Aerial Robotic Controls Task

Soumick Pyne , 18AE10027.

***Abstract*— This paper briefly sums up the work I have done as a part of the controls task assigned to me by Aerial Robotics Kharagpur.**

The control system designed for a quadcopter can be used for stable and accurate control of quadcopter for purposes like video shooting of inaccessible places, package delivery etc.

I. INTRODUCTION

The task basically involved developing an accurate simulation model of a quadcopter on matlab/simulink and then designing an altitude controller for it. The mathematical modelling was done based on the Euler Lagrange formalism and the altitude controller evolved from a proportional controller to a PID controller.

II. PROBLEM STATEMENT

The task broadly consisted mainly of three parts. First, to understand the dynamics and controllable parameters of a quadcopter.This involves deciding co-ordinate system and analysing the degrees of freedom of the object with respect to the co-ordinate system. After this we have to identify the actuators that can be controlled to give the desired motion to the object.

Second, to develop an accurate simulation model of the quadcopter on simulink. This requires writing physical equations governing the dynamics and modelling the actuators of the object.

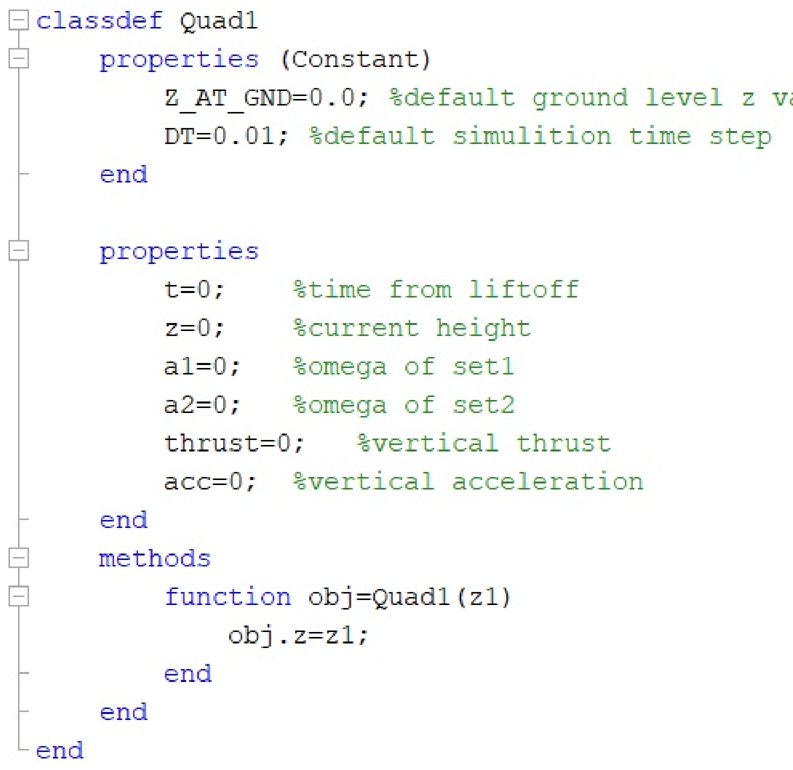
Third, to design an altitude controller for the quadcopter that keeps the quadcopter at desired height from the ground which is taken as input from the user.The objective is to stabilise the quadcopter in the least time possible in spite of noise.

III. RELATED WORK

There are several simulation models developed for quadcopter control based on Laupnov theory, adaptive optimal control theory and other theories. Links to these projects are in the references.

IV. INITIAL ATTEMPTS

The initial approach was an object oriented approach where I defined a class called Quad with all its associated properties and member functions to simulate its behaviour.An excerpt of the code is given below.



```
classdef Quad1
    properties (Constant)
        Z_AT_GND=0.0; %default ground level z va
        DT=0.01; %default simulition time step
    end

    properties
        t=0;    %time from liftoff
        z=0;    %current height
        a1=0;   %omega of set1
        a2=0;   %omega of set2
        thrust=0;   %vertical thrust
        acc=0;  %vertical acceleration
    end
    methods
        function obj=Quad1(z1)
            obj.z=z1;
        end
    end
end
```

This approach was however discarded because I found it cumbersome and impractical to decompose it to separate blocks of a control system.

The basic factors affecting dynamics were taken as below.

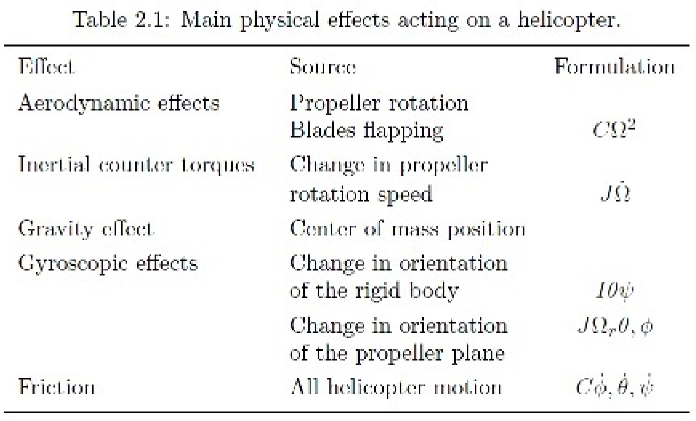
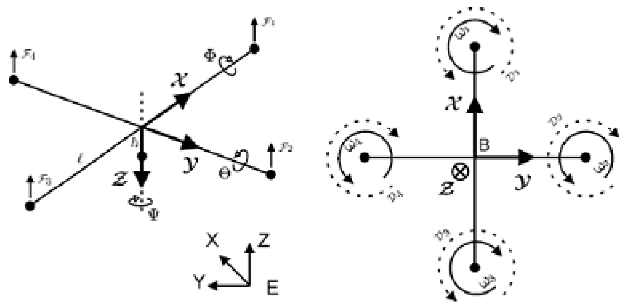


Table 2.1: Main physical effects acting on a helicopter.

Effect	Source	Formulation
Aerodynamic effects	Propeller rotation Blades flapping	$C\Omega^2$
Inertial counter torques	Change in propeller rotation speed	$J\dot{\Omega}$
Gravity effect	Center of mass position	
Gyroscopic effects	Change in orientation of the rigid body	$I\theta\psi$
	Change in orientation of the propeller plane	$J\Omega_r\theta, \phi$
Friction	All helicopter motion	$C\dot{\phi}, \dot{\theta}, \dot{\psi}$

The co-ordinate system was chosen as follows.

Thanks to Aditi Singh for guiding me.

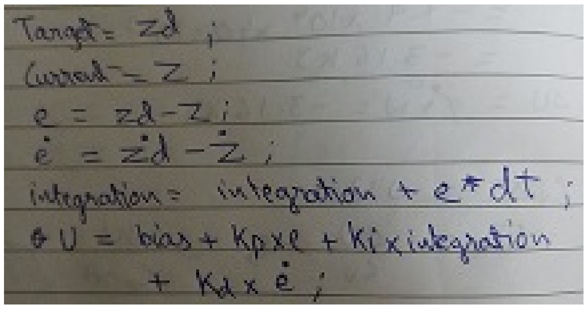


V. Final Approach

”aero.m” is the block for computing all the torques, hub forces, roll moments etc on the quadcopter. The block takes as input 12 system states and 4 actuator states, computes parameters like inflow ratio, advance ratio etc. which are used to calcuate the outputs.The formulas used were referred from Gary Fay’s paper on blade theory.

”dynamica.m” is the block that takes as input, the forces and torques on system and gives as output the system accelerations.

”altcontrol” is the altitude controller of the system which takes as input the present altitude, destination altitude, rate at with which height is changing, rate with which destination altitude is changing and thus uses a PID equation to calculate the total thrust required. The pseudo code goes:

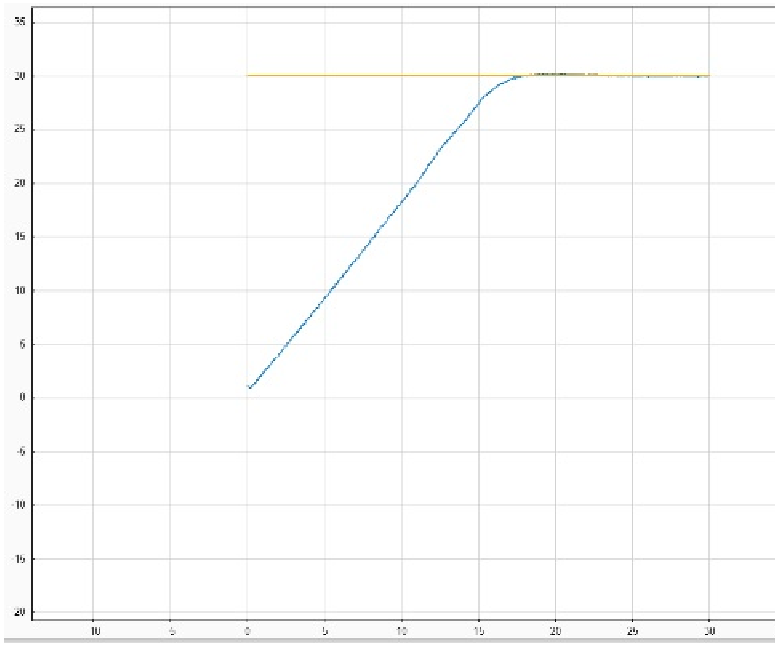


```
Target = zd ;
Current = z ;
e = zd - z ;
ė = żd - ż ;
integration = integration + e*dt ;
U = bias + Kp×e + Ki×integration
        + Kd×ė ;
```

VI. Results and Observation

The PID was tuned using ZieglerNichols method(ku=6.3,Pu=6) first but it generated an offset between the final constant value and desired value, so the values were further fine tuned from there.

The tuned response for final altitude of 30 meteres was as below: (kp=3.78,ki=0.01,kd=3.5)



VII. Future Work

The next stage will be a more robust controller that can tolerate increased noise and a model that works for highly turbulent air conditions.

CONCLUSION

The main problem with tuning the controller was the output cap because of which all output values less then one are rounded to one and all values greater that 48 are rounded to 48. This imposed a constrained on the time required to attain the desired value and also on the time for the oscillations to decay.

A lot of improvements at the fundamental level are still possible to make the system more accurate and adaptable and I look forward to do that myself in the coming future.

References

[1] G. Fay, Derivation of the aerodynamic forces for the mesicopter simulation, 2001.
[2] S. Bouabdallah et al., Design and control of an indoor micro quadrotor, in Proc. (IEEE) International Conference on Robotics and Automation (ICRA04), (New Orleans, USA), 2004.
[3] http://robotsforroboticists.com/pid-control/
[4] https://www.thorlabs.com/tutorials.cfm?tabID=5dfca308-d07e-46c9-baa0-4defc5c40c3e
[5] Eswarmurthi Gopalakrishnan,”QUADCOPTER FLIGHT MECHANICS MODEL AND CONTROL ALGORITHMS”