Design and Implementation of Control algorithms for Quadcopter

Prasham Patel
Robotcs Enginering Department
Worcester Polytechnic Institute
Worcester, USA
pspatel@wpi.edu

Purna Patel
Robotcs Enginering Department
Worcester Polytechnic Institute
Worcester, USA
prpatel@wpi.edu

***Abstract*— Design of feedback linearized control and adaptive control for a quadcopter with uncertain dynamics has been discussed in this paper. The dynamics of the quadcopter in Manipulator form as well parametric form are considered. The testing is performed in Gazebo as well as in ODE45 based simulation in MATLAB. A wind force estimator controller has also been discussed in the end. And finally the results of the testing along with graphs are presented.**

Keywords—quadcopter, dynamic modelling, feedback linearization, adaptive control, gazebo simulation

I. INTRODUCTION

Unmanned Aerial Vehicles (UAVs) have gained popularity in many sectors like military surveillance and transportation. And quadcopter among the UAVs is the most preferred for low altitude flights missions. Such missions often require reliable trajectory tracking controller. However, due to the non-linear coupled dynamics of the quadcopter makes it much more difficult, especially when the dynamics are uncertain.

To address the same we perform a gazebo simulation for trajectory tracking task with Feedback Linearized Controller. And same trajectory tracking task is then performed in a ODE45 based MATLAB simulation using an Adaptive Inverse Dynamics control. In section II, dynamic model of the quadcopter in manipulator and parametric form are discussed. Using this models, feedback linearized control law for 3D quadcopter control is derived in section III. Section IV discussed the derivation of adaption law for adaptive control.

These derived controller were then applied to the simulation in MATLAB and Gazebo environments. In section V, the results of simulation in MATLAB are discussed. Section VI discusses the results and problems faced during Gazebo simulation. Let's start with the dynamic modelling of the Quadcopter.

II. DYNAMIC MODEL

Quadcopter is a underactuated system as it is has 6 degrees of freedom and only 4 actuators. Their is no independent control on two degrees of freedom. In this project, the objective is to control the position in three dimensions and the yaw orientation according to the defined trajectory. The actuators (i.e. the four motors) on the quadcopter controls the attitude and the total thrust and thus they have no direct control the motion in x and y direction. Thus the controller must be able to control the pitch and yaw position of the quadcopter accordingly by taking the input of desired position in x-y direction from the trajectory.

Controller pseudo layout:

- Trajectory provides the desired x, y, z and yaw angle (orientation).
- Desired roll and pitch angle are calculated at every iteration.
- Controller controls the z position, yaw , roll and pitch.

So according to our pseudo controller, the desired output of the controller must be total thrust and moments in three axis. Following part discuss the derivation of the dynamics which satisfies the controller pseudo layout.

A. Manipulator form

Deriving the dynamic model using Eular-Lagrange method.

- Kinetic energy:

$$K = \frac{1}{2}m\dot{x}^2 + \frac{1}{2}m\dot{y}^2 + \frac{1}{2}m\dot{z}^2 + \frac{1}{2}I{\omega_x}^2 + \frac{1}{2}I{\omega_y}^2 + \frac{1}{2}I{\omega_z}^2$$

- Potential Energy:

$$P = mgz + mgrcos\theta + mgrcos\emptyset$$

- L = K – P
- Calculating forces and moments in z, roll. Pitch and yaw direction.

- $\frac{\partial L}{\partial z} = -mg \qquad \frac{\partial L}{\partial \dot{z}} = m\dot{z}$
  $\therefore m\ddot{z} + mg = Fcos\theta cos\emptyset$

- $\frac{\partial L}{\partial \emptyset} = mgrsin\emptyset \qquad \frac{\partial L}{\partial \dot{\emptyset}} = I\dot{\emptyset}$
  $\therefore I_\phi\ddot{\emptyset} - mgrsin\emptyset = \tau_\emptyset$

- $\frac{\partial L}{\partial \theta} = mgrsin\theta \qquad \frac{\partial L}{\partial \dot{\theta}} = I\dot{\theta}$
  $\therefore I_\theta\ddot{\theta} - mgrsin\theta = \tau_\theta$

- $\frac{\partial L}{\partial \psi} = 0 \qquad \frac{\partial L}{\partial \dot{\psi}} = I\dot{\psi}$
  $\therefore I_\psi\ddot{\psi} = \tau_\psi$

- Writing the equations in matrix form.

$$\begin{bmatrix} m & 0 & 0 & 0 \\ 0 & I_\phi & 0 & 0 \\ 0 & 0 & I_\theta & 0 \\ 0 & 0 & 0 & I_\psi \end{bmatrix}\begin{bmatrix} \ddot{z} \\ \ddot{\emptyset} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} + \begin{bmatrix} mg \\ -mgrsin\emptyset \\ -mgrsin\theta \\ 0 \end{bmatrix} = \begin{bmatrix} cos\theta cos\emptyset & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} F \\ \tau_\emptyset \\ \tau_\theta \\ \tau_\psi \end{bmatrix}$$

- Transforming the equation in the manipulator form (multiplying the inverse of the 4x4 matrix on the RHS to the equation).

$$\begin{bmatrix} \frac{m}{cos\theta cos\emptyset} & 0 & 0 & 0 \\ 0 & I_\phi & 0 & 0 \\ 0 & 0 & I_\theta & 0 \\ 0 & 0 & 0 & I_\psi \end{bmatrix}\begin{bmatrix} \ddot{z} \\ \ddot{\emptyset} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} + \begin{bmatrix} mg \\ -mgrsin\emptyset \\ -mgrsin\theta \\ 0 \end{bmatrix} = \begin{bmatrix} F \\ \tau_\emptyset \\ \tau_\theta \\ \tau_\psi \end{bmatrix} \quad (1)$$

- Comparing the equation with the standard manipulator form

$$M(q,\dot{q})\ddot{q} + g(q) = \tau \quad (2)$$

$$\therefore M(q,\dot{q}) = \begin{bmatrix} \frac{m}{cos\theta cos\emptyset} & 0 & 0 & 0 \\ 0 & I_\phi & 0 & 0 \\ 0 & 0 & I_\theta & 0 \\ 0 & 0 & 0 & I_\psi \end{bmatrix} \quad (3)$$

$$g(q) = \begin{bmatrix} mg \\ -mgrsin\emptyset \\ -mgrsin\theta \\ 0 \end{bmatrix} \quad (4)$$

$$\tau = \begin{bmatrix} F \\ \tau_\emptyset \\ \tau_\theta \\ \tau_\psi \end{bmatrix} \quad (5)$$

B. *Parametric form*

To design an adaptive controller Parametric form of the dynamic model is required. Following equations are obtained by converting the above stated manipulator form dynamic equation into parametric form $Y\alpha = u$.

$$Y = \begin{bmatrix} \frac{\ddot{q_1}+g}{cos\phi cos\theta} & 0 & 0 & 0 & 0 \\ 0 & \ddot{q_2} & 0 & 0 & -gsin\phi \\ 0 & 0 & \ddot{q_3} & 0 & -gsin\theta \\ 0 & 0 & 0 & \ddot{q_4} & 0 \end{bmatrix} \quad (6)$$

$$\alpha = \begin{bmatrix} m \\ I_\phi \\ I_\theta \\ I_\psi \\ mr \end{bmatrix} \quad (7)$$

Here,

$$q = \begin{bmatrix} z \\ \phi \\ \theta \\ \psi \end{bmatrix}$$

M in terms of alpha can be stated as below.

$$M = \begin{bmatrix} \frac{\alpha(1)}{cos\phi cos\theta} & 0 & 0 & 0 \\ 0 & \alpha(2) & 0 & 0 \\ 0 & 0 & \alpha(3) & 0 \\ 0 & 0 & 0 & \alpha(4) \end{bmatrix} \quad (8)$$

III. Feedback Linearized Controller

To design feedback linearized controller, manipulation form of quadcopter dynamic equation (equation 1) is used. This section discuss the design of the feedback linearized controller implemented in this project.

A. *Feedback Linearizaton and Control Law*

- Following stated is the proposed control lay to cancel the nonlinearities in the dynamics.

$$u = M(q,\dot{q})v + g(q) \quad (9)$$

- Comparing equation (9) with equation (2).

$$\ddot{q} = v \quad (10)$$

- To control this linearized dynamics, a PD controller is suggested as follows.

$$v = \begin{bmatrix} z\ddot{_{des}} + Kp_z(z - z_{des}) + Kd_z(\dot{z} - z\dot{_{des}}) \\ Kp_\emptyset(\phi - \phi_{des}) + Kd_\emptyset(\dot{\phi} - \dot{\phi_{des}}) \\ Kp_\theta(\theta - \theta_{des}) + Kd_\theta(\dot{\theta} - \dot{\theta_{des}}) \\ Kp_\psi(\psi - \psi_{des}) + Kd_\psi(\dot{\psi} - \dot{\psi_{des}}) \end{bmatrix} \quad (11)$$

B. *Finding desired Roll and Pitch*

- From the defined trajectory, desired position, velocity and acceleration in x, y and z direction and desired yaw angle are obtained.
- To find desired roll and pitch angle in order to achieve desired states in x and y direction, internal dynamics of the quadcopter are used.
- Following are the proposed simplified internal dynamics used in this project to find desired roll and pitch.

$$\phi_{des} = \tan^{-1}\frac{(v_x \sin\psi - v_y \cos\psi)}{g} \quad (12)$$

$$\theta_{des} = \tan^{-1}\frac{(v_x \cos\psi + v_y \sin\psi)}{g} \quad (13)$$

- In the above equations, v_x and v_x are thrust in x and y direction required to follow the desired trajectory.
- So in order to find these thrusts, another simple PD controller is used.

$$v_x = x\ddot{_{des}} + Kp_x(x - x_{des}) + Kd_x(\dot{x} - x\dot{_{des}}) \quad (14)$$

$$v_y = y\ddot{_{des}} + Kp_y(y - y_{des}) + Kd_y(\dot{y} - y\dot{_{des}}) \quad (15)$$

C. *Control efforts to Motor Speed*

From applying this controller to the system, the final output will be total thrust of four motors and the moments on the three axis. The motors used in quadcopters are BLDC and the input to this motors is a PWM signal specifying the angular speed it need to rotate.

Thus in order to control the quadcopter, it is necessary to calculate required motor velocity to produce that control efforts obtained from the control law.

Following are the dynamics of quadcopter of X-configuration that relates our control outputs to the thrusts and moment produced by each motors.

$$T_1 + T_2 + T_3 + T_4 = v_1$$
$$T_1 - T_2 + T_3 - T_4 = v_2/L_1$$
$$T_1 + T_2 - T_3 - T_4 = v_3/L_2$$
$$M_1 - M_2 - M_3 + M_4 = -v_4$$

Here T_n = thrust of n^{th} motor and M_n = moment of n^{th} motor. L_1 and L_2 are the lengths of the sides of the rectangle formed by the four motors.

Considering the equations for drag and lift of an airfoil (of the propellers), it can be said that thrust and moments are directly proportional to the velocity squared. To simplify the equations and considering the disturbances, thrusts and moments are considered directly proportional to the angular velocity of the motors.

Considering these assumptions, we get following matrix equation and solving this equation we can find the motor velocities required to produce the calculated control efforts.

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} \quad (16)$$

IV. Adaptive Controller

To design adaptive controller, parametric form of quadcopter dynamic equation (equations 6 and 7) is used. This section discuss the design of the adaptive controller implemented in this project.

The states of the quadcopter are controlled using control law similar to equation (9). Only the difference is that this time the dynamics of the system are unknown. And these dynamics are updated using the adaption law to be designed in this section.

- Proposed Control law:

$$u = \hat{M}v + \hat{g} \quad (17)$$

- Proposed adaption law:

$$\dot{\hat{\alpha}} = \Gamma^{-1}\phi^T B^T P e \quad (18)$$

- In the above adaption law, Γ is a 5x5 diagonal matrix and a tuning parameter.

$$\phi = \hat{M}^{-1}Y \quad (19)$$

- $\hat{M}$ can be calculated using the equation (8).
- Y can be calculated using equation (6).
- Now to find P following 8x8 Acl matrix and a positive definite Q matrix is used.

$$Acl = \begin{bmatrix} 0 & I \\ -K_p & -K_d \end{bmatrix} \quad (20)$$

$$P = \text{lyap}(Acl', Q) \quad (21)$$

- B is a 8x4 matrix defined as follows.

$$B = \begin{bmatrix} 0 \\ I \end{bmatrix} \quad (22)$$

- Now $\hat{\alpha}$ is obtained by integrating $\dot{\hat{\alpha}}$ at every iteration.

V. Wind Force Estimator

The disturbance caused by the wind can be countered by applying inverse dynamics control, but we must know wind velocity for the same. An approach to estimate the same is discussed here.

For simplicity consider a 2-d model of the quadcopter and the wind only flows in the horizontal direction. The quadcopter is treated as a simple plank when calculating the wind force. The actual acceleration measured by the IMU can be written as:

$$\ddot{x}_a = \ddot{x}_m + C \cdot V_x|V_x| \cdot \sin\theta^2 \quad (23)$$

From this equation we can get V_x as shown below:

$$\tilde{\ddot{x}} = \ddot{x}_a - \ddot{x}_m \quad (24)$$

$$V_x|V_x| = \frac{\tilde{\ddot{x}}}{C \cdot \sin\theta^2} \quad (25)$$

$$V_x = sign\left(\frac{\tilde{\ddot{x}}}{C \cdot \sin\theta^2}\right) \cdot \sqrt{\frac{\tilde{\ddot{x}}}{C \cdot \sin\theta^2}} \quad (26)$$

x_m is calculated by the model dynamics which we used in the feedback linearized control by giving it the control input we applied.

Similarly, we can also get V_x by measuring acceleration in z direction as:

$$V_x = sign\left(\frac{\tilde{z}}{C \cdot \sin\theta \cdot \cos\theta}\right) \cdot \sqrt{\frac{\tilde{z}}{C \cdot \sin\theta \cdot \cos\theta}} \quad (27)$$

The value of V_x can be overtime estimated with least squares estimator. However, here we only take into account the last N values of the V_x. The reason is that the wind is expected to change and earlier values would just generate error in our estimation.

Below given equation is used to update the value of the Vx:

$$V_{est,t} = \frac{\sum_1^N V_t}{N} \quad (28)$$

$$V_t = \frac{(1 - K_k)\cdot V_{x,t} + K_k \cdot V_{est,t-1} \cdot N}{N+1} \quad (29)$$

$$K_k = K \cdot \left(\frac{V_{est,t} - V_{x,t}}{V_{est,t}}\right) \quad (30)$$

After this we update the array to remove the oldest estimate of the V_t and replace it with the new estimate.

However, we were not able to implement and test this algorithm. But a potential flow is that we need to know the exact dynamics of the system to estimate the wind force. Uncertain dynamics will induce error in our estimation as we are using the model to get x_m.

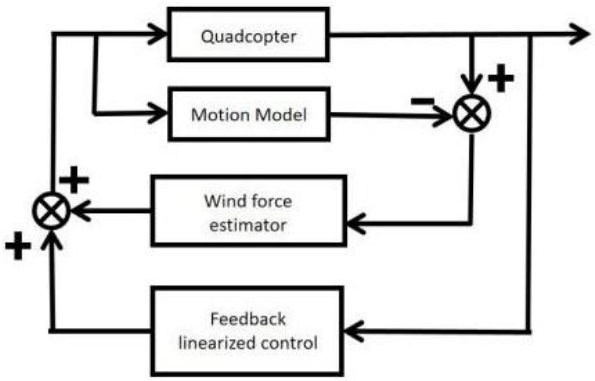


Figure 1: Controller block diagram

VI. MATLAB SIMULATIONS

ODE45 based simulations were run in MATLAB environment in order to simulate the designed controllers. The simulations were run in three steps. First, the feedback linearized controller was tuned and tested in the simulator. Then a variation was given in the controller dynamics and results were simulated. Now to overcome this disturbances and adapt to it, adaptive controller developed in the project was implemented and the results were simulated. Results of each simulation are discussed in the following sections.

A. Feedback linearized controller without disturbances

Following are the original parameters of the model used in the simulation.

- Mass = 0.18
- I_pitch = 0.00025
- I_roll = 0.00025
- I_yaw = 0.00037

The trajectory provided was a helix in 3D. After tuning the parameters, following were the results obtained.

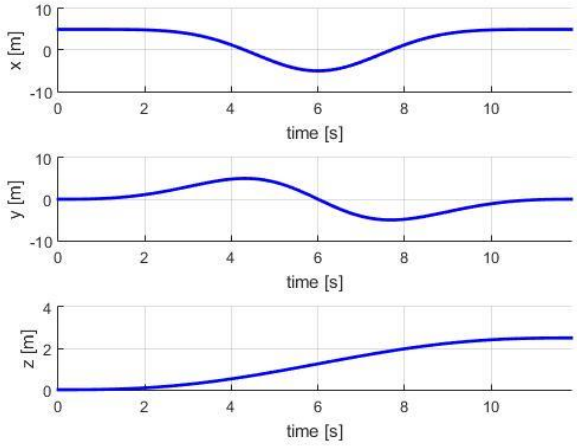


Figure 2: Feedback linearized controller Position vs time

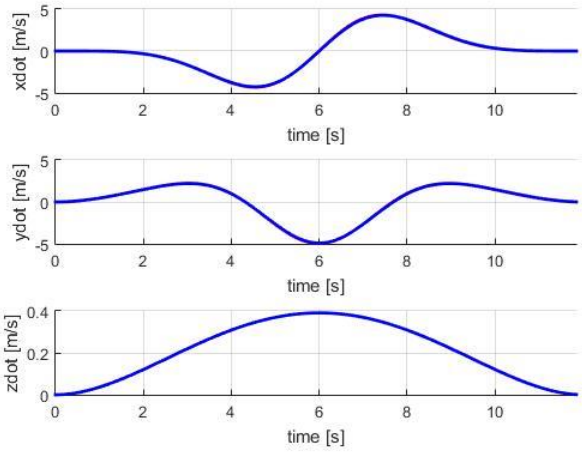


Figure 3: Feedback linearized controller Velocity vs time

As seen in the above plots, the system exactly follows the trajectory. This shows the feedback linearized controller is able to control the system accurately if the dynamics of the system are known.

B. Feedback linearized controller with disturbances

Following are the erroneous parameters of the used in the controller.

- Mass = 0.05
- I_pitch = 0.00015
- I_roll = 0.000332
- I_yaw = 0.0002728

Following were the results obtained for the same helix trajectory.

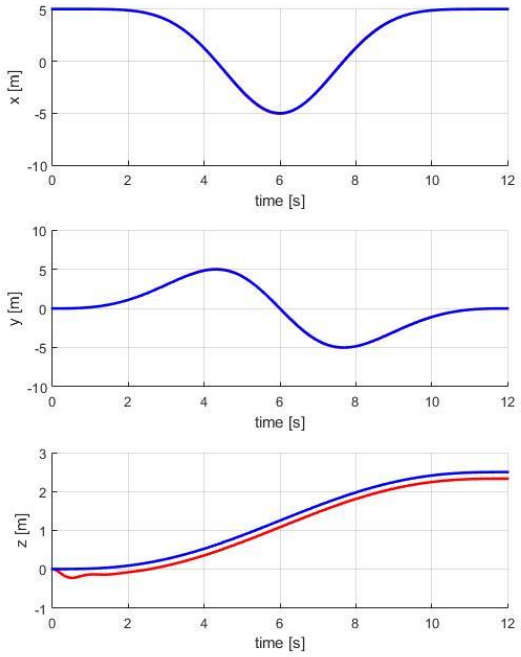


Figure 4: Disturbance Position vs time

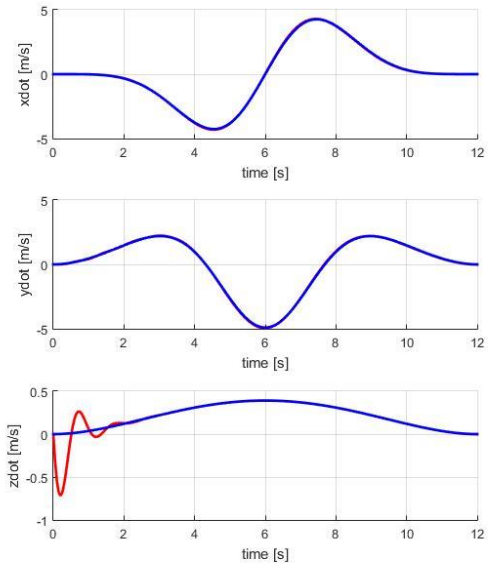


Figure 5: Disturbance Velocity vs time

It can be seen in the above plots that a constant error can be seen in the position in z direction. As the controller is well tuned and the system being in MATLAB, the controller is able to correct the errors in the other states. To overcome these disturbances, adaptive controller is implemented in the following simulation.

C. Adaptive Controller with disturbances

The erroneous parameters discussed in the above simulation are considered as nominal parameters. Following are the results obtained for the helix trajectory.

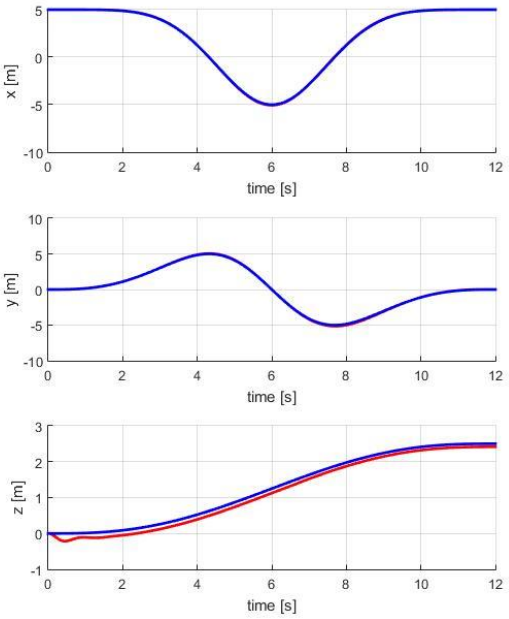


Figure 6: Adaptive controller Position vs time

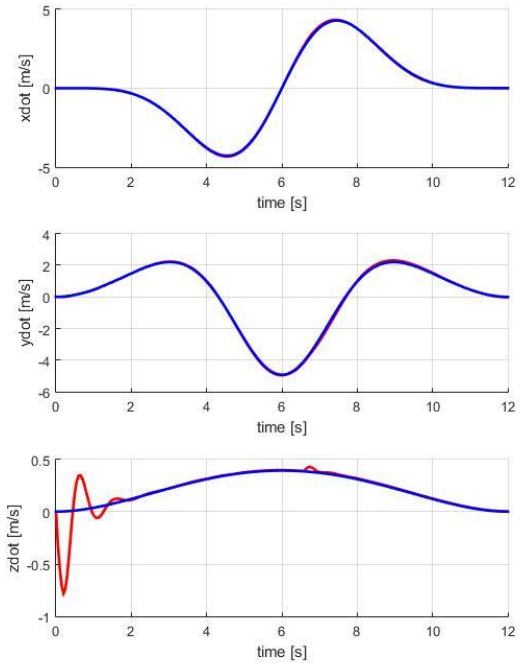


Figure 7: Adaptive controller Velocity vs time

It can be in the above plots that the adaptive controller is able to minimize the steady state error in the z direction. This proves that the adaptive controller developed is able to adapt to the dynamics and control the system accurately. The alpha updates as the time proceeds and it settles to a constant value. Following plot shows values of five aplhas vs time.

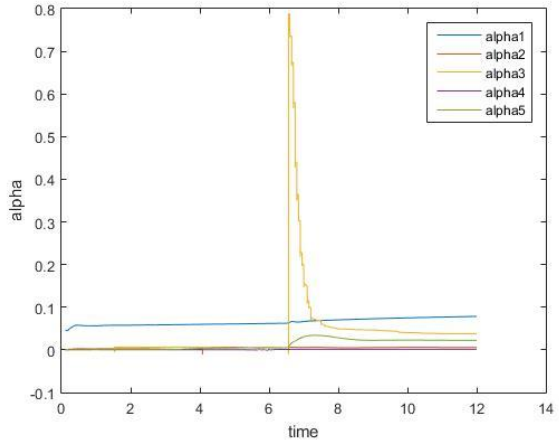


Figure 8: alpha vs time

It can be seen from the plot that as the time proceeds, alpha settles to a constant value. Though this values are not similar to the original values which are [0.1800; 0.0003; 0.0002; 0.0004; 0]. A significant peak was observed in the value of alpha 3 but then the system stabilizes and settles to a constant value. Following chapter discusses the implementation of feedback linearized controller on a gazebo quadcopter model.

VII. Gazebo Simulation

Feedback linearized controller derived above was implemented on the gazebo model of the quadcopter. The main difference between the MATLAB simulation and the gazebo simulation was the final control parameters. The MATLAB simulation was designed to take just the input of total thrust of four motor and the moments. But in the gazebo model, the final control input is the velocity of the motors.

Thus the derived conversion equation was applied in order to find the velocities of the four motor. Tuning this model was the most challenging task. The main reason for this is the limits of the motors. The motors are not capable of producing negative thrust and also there is a upper limit for the thrust produced. This puts a limit on the control input applied.

If the control effort exceeds this limit, the controller will not be able to stabilize the quadcopter.

After carefully tuning the parameters, the controller was able to guide the quadcopter on a cubic trajectory between two points. Though the max speed was highly limited and there were minute oscillations observed.

Code for automated trajectory generation was developed which develops trajectory passing through set of points provided. Applying this to the controller, the quadcopter was able to trace a trajectory passing through a set of points. Following image shows the algorithm in action and quadcopter successfully achieving the target point.

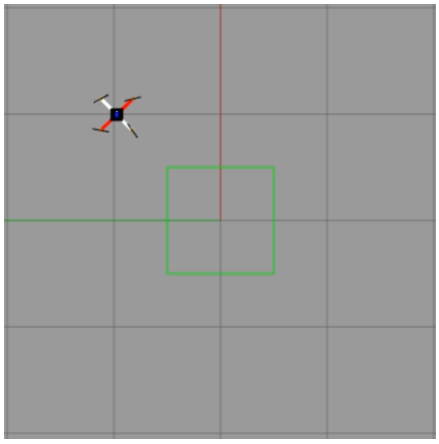


Figure 9: Gazebo Simulation 1

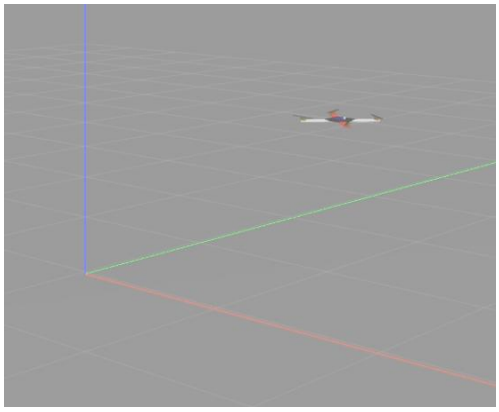
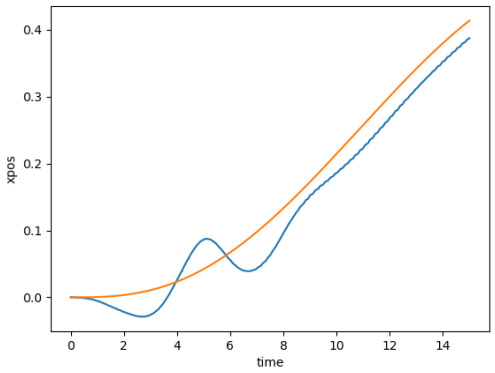
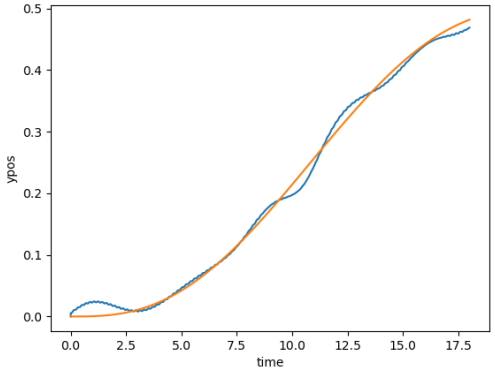


Figure 10: Gazebo Simulation 2

Following plots were obtained for a trajectory developed by the trajectory generator between two defined points.





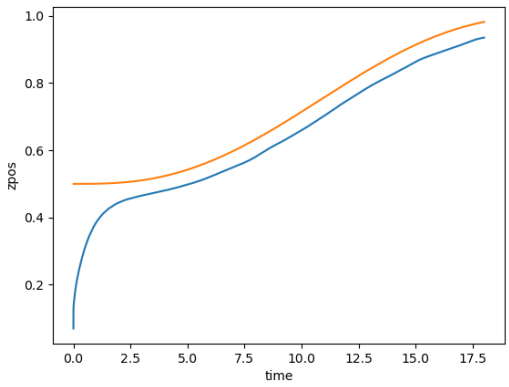


Figure 11: Gazebo Plots

As seen from the plots, the controller was able to converge the system to the trajectory and follow it. There were still some oscillations and steady state errors observed.

VIII. Conclusion

In this project, dynamics of a quadcopter were studied and controller based on that dynamic were derived. Feedback linearized controller was derived and tested in MATLAB and Gazebo environments.

The behaviour of feedback linearized controller was studied under disturbance. To overcome the problems in feedback linearized controller, adaption law was designed. The final derived adaptive controller was implemented and tested in MATLAB environment. The effects of tuning parameters on the controller were studied.

Feedback linearized controller was implemented on the gazebo model from scratch. Effects of various environmental parameters on the quadcopter were studied. The control parameters were tuned to control the quadcopter in the gazebo environment.

Moreover, a trajectory generator was developed which generates the trajectory through the given set of points. In summation, we were able to control the quadcopter just by defining a set of check points for the quadcopter to reach.

The controllers developed in this project can be used to control a quadcopter in the real world. Moreover, adaptive controller developed during the project can be implemented on the gazebo model.

There is a huge scope of development in the field of aerial robotics and quadcopter design being the most used design for aerial robots, there is a huge scope of research for developing advanced controllers for more accurate and agile control.

References

[1] N. Michael, D. Mellinger, Q. Lindsey and V. Kumar, "The GRASP Multiple Micro-UAV Testbed," in IEEE Robotics & Automation Magazine, vol. 17, no. 3, pp. 56-65, Sept. 2010, doi: 10.1109/MRA.2010.937855.

[2] P. Patel and J. Dave, "Design and Dynamic Modelling of Quadrotor VTOL aircraft," 2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA), 2020, pp. 105-112, doi: 10.1109/ICECA49313.2020.9297379.

[3] V. Ghadiok, J. Goldin and W. Ren, "Autonomous indoor aerial gripping using a quadrotor," 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Francisco, CA, 2011, pp. 4645-4651, doi: 10.1109/IROS.2011.6094690.

[4] J. P. F. Guimarães et al., "Fully Autonomous Quadrotor: A Testbed Platform for Aerial Robotics Tasks," 2012 Brazilian Robotics Symposium and Latin American Robotics Symposium, Fortaleza, 2012, pp. 68-73, doi: 10.1109/SBR-LARS.2012.18.