MODELING QUADCOPTER

FINAL PROJECT REPORT

For course:

Project in Computer Science and Engineering

Arjun Vijayanatha Kurup

Suprervisor: Johan Thunberg

June 02, 2019

Abstract

With the current paradigm shift of automation, there is a common strive towards making vehicles such as cars and trucks more automated and autonomous. This is captured by the six levels of automation issued by Society of Automotive Engineers (SAE); they describe the steps between no automation, Level 0, and full automation, Level 6. Such automation initiatives reach beyond land based vehicles, and are also applied to airplanes and Unmanned Aerial Vehicles (UAVs) or drones. With the introduction of modern AI-algorithms in combination with increasing computational resources, we currently witness a transition into uncharted territory where machine learning methods, based on for example Deep Learning, bring new opportunities. Machines have the capability to surpass and outperform human capabilities in the context of autonomous sensing, control and navigation. In this project, we combine mathematical modeling, control theory and Deep Learning to implement an automated UAV simulation framework. The reason behind developing simulation framework instead of a physical model is due to the following reasons. Firstly, the expenditure in building a quadcopter is quite high; secondly, a mathematical model of a quadcopter can be derived easily based on the dynamics, kinematics and maneuverability; thirdly, different control mechanisms such as a PID controller or a MPC controller can be used and tested safely our simulation framework, which serves as a digital clone of the physical model itself. We use a deep learning neural network called as Alexnet to detect the obstacles in the trajectory path of a mathematically modeled quadcopter, which is being simulated in a 3D Euclidean space using Matlab Simulink. A PID controller is being used to stabilize the quadcopter and to reduce the error generated by the whole model during simulation. At the discrete time points, the learning algorithm provides coordinate points to which the quadcopter shall move

in order to avoiding an obstacle. The algorithm implemented for the purpose of automation is illustrated down in the result section.

Contents

List of Figures

1 Introduction

Recently, there has been an increased interest in drone applications. Aerial robotics is one of the fast growing and interesting concept in the field on robotics. Unmanned Aerial Vehicles (UAVs) also named drones have been developed and used for various purposes. It is an aircraft without a human pilot where control is done remotely. The market for drones and applications, because there is no pilot, a vast amount of applications has increased dramatically over the last decade. Military applications, firefighting, and even e-commerce drive this development. As of today, highly accomplished solutions are offered for easiness and reliability. The drones use low-cost microprocessors, optimized mechanical structures, and advanced software solutions to give safe and reliable flight missions. The drones are used to provide safe and reliable flight missions. The drones are used in missions where human beings are not capable of handling the situation [6].

The aim of this project is to provide a non-linear dynamic model of a quadcopter for simulation purpose and then furthermore provide a controller for stabilization. Over the past years, abundant research on quadcopter modeling has been done. Linear and non-linear models have been provided by researchers [1]. The quadcopter has fixed pitch rotors with variations in rotor speed for controlling the vehicle motion. Most of the techniques for controlling the rotors are based on the classical approaches from linear and non-linear control. The most widely used linear technique is Proportional-Integral-Derivative(PID) or Proportional-Derivative(PD)

controller. The non-linear controllers comprise, besides feedback linearization, also hybrid and learning-based flight control systems. However, the most common control system used is PID or restricted PD. Such controllers can be used in both the linear and non-linear models of the quadcopter and have proven to be more effective. The dynamic model of the quadcopter is nonlinear and a linear model is simply an approximation of the non-linear model itself, i.e., the linearization of the non-linear system close to a reference point.

1.1 Problem Formulation

Classical control mechanisms such as PID controller, PD controller, MPC have been used for a long time by different researchers and hobbiest. But there tends to be lack of controller implementation that works on the basis of deep learning and machine learning techniques. The classical approach can be used up to a point but when it comes to automation, there tends to be a problem being faced by the controller, that is the accuracy of the classical controller is limited. In this project, we propose a different control mechanism based on deep learning in order to acquire accuracy based on prediction learned from the data provided by the quadcopter itself. This provides a closed loop control system that can navigate autonomously in the 3D Eucledian space.

Secondly, a demonstration of how effectively matlab can be used to implement the model, control mechanism, as well as deep learning, and simulation without using any other tools and softwares.

1.2 Background

A quadcopter is a UAV and a drone which has four rotors placed in a quadratic formation on a square body frame. Thus, it has twice the number of rotors in comparison to the helicopter. This mechanism can be seen to be complicated at first sight. A quadcopter's dynamical state has six degrees of freedom(three translational and three rotational) but there are only four rotors, which constraint the directions of motions and reduces the degrees of freedom. The rotors are fixed, and moments in different directions are achieved by increasing and decreasing the rotor speed. Electronic assistance is used to control the rotor speed and the quadcopter itself. The rotors are placed in such a way that two rotors opposite to each other rotate in a clockwise direction while the other two rotate in a counterclockwise direction. Such configuration provides stability and hovering capabilities, which enables attaining full range of motion in multiple directions. The quadcopter comprises a microprocessor, rotors, rigid body frame.

1.3 Related Work

In [8], Wang covers the stabilization of the dynamic model of the quadcopter by using a cascade PID feedback control algorithm in the presence of external disturbances. In [5], Luukkonen proposed the basics of quadcopter modeling and control. The paper presents the differential equations of the quadcopter modeling which are derived from Newton-Euler equations and Euler-Lagrange equations [2]. The paper also proposes the use of a PD controller instead of a PID controller for the stabilization of the quadcopter dynamics. From the described works, the dynamics of the quadcopter was adopted into this project.

2 Mathematical Model

2.1 Kinematic model

The quadcopter is represented in two references coordinate system. Reference coordinates are used to describe the position and orientation of the quadcopter. The first being fixed and referred to as the inertial coordinate system, shown in Figure 4 represented as ξ. The coordinates of this frame are captured by the triple(x, y, z), influenced by the rotors where z will be in the downward direction and the other two in x and y-direction, are given in the Figure 4. The second coordinate system is a mobile navigation frame based on the body-frame of the quadcopter itself, shown in Figure 5. The origin of the navigation frame will be at the center of mass of the quadcopter(quadcopter).

Apart from the coordinate systems, the quadcopter is defined by the angular position in the body frame, η which comprises three angles: pitch angle determining the rotation of the quadcopter in the y-axis Figure 1, the roll angle determining the rotation about the x-axis Figure 2 and the yaw angle determining the rotation in the z-axis Figure 3. These angles are represented as ϕ for pitch angle, θ for roll angle and ψ for yaw angle and together denoted as η. We define the vector q, see below, by concatenating the two vectors ξ and η [5].

$$\xi = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \quad \eta = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix}, \quad q = \begin{bmatrix} \xi \\ \eta \end{bmatrix} \tag{1}$$

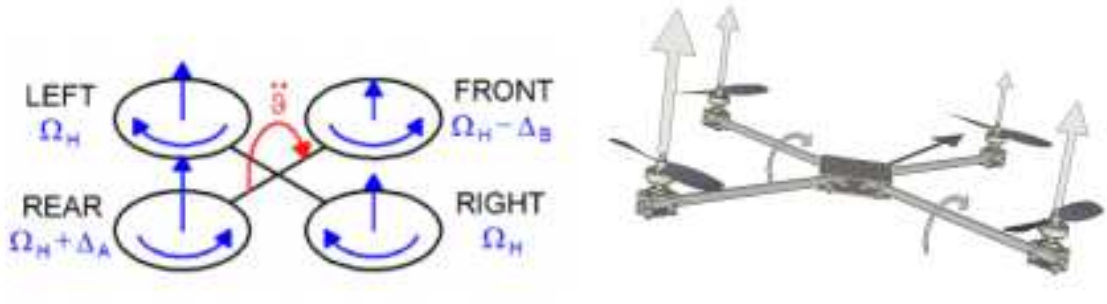

Figure 1: Pitch position [7]

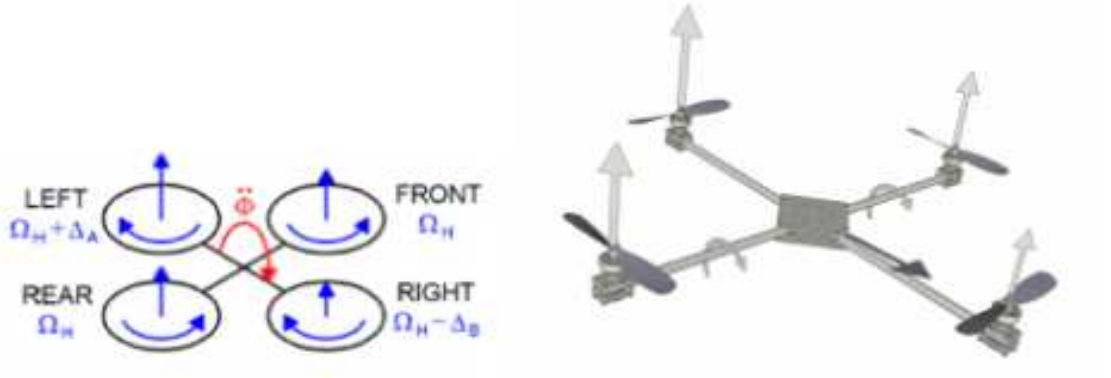

Figure 2: Roll position [7]

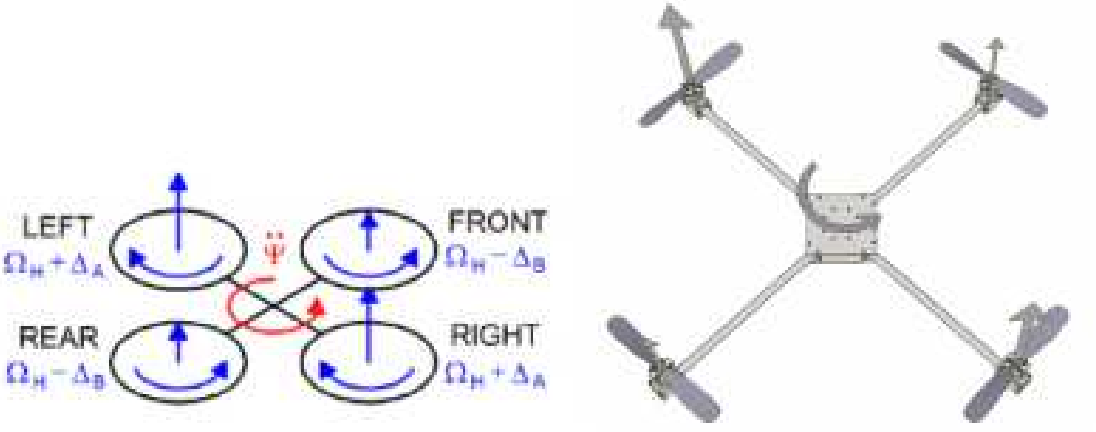

Figure 3: Yaw position [7]

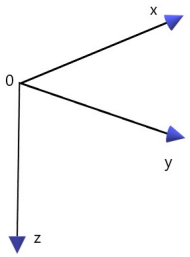

Figure 4: Inertial Frame

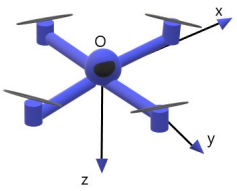

Figure 5: Body Frame

We continue to intrude the rotation matrix. It is used to perform a rotation in Euclidean space.The vector of angles, i.e, η as defined above, is a representation of the rotation matrix. In the case of the quadcopter, the rotation matrix is used to convert the attitude and angular speed of the quadcopter from the body frame to an inertial frame. We denote the rotation matrix representing the rotation as R and from inertial frame to body frame by R (where $R^{-1} = R^T$). The forces and moment necessary for the rotation and orientation of the quadcopter need to be corrected in the body frame. Hence the rotation matrix is used to adjust the angular positions such as the roll, pitch and yaw angles(ϕ, θ, ψ), the Euler angles about the x, y and z-axes respectively.

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\phi) & -sin(\phi) \\ 0 & sin(\phi) & cos(\phi) \end{bmatrix}$$

$$R_y = \begin{bmatrix} cos(\theta) & 0 & sin(\theta) \\ 0 & 1 & 0 \\ -sin(\theta) & 0 & cos(\theta) \end{bmatrix}$$

$$R_z = \begin{bmatrix} cos(\psi) & -sin(\psi) & 0 \\ sin(\psi) & cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Now, the rotation matrix R can be explicitly constructed from η. This is done by constructing three matrices, $R_x, Rx_y and R_z$, see above, and then combine them to get $R = R_x R_y R_z$. The angular velocity of the three angles is determined as [p q r] in the body coordinate system. $\dot{\phi}, \dot{\theta}, \dot{\psi}$ are represented as angular velocity in the inertial frame.

$$W_\eta = \begin{bmatrix} 1 & 0 & -sin(\theta) \\ 0 & cos(\phi) & cos(\theta)sin(\phi) \\ 0 & -sin(\phi) & cos(\theta)cos(\phi) \end{bmatrix}, \quad \dot{\eta} = \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}, \quad \nu = \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

$$\nu = W_\eta \dot{\eta} \tag{2}$$

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -sin(\theta) \\ 0 & cos(\phi) & cos(\theta)sin(\phi) \\ 0 & -sin(\phi) & cos(\theta)cos(\phi) \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}$$

2.2 Dynamic model

The inertial matrix for the quadcopter is a diagonal matrix consisting of Ixx, Iyy, Izz and the other elements in the matrix are considered zero due to the symmetry of the quadcopter.

$$I = \begin{bmatrix} Ixx & 0 & 0 \\ 0 & Iyy & 0 \\ 0 & 0 & Izz \end{bmatrix} \tag{3}$$

Where the Ixx, Iyy and Izz are the area moments of inertia about the principle axes in the body frame.

In this section, the dynamic model of the quadcopter is derived. The equations are based on Newton-Euler equations. Newton-Euler equations are used to describe the rotational and translational motion of a rigid body. The rotational motion is derived in the body frame using the Newton Euler method using the following equation

$$R = R_x.R_y.R_z \tag{4}$$

$$R = \begin{bmatrix} c(\psi)c(\theta) & c(\psi)s(\theta)s(\phi) - s(\psi)c(\phi) & c(\psi)s(\theta)c(\phi) + s(\psi)s(\phi) \\ s(\psi)c(\theta) & s(\psi)s(\theta)s(\phi) + c(\psi)c(\phi) & s(\psi)s(\theta)c(\phi) - c(\psi)s(\phi) \\ -s(\theta) & c(\theta)s(\phi) & c(\theta)c(\phi) \end{bmatrix}$$

in which c(x) = cos(x) and s(x) = sin(x).

Rotor speed

The movement of the quadcopter is determined by the thrust produced by each rotor. The quadcopter is under actuated in that it only has four degrees of freedoms in actuation for controlling the quadcopter's 6DOF pose from four inputs. The thrust produced by each rotor is controlled by the control system that provides the accurate thrust required to provide the desired maneuverer. The 6DOF consist of three rotational movements(yaw, pitch, and roll) and three translational movements(movement in x, y, and z-axis). The angular velocity of the rotors are represented by

$$\omega_i = \begin{bmatrix} 1 & 0 & -sin(\theta) \\ 0 & cos(\phi) & cos(\theta)sin(\phi) \\ 0 & -sin(\phi) & cos(\theta)cos(\phi) \end{bmatrix} \tag{5}$$

where i is the rotor speed of each individual rotor created according to the euler angles.

Forces

The angular velocity of the rotors ω_i cause forces f_i in the opposite direction of the z-axis. There is a requirement to define two physical effects which are the aerodynamic forces and moments produced by the rotors in order to create a lift force and aerodynamic moment. Equation (6) is used for deriving the lift force and equation (7) is used to derive the moment force.

$$f_i = k\omega_i^2 \tag{6}$$

$$\tau_{M_i} = b\omega_i^2 + I_M\dot{\omega}_i \tag{7}$$

Where, k is the lift constant,

b is the drag constant,

I_M is the inertial moment,

$\dot{\omega}_i$ is omitted in this case as the effect is considered small. By identifying the forces and the moments generated by the rotors, we can study the moments acting on the quadcopter.

Actuator dynamics

The inputs to the rotors determine the degrees of freedom. Commonly the control inputs that are considered are one for the vertical thrust and one for each of the angular motions. The combined forces generated by the rotors create thrust T in the direction of the body z-axis. Using the equation 6 derived before, the Thrust which is required to produce the uplift on the quadcopter can be produced by using the equation,

$$T = \sum_{i=1}^{4} f_i = k\sum_{i=1}^{4} \omega_i^2 \tag{8}$$

The torque τ_B consist of τ_ϕ, τ_θ, τ_ψ which is the moment's input provided into the actuators to create the angular positional changes. They can be explained

as [8]:

$$\tau_B = \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} lk[({\omega_2}^2 + {\omega_3}^2) - ({\omega_1}^2 + {\omega_4}^2)] \\ lk[({\omega_1}^2 + {\omega_2}^2) - ({\omega_3}^2 + \omega_4 2)] \\ lk[({\omega_1}^2 + {\omega_3}^2) - ({\omega_2}^2 + {\omega_4}^2)] \end{bmatrix} \tag{9}$$

l being the distance between the rotor and the center of mass of the quadcopter, defined with a value of $0.225m$ from [5], k being the lift constant, value defined as $2.980.10^{-6}$ from [5] By using these equations of motion, the roll movement, the pitch movement, and the yaw movement can be acquired in the quadcopter.

Newton-Euler equation

The dynamics of the rigid body of the quadcopter can be described with the Newton Euler equations. It is the grouping together of Newton's two laws of motion for a rigid body. From the equation 10, Newton's law states the following matrix relation for the total force acting on the quadcopter

$$m(\omega_B \times V_B + \dot{V}_B) = R^T G + T_B \tag{10}$$

Where m is the mass of the quadcopter, G is the gravitational force, and T_B is the total thrust of the rotors.

$$\ddot{\xi} = \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix}$$

$$G = -g \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$RT_B = \frac{T}{m} \begin{bmatrix} cos(\psi)sin(\theta)cos(\phi) + sin(\psi)sin(\phi) \\ sin(\psi)sin(\theta)cos(\phi) - cos(\psi)sin(\phi) \\ cos(\theta)cos(\phi) \end{bmatrix}$$

$$m\ddot{\xi} = G + RT_B \tag{11}$$

In the inertial frame, the centrifugal force is nullified as the term is found to be small and for this reason, the gyroscopic moments are removed in the controller formulation. Equation. From the Euler equations provided in the paper [5], the angular acceleration of inertial , centripetal force and the gyroscopic forces are equal to the external torque in the body frame.

$$I\dot{\nu} + \nu \times (I\nu) + \Gamma = \tau \tag{12}$$

Hence the angular acceleration can be calculated from the equation 12 as

$$\dot{\nu} = I^{-1} (\tau - \nu \times (I\nu) - \Gamma) \tag{13}$$

$$\Gamma = I \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \omega_\Gamma \tag{14}$$

$$\omega_\Gamma = \omega_1 - \omega_2 + \omega_3 - \omega_4$$

where I is the inertia matrix 3, τ is the moments from 9, ν is the angular velocity in the body frame 2.1, and Γ 14 the gyroscopic forces acting on the body.

The angular acceleration is calculated using the transformation matrix and its time derivative equation These equations are included in the Aerospace blockset in the Simulink Figure 6, which enables to use of any type of equations of motion to test such as the 6DOF Quaternion which works with respect to body axes, 6DOF wind(Quaternion), which works on the basis of the wind axes, etc.

The angular acceleration $\ddot{\phi}, \ddot{\theta}, \ddot{\psi}$ for obtaining the angular positions ϕ, θ, ψ in the body frame are calculated using the equation 15

$$\ddot{\eta} = \frac{d}{dt}\left(W_\eta^{-1}\nu\right) = \frac{d}{dt}\left(W_\eta^{-1}\right)\nu + W_\eta^{-1}\dot{\nu} \tag{15}$$

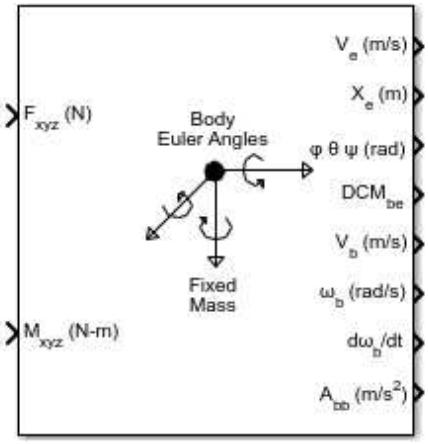

Figure 6: 6DOF Euler Angles Block

Aerodynamic forces

The model is a simplification of the complex dynamic interaction. To derive a realistic result, the drag force generated y air resistance is included.

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = -g \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} + \frac{T}{m} \begin{bmatrix} cos(\psi)sin(\theta)cos(\phi) + sin(\psi)sin(\phi) \\ sin(\psi)sin(\theta)cos(\pi) - cos(\psi)sin(\phi) \\ cos(\theta)cos(\phi) \end{bmatrix} - \frac{1}{m} \begin{bmatrix} A_x & 0 & 0 \\ 0 & A_y & 0 \\ 0 & 0 & A_z \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \tag{16}$$

In which Ax, Ay, Az is drag forces coefficient to velocities in the corresponding direction of the inertial frame. Several other aerodynamic forces can be applied, but due to the complexity of the equations, they are avoided in this model.

3 Controller

3.1 PID Controller

After the implementation of the quadcopter model based on the derivatives and equations, a PID controller was developed and implemented. The implementation was done after verifying the quadcopter model by implementing an open-loop controller where the input parameters were given for the simulation. The PID controller is used to generate the control signals that is required for stabilization of the quadcopter at a reference point. Within the controller an error is calculated as the difference between the desired and measured signal values. The controller attempts to minimize the error value by adjusting the control inputs [5] [4]. The block diagram for the PID controler

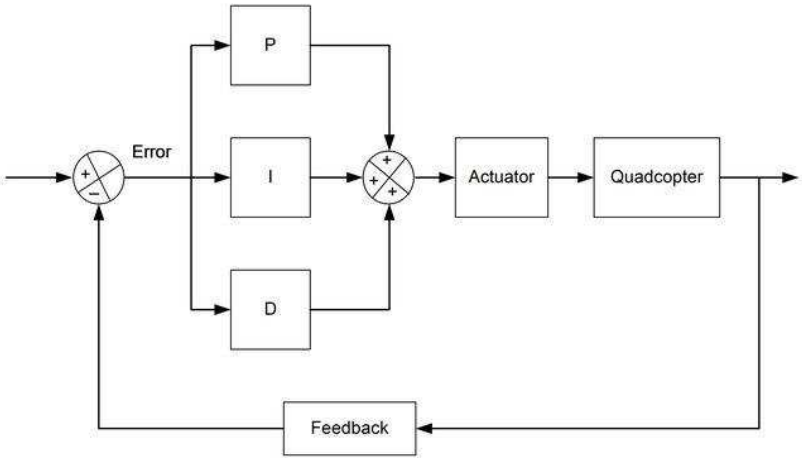

Figure 7: PID Controller Block diagram

is show in the figure 7.

Altitude Controller

An altitude controller for the mathematical model of the quadcopter is developed using the equation (17). The generated output will be the $U1$ which is the parameter input for the determining the altitude of the quadcopter. Figure 8 shows the implementation of the PID controller for the calculation of thrust.

$$U1 = k_p(z - zd) + k_d(\dot{z} - \dot{z_d}) + k_i \int (z - z_d)dt \tag{17}$$

where

k_p being the proportional gain

z_d being the desired altitude

k_d being the derivative gain

$\dot{z_d}$ being the Desired altitude rate of change k_i being the Integral gain

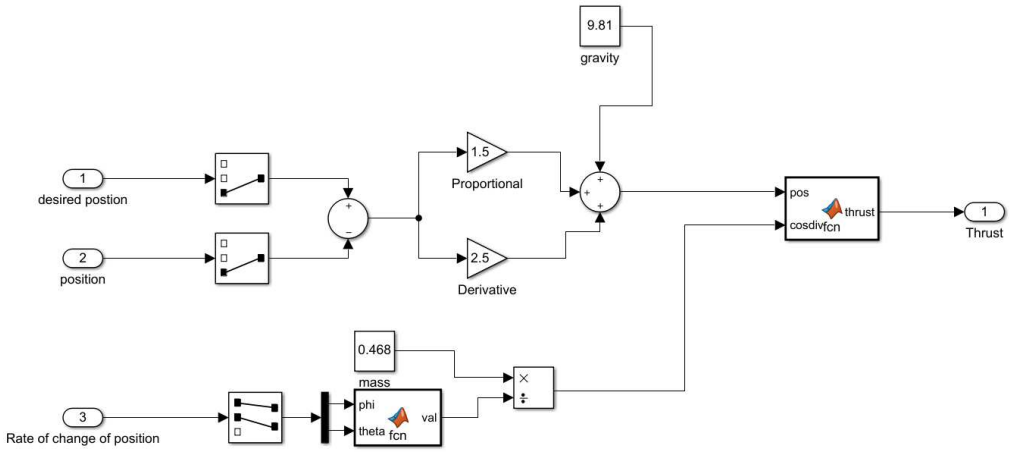

Figure 8: PID Controller Block Thrust

Attitude Controller

Three other PID controller is implemented to control the roll angle(ϕ), pitch angle(θ) and yaw angle(ψ) which are derived as $U2, U3$ and $U4$. These three angles works the same way as the altitude controller. The equations used for determining the angles are given below.

$$U2 = k_p(\phi_d - \phi) + k_d(\dot{\phi}_d - \dot{\phi}) + k_i \int (\phi_d - \phi)dt \tag{18}$$

$$U3 = k_p(\theta_d - \theta) + k_d(\dot{\theta}_d - \dot{\theta}) + k_i \int (\theta_d - \theta)dt \tag{19}$$

$$U4 = k_p(\psi_d - \psi) + k_d(\dot{\psi}_d - \dot{\psi}) + k_i \int (\psi_d - \psi)dt \tag{20}$$

where ϕ_d being the desired roll angle $\dot{\phi}$ being the desired roll angle rate of change θ_d being desired pitch angle $\dot{\theta}$ being the desired pitch angle rate of change ψ_d being desired yaw angle $\dot{\psi}$ being the desired yaw angle rate of change

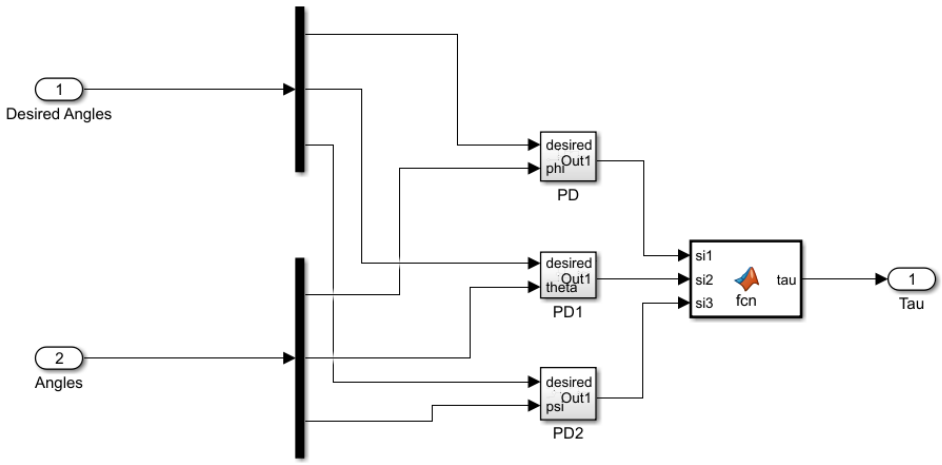

Figure 9: PID Controller Block Angles

3.2 Learning based controller

For the trajectory planning and control methodology of the quadcopter simulation in a closed environment, a pretrained deep learning network called as the Alexnet [3] developed by Alex Kriszhevsky is used. Alexnet is a convolutional neural network(CNN) trained on data set containing more than a milion images. Alexnet is mostly used for image classification and is capable of classifying 1000 different category of images. Matlab contains the supporting toolboxes and packages necessary for implementing the Alexnet. Alexnet was chosen compared to other neural network algorithms due to the accuracy of the prediction and reduced error rate. Training a deep learning network model can take hours, days or sometimes weeks, depending on the size of the data set and the amount of processing power. Alexnet consists of eight layers in total, five of the layers are convolutional and three are fully-connected as shown in the Figure 10 [3].

For the purpose of transfer learning, the final layers were trained with specific

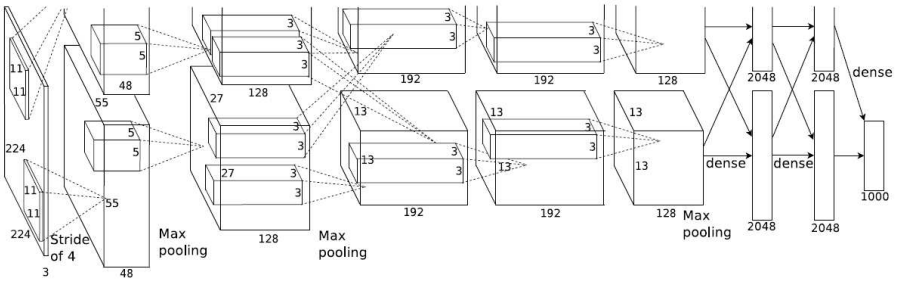

Figure 10: Alexnet Architecture

data sets. For this purpose, an NVIDIA GTX 1050Ti was used to train the network using MATLAB. This provides more speed compared to training the network using a normal CPU processor which would have taken days to train. The classification of the object acquired from the view point which acts as a camera is necessary to determine the center of the object and determine a radius around the object.

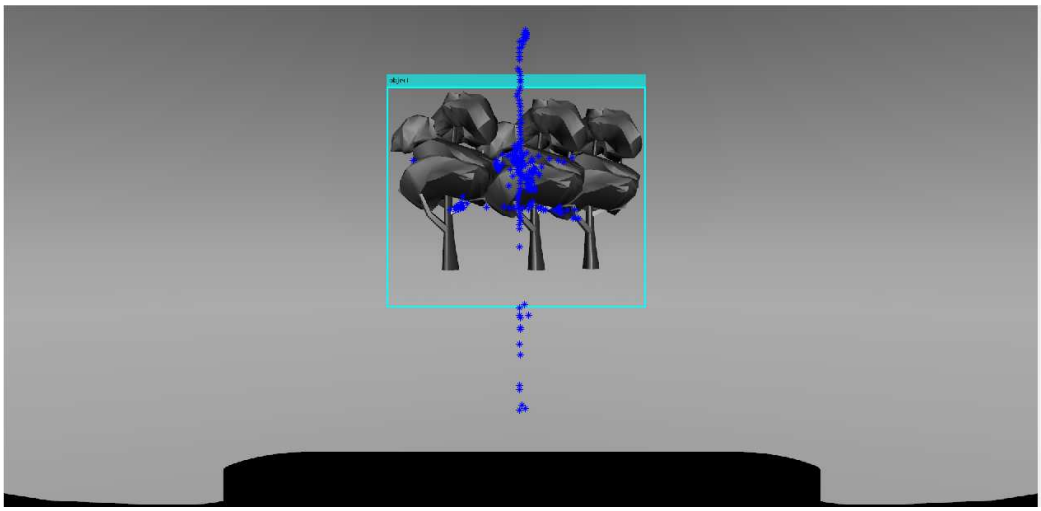

Figure 11: Obstacle Detection

A boundary point from the radius of the object will be chosen in order to obtain the trajectory around the object to reach a point. The output from the Alexnet

box will be the coordinate or the point to which the quadcopter will be able to fly in the simulation without facing the obstacle. The coordinates will be given as desired input into the PID controller as x and y in the equation (1).

4 Simulation

The simulation of the mathematical model, the control mechanism, learning process are were simulated using Matlab and Simulink. Different toolboxes such as the Aerospace Blockset, Computer Vision Toolbox, Deep learning Toolbox and Simulink 3D animation toolboxes were used to simulate the whole scenario. The mathematical equations were written in Matlab function block. Input parameters necessary for the simulation were taken from [5]. Figure 12 shows the Simulink block consisting of the VR-Sink containing the CAD model, and the output from the Video output is shown in the Figure 13. The initial input values such as the x,y and z coordinates were given as input to the controller block, which is the desired coordinate position that the quadcopter needs to acquire from the learning.

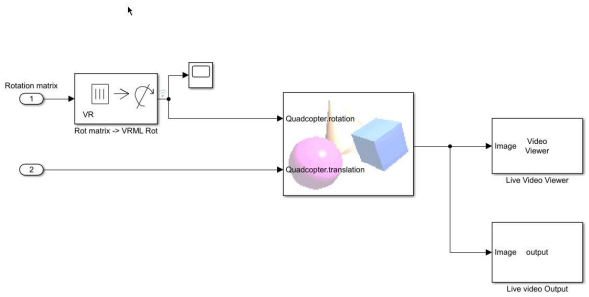

Figure 12: Simulation Block

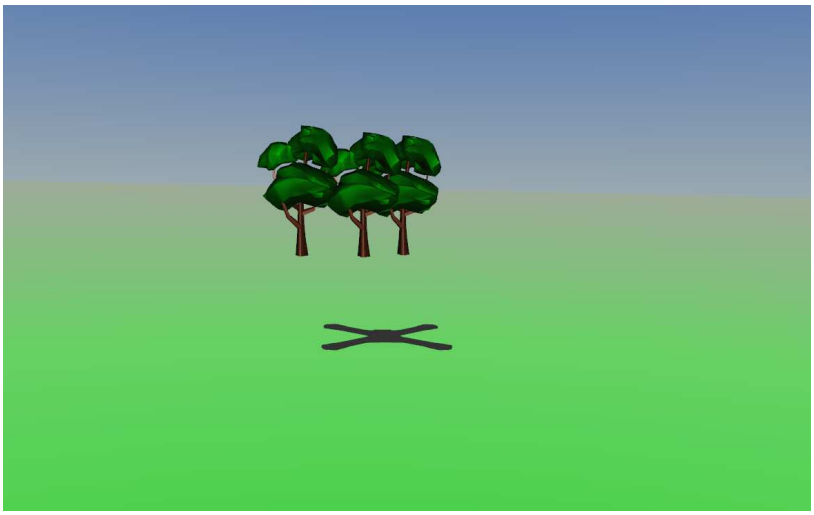

Figure 13: Image from Simulation Block

5 Results

The simulation of the quadcopter is done using Matlab and Simulink. The above provided mathematical derivations were used to develop a model of the quadcopter. The Model blocks of the quadcopter are shown in the figure 14 The control gain values required for the stabilization of the quadcopter was implemented using the inbuilt tool provided by Matlab called the PID tuner. In order to test the quadcopter, the parameters required such as the x, y, z values and ϕ, θ, and ψ were provided as a script using the 'From spreadsheet' block provided inside Simulink. This helped in performing a close examination of the results provided. For the simulation of the quadcopter, apart from the plots, a CAD model which is imported into the Simulink using VR-Sink block from 3D animation toolbox is used to demonstrate the working of the quadcopter in a simulation environment. The outputs from the 6DOF-Euler block, xyz position and the three angles ϕ, θ, ψ were taken as the input for the VR-Sink block which contains the CAD file as .wrl extension.

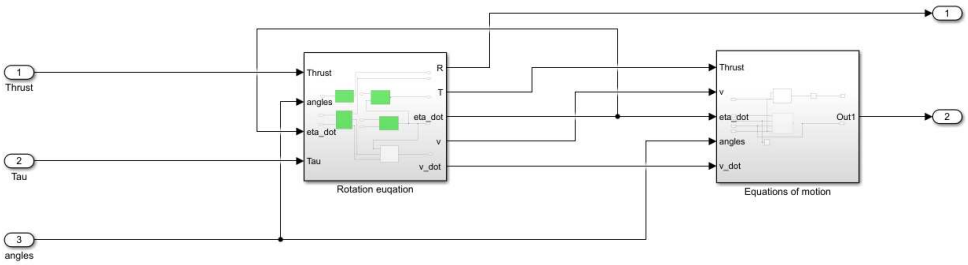

Figure 14: Controller and Model BLock

6 Conclusion and Future development

In this work, firstly a mathematical model of a quadcopter is derived using Newton-Euler's equation and the law of motion. The model was verified through simulation. A non-linear model of the quadcopter is obtained and a linear controller is used to stabilize the quadcopter and to provide linear acceleration. The classic PID controller is implemented in this work which proved to be the most suitable controller for the project in the case of stabilization. Apart from the stabilization of the quadcopter, a non-linear control method i.e, learning based controller was implemented using a convolutional neural netwok called as the Alexnet for detecting the obstacles in the trajectory of the quadcopter. From the detected object, the distance and radius is used to determine the avoiding parameter, which is the x, y coordinate that was provided as the input to the PID controller. This enabled in detection and avoidance of obstacle using deep learning methodology and to accurately calculate the trajectory path from one point to the other point. The future work can be done by implementing a non-linear controller for the quadcopter. The use of learning algorithms such as neural network can be used to determine the accurate position, velocity, thrust and torque required for the quadcopter for an accurate flight. Apart from the con-

troller, SLAM(Simultaneous Localization and Mapping) can be implemented with the simulation so as to acquire the map of the simulated quadcopter. This can be used to determine the obstacles in an environment and to provide a trajectory to the quadcopter that should be followed.

Bibliography

[1] Younes M Al-Younes, Mohammed A Al-Jarrah, and Ali A Jhemi. Linear vs. non-linear control techniques for a quadrotor vehicle. In *7th International Symposium on Mechatronics and its Applications*, pages 1–10. IEEE, 2010.

[2] Hubert Hahn. *Rigid Body Dynamics of Mechanisms: 1 Theoretical Basis*. Springer Science & Business Media, 2013.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[4] James R Leigh. *Control theory*, volume 64. Iet, 2004.

[5] Teppo Luukkonen. Modelling and control of quadcopter. *Independent research project in applied mathematics, Espoo*, 22, 2011.

[6] Raquel Remington, R Cordero, L March, and D Villanueva. Multi-purpose aerial drone for bridge inspection and fire extinguishing. *Unpublished Thesis). Florida International University. Retrieved April*, 10:2016, 2014.

[7] Francesco Sabatino. Quadrotor control: modeling, nonlinearcontrol design, and simulation, 2015.

[8] Pengcheng Wang, Zhihong Man, Zhenwei Cao, Jinchuan Zheng, and Yong Zhao. Dynamics modelling and linear control of quadcopter. In *2016 International Conference on Advanced Mechatronic Systems (ICAMechS)*, pages 498–503. IEEE, 2016.