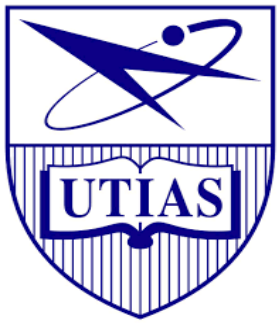

University of Toronto, Institute for Aerospace Studies

AER1217: Development of UAVs

Lab 1 Report:

Quadrotor Simulation and Control Design

Team: Try Guys

Akshata Puranik 1006225540

Igor Monteiro 1005577644

Yushen Zhou 1002666732

1 Introduction

The purpose is to design a position controller for the quadrotor: Parrot AR Drone, implement it on ROS (Robot Operating System) and simulate the work on Gazebo for a linear and a circular trajectory for a given set of coordinates using a VICON motion capture system for position and altitude measurements. Additionally, generate the graphs presenting the desired and the actual path taken by quadrotor to fly from one point to another along all the axes and yaw angle. Finally, show the value of error changes with time.

2 Controller Setup

The details of file structure and communication established between them can be understood from the diagram below:

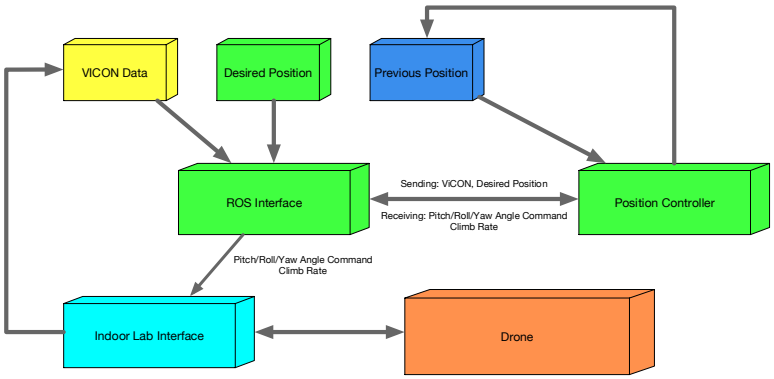

Figure 1. Overall Control Architecture

2.1 Position Controller

After considering all the applicable forces on a quadrotor, we get the following equations for acceleration in all the three directions.

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = R(\phi, \theta, \psi) \begin{bmatrix} 0 \\ 0 \\ f \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}$$

Here, R is rotation matrix, f is mass-normalized thrust, g is the gravitational force, θ is roll angle, ϕ is pitch angle and ψ is yaw angle. On simplifying the above equation and substituting $\psi = 0$, we get:

$$\ddot{x} = f \cos\phi \sin\theta$$
$$\ddot{y} = -f \sin\phi$$
$$\ddot{z} = f \cos\theta - g$$

On rearranging, we have the final equations for roll angle (θ), pitch angle (ϕ) and f as:

$$\phi = sin^{-1}\frac{\ddot{y}}{f}; \quad \theta = sin^{-1}\frac{\ddot{x}}{f\cos\phi}; \quad f = \frac{\ddot{z}+g}{\cos\theta\cos\phi}$$

Having received all the desired and current position values and calculating f, roll and pitch angles using the above equations, we find the values of acceleration in x and y directions using:

$$\ddot{x_c} = 2\zeta_x \omega_{n,x}(\dot{x_d} - \dot{x}) + \omega_{n,x}^2(x_d - x)$$

$$\ddot{y_c} = 2\zeta_y \omega_{n,y}(\dot{y_d} - \dot{y}) + \omega_{n,y}^2(y_d - y)$$

where, $\dot{x}_d - x$ and $\dot{y}_d - y$ is error in reference velocity, $x_d - x$ and $y_d - y$ is error in position in x and y direction respectively.

Followed by determining the commanded roll (θ_c) and pitch (ϕ_c) at $\psi = 0$ using the equation of ϕ and θ mentioned above. Lastly, the values of commanded climb rate and yaw rate are evaluated by the relations given as under:

$$\dot{z_c} = \frac{1}{\tau_z}(z_d - z); \qquad \dot{\psi_c} = \frac{1}{\tau_\psi}(\psi_d - \psi)$$

The designed position controller is called as a separate function instead of an active node in this architecture. Every the drone position adjustment is required, the ros_interface.py will call this function and pass over the VICON data and desired position data to calculate required pitch/roll/yaw angle command as well as the z-direction acceleration.

The position controller have a pair of tuning parameter (Damping Ratio - ζ; Natural Frequency - ω_n) on all three x,y,z directions. The drone performance is tested and simulated under Gazebo by comparing the results and tuning those parameters. Both final tuning parameters and performance results will be in the next section.

2.2 Desired Position Controller

The desired position controller publishes the two trajectories coordinates that drone are expected to follow. The script is designed to given the desired trajectories based on the user command (Linear or Circular).

3 Controller Performance

3.1 Tuning Parameters

X-direction: $\zeta = 0.6$; $\omega_n = 0.8$

Y-direction: $\zeta = 0.9$; $\omega_n = 0.8$

Z-direction: $\zeta = 0.7$; $\omega_n = 0.95$

The tuning process is based on second-order system to reduce the percent of overshoot and the response time of the drone to react to the given command. Note that the drone is an non-linear system, therefore tuning based on second-order system relation does not necessarily affect the drone performance as expected. However, some fundamental control system characteristic does apply.

The output of this experiment can be summarized in the graphs below. The first set of graphs simultaneously represents the desired position values and actual position values of quadrotor at a given instance of time in x, y, and z direction. While the second set signifies the difference between these values or error with respect to time in x, y, z directions and yaw angle.

(*Note: The Errors were not calculated accurately as corresponding timestamps weren't used)

3.2 Linear Trajectory

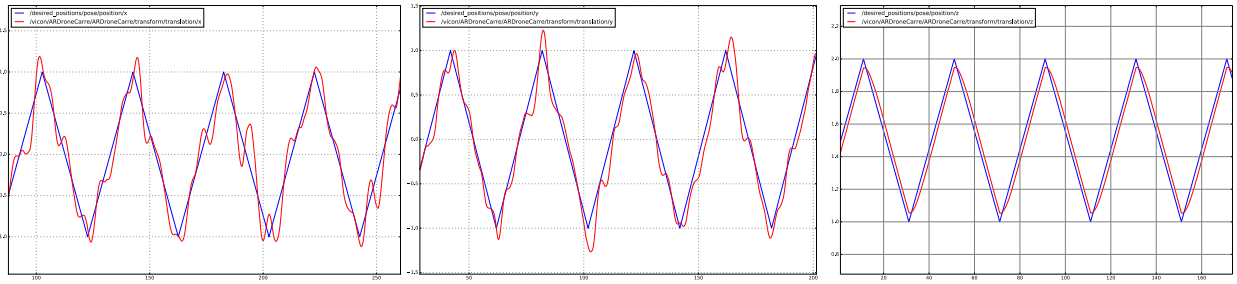

Figure 2.Linear - Actual Position vs Desired Position (x-left; y-middle; z-right)

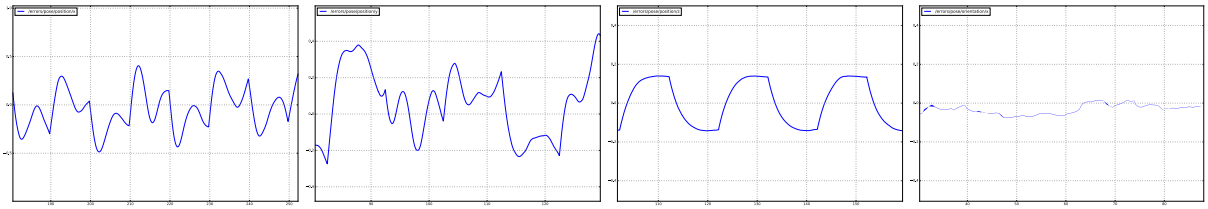

Figure 3. Linear - Error in x,y,z Position and Yaw (x-1st; y-2nd ; z-3rd; yaw-4th)

3.3 Spiral Trajectory

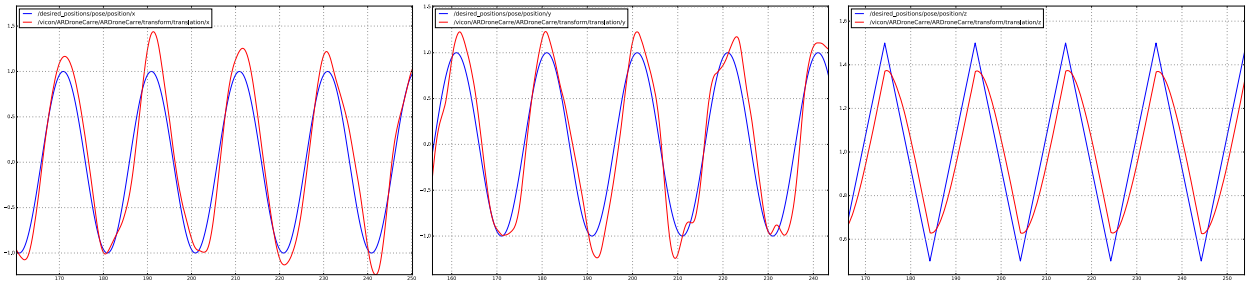

Figure 4. Spiral - Actual Position vs Desired Position (x-left; y-middle; z-right)

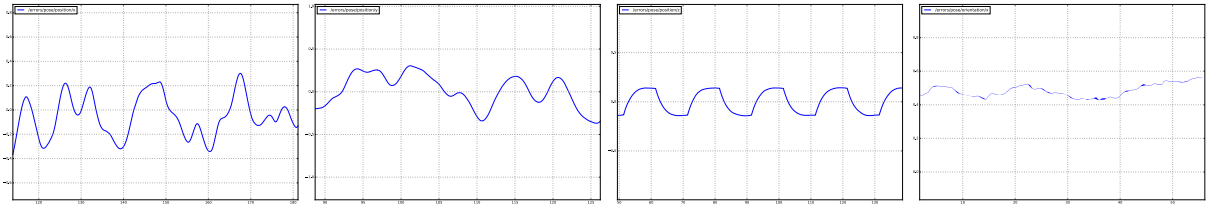

Figure 5. Spiral - Error in x,y,z Position and Yaw (x-1st; y-2nd; z-3rd; yaw-4th)