control system theory to place the closed-loop poles of a dynamic system in pre-determined locations in the S-plane. I used a linearized system to perform pole placement that was given by the MIT toolbox and designed my own controller based on their linearization method.

This is a result obtained by using poles that were placed in the negative half plane that allowed the UAV to have fast dynamics, as seen by the multiple oscillations in the position graph. This is still a linearized system hence does not accurately take into account all quantities of the non-linear system and hence is only locally asymptotically stable.

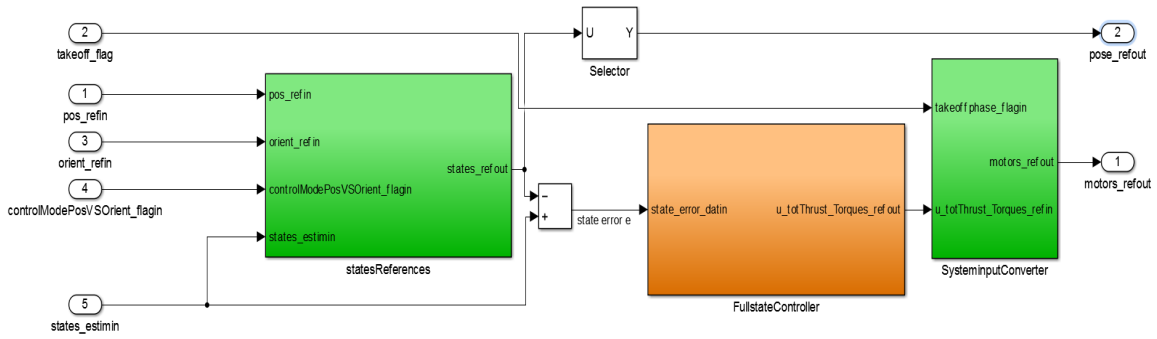

Figure 8: Pole placemement controller

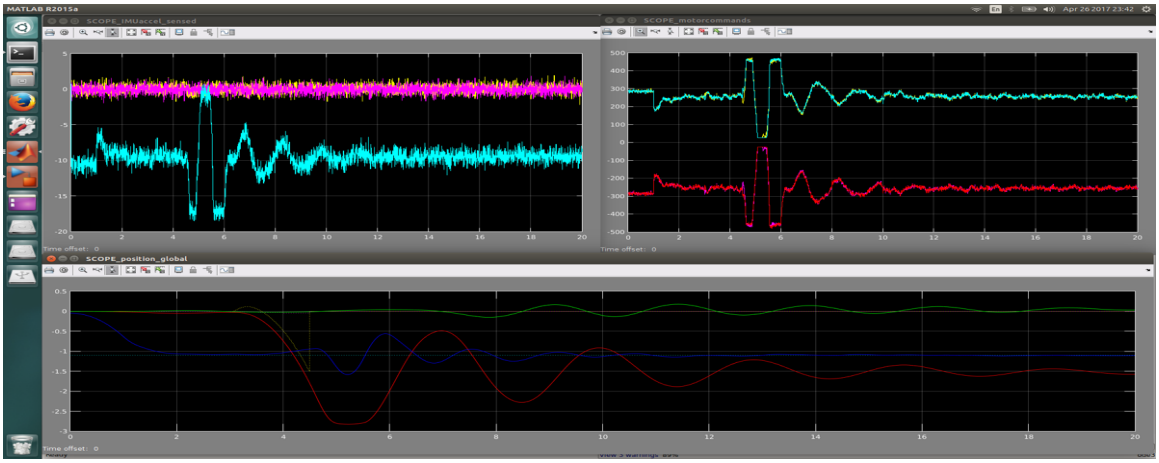

Figure 9: Pole placemement results

4.2.3 LQR controller

Optimal controllers, i.e., controllers that "best" possible, according to some figure of merit, turn out to generate only stabilizing controllers for multiple input multiple output plants. In this sense, optimal control solutions provide an automated design procesure, the linear quadratic regulator (LQR) is a well-known design technique that provides practical feedback gains. Pole placement techniques are again used to design the controllers and the Riccati equation is used to compute the optimal feedback gain that can be easily solved by standard numerical tools in linear algebra.
The reusults show that the stabilty of the position of the UAV is known to be much more smooth and stable with minimal overshoot.

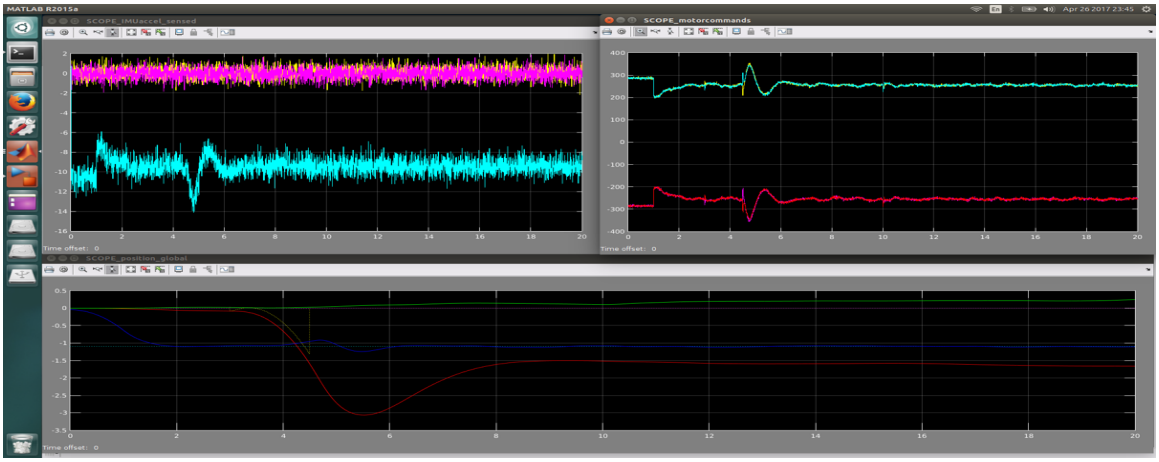

Figure 10: LQR controller results

4.3 Simulation of the geometric controller

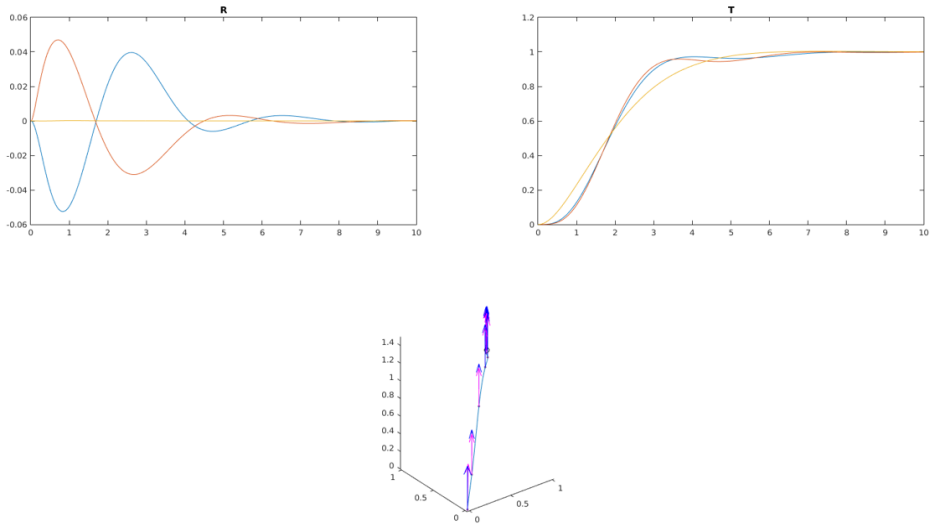

Figure 11: Orientation, Position and the trajectory of the UAV using geometric control

This simulation is considering a point mass that just linearly moves in the X, Y and Z-direction. Friction has not been taken into account for this simulation. The Rotation graph shows how the angles are changing in the space. The y-axis of these plots have been normalized to 0-1 range. The units for the orientation are in radians and for the translation are meters.

4.4 Challenges

One of the most major challenges that I faced in implementing the controllers on the Parrot Rolling Spider was to not include optical flow and image processing techniques for preliminary state estimation. This code is embedded into the simulink block for state estimation that uses kalman filter to filter the sensed signals and use them to provide an estimate of the current states. The optical flow from the camera actually looks at the change in pixel values in consecutive frames and tries to stabilize its position. This, for our purpose, is not necessarily needed at this point. Also there are no methods to view each frame of the video from the UAV to check how well it is performing as transferring video data over bluetooth is not possible and the storage available on the UAV itself does not allow for videos to be stored.
The battery of the UAV also lasted only for 7-8 minutes which was very limiting to actually perform multiple trials at once, the maximum it allowed was about 5-6 trials before I had to charge the battery, which took about an hour to charge fully.

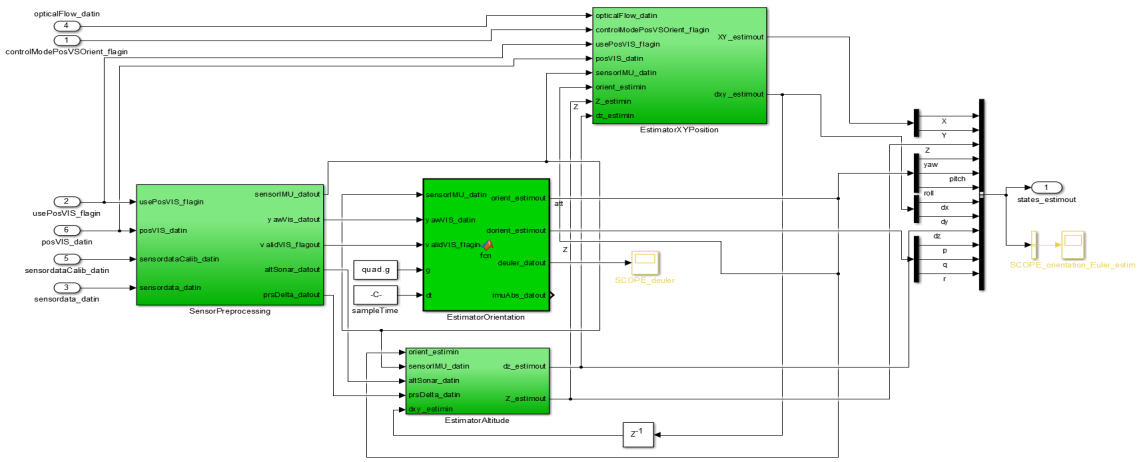

Figure 12: State estimation using the physical parameters

5 Conclusion

In conclusion, I implemented a geometric position controller of an UAV that provides global exponential stability for all attitude tracking errors. I also tested linearized controllers on the Parrot Rolling Spider UAV, using the help of a MIT toolbox that provides the user to control the Parrot and use any controller.
In the future, I would like to implement the geometric controller on the Parrot drone itself, and also perofrm more complex maneuvers with it. I would work work on getting exponential stability for both position and attitude tracking errors for the controller and testing this on the Parrot drone.

References

[1] G. Hoffmann, H. Huang, S. Waslander, and C. Tomlin, "Quadrotor helicopter flight dynamics and control: Theory and experiment, in *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, 2007, AIAA 2007-6461.

[2] S. Bouabdalla, P. Murrieri, and R. Siegward, "Towards autonomous indoor micro VTOL, *Autonomous Robots*, vol. 18, no. 2, pp. 171183, 2005.

[3] V. Jurjevic, "Geometric Control Theory", Cambridge Univeristy, 1997.

[4] T. Lee and M. Leoky and N. H. McClamroch, "Geometric tracking control of a quadrotor UAV on SE(3)", *49th IEEE Conference on Decision and Control (CDC)*, 2010, pages 5420-5425.

[5] F. Riether, S. Keraman, "ROSMAT-Rolling Spider MATLAB Toolbox", https://github.com/Parrot-Developers/RollingSpiderEdu