International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 04 Issue: 03 | Mar -2017 www.irjet.net p-ISSN: 2395-0072

Hexacopter using MATLAB Simulink and MPU Sensing

Simanti Bose[1], Adrija Bagchi[2], Naisargi Dave[3]

[1]*B. Tech student in Electronics and Telecommunication Engineering, NMIMS MPSTME, India*
[2]*B. Tech student in Electronics and Telecommunication Engineering, NMIMS MPSTME, India*
[3]*B. Tech student in Electronics and Telecommunication Engineering, NMIMS MPSTME, India*

Abstract - ***This paper deals with the control of an Unmanned Armed Vehicle with 6 rotors, commonly known as a hexacopter. It is highly unstable and difficult to control due to its non-linear nature. A hexacopter flying machine is underactuated; it has six degrees of freedom and four control signals (one thrust and three torques). The control algorithm is developed using Newton-Euler angles and PID controllers. The mathematical model of the hexacopter is described. The results of the algorithm are obtained as presented in this paper.***

Key Words: **Hexacopter, Mathematical modeling, PID Control, MPU, Arduino Mega, MATLAB Simulation**

1. INTRODUCTION

A lot of research is performed on autonomous control of unmanned aerial vehicles (UAVs). UAVs come in all sizes, fixed wing or rotary, and are used to perform multiple applications. Multirotor is a true pioneer in the UAV market. It is a key influencer and forerunner of one of the most interesting and promising technology trends of the last couple of years. Reasons for choosing hexacopter are its high stability, higher overall payload and greater overall power and flight speed.

A hexacopter aircraft is a type of multirotor and has various characteristics which includes Vertical Takeoff and Landing (VTOL), stability, hovering capabilities and so on. But its advantages come at a cost: The hexacopter has non-linear dynamics and is an underactuated system, which makes it necessary to use a feedback mechanism. An underactuated system has less number of control inputs as compared to the system's degrees of freedom. This gives rise to nonlinear coupling between the four actuators and the six degrees of freedom, thus, making the system difficult to control. A Proportional-Derivative-Integral (PID) controller is used to control the altitude and attitude and for hovering the hexacopter in space. MATLAB simulation based experiments are also conducted to check and evaluate the dynamic performance of the hexacopter using the PID controller. [1]

The main aim of this paper is to derive an accurate and detailed mathematical model of the hexarotor unmanned aerial vehicle (UAV), develop linear and nonlinear control algorithms and apply those on the derived mathematical model in computer based simulations.

The paper organization is as follows: In the following section the mathematical modeling of hexacopter is described while the PID control technique is explained in section III. In section IV, the MATLAB simulation is presented followed by the implementation procedure and results in section V. Section VI gives the conclusion.

2. MATHEMATICAL MODELING

A hexacopter consists of six rotors located at equal distance from its center of mass. Adjusting the angular velocities of the rotors controls the copter. It is assumed as a symmetrical and rigid body and

As we know, the hexacopter has six degrees of freedom and four control inputs. These four control inputs are: Roll (movement along the X-axis), yaw (movement along the Z-axis) and pitch (movement along the Y-axis).

2.1 Reference system

The hexacopter is assumed as a symmetrical and a rigid body and therefore the differential equations of the hexacopter dynamics can be derived from the Euler angles.

Consider two reference frames, the earth inertial frame and the body fixed frame attached to the center of mass of the copter. The starting point of the earth fixed frame is fixed on the earth surface and X, Y and Z axis are directed towards the North, East and down direction respectively and in this frame the initial position coordinates of the hexacopter is defined. The body fixed frame is centered on the hexacopter's center of mass and its orientation is given by the three Euler angles namely; roll angle (ϕ), yaw angle (Ψ) and pitch angle (θ).

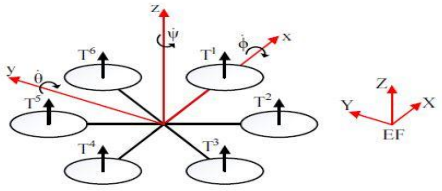

Fig -1: Body fixed frame and earth inertial frame of hexacopter [2]

The position of the hexacopter in the earth frame of reference is given by the vector ξ = [x y z] T and the Euler angles form a vector given as η = [ϕ θ Ψ] T.

The following rotational matrix R gives the transformation from the body fixed frame to the earth frame: [3]

$$R = \begin{bmatrix} \cos\theta\cos\psi & \cos\psi\sin\theta\sin\phi - \cos\phi\sin\psi & \cos\phi\cos\psi\sin\theta + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \cos\phi\cos\psi + \sin\theta\sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - \cos\psi\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix} \quad (1)$$

2.2 Hexacopter Model

The translational and rotational motion of the hexacopter body frame is given as follows: [4]

2.2.1 Translational dynamics

The equations for the translational motion are given as:

$$\ddot{x} = 1/m\,(\cos j\ \cos y \sin q + \sin j\ \sin y)\ U1\text{-}kftx\ \dot{x}/m$$

$$\ddot{y} = 1/m\,(\cos j\ \sin q \sin y - \sin j\ \cos y)\ U1\text{-}kfty\ \dot{y}/m \quad (2)$$

$$\ddot{z} = 1/m\,(\cos q \cos j\ )\ U1\text{-}kftz\ \dot{z}/m\text{-}g$$

2.2.2 Rotational dynamics

The equations for the rotational motion are given as:

$$\begin{aligned} J_{xx}\ddot{\phi} &= \dot{\theta}\dot{\psi}(J_{yy} - J_{zz})\,K_{fax}\dot{\phi}^2 - J_r\Omega_r\dot{\theta} + lU_\phi \\ J_{yy}\ddot{\theta} &= \dot{\phi}\dot{\psi}(J_{zz} - J_{xx})\,K_{fay}\dot{\theta}^2 + J_r\Omega_r\dot{\phi} + lU_\theta \\ J_{zz}\ddot{\psi} &= \dot{\phi}\dot{\theta}(J_{xx} - J_{yy})\,K_{faz}\dot{\psi}^2 + lU_\psi \end{aligned} \quad (3)$$

2.2.3 Total system dynamics

The 2nd order diffrential equations for the hexacopter's position and attitude are thus obtained and can be written as the following equations:

$$\ddot{\phi} = \frac{1}{J_{xx}}[\dot{\theta}\dot{\psi}\left(J_{yy} - J_{zz}\right) - K_{fax}\dot{\phi}^2 - J_r\Omega_r\dot{\theta} + lU_\phi]$$

$$\ddot{\theta} = \frac{1}{J_{yy}}[\dot{\phi}\dot{\psi}\,(J_{zz} - J_{xx})] - K_{fay}\dot{\theta}^2 + J_r\Omega_r\dot{\phi} + lU_\theta]$$

$$\ddot{\psi} = \frac{1}{J_{zz}}[\dot{\varphi}\dot{\theta}(J_{xx} - J_{zz}) - K_{faz}\dot{\psi}^2 + U_\psi]$$

$$\ddot{x} = -\frac{k_{ftx}}{m}\dot{x} + \frac{1}{m}U_x\,U_1 \quad (4)$$

$$\ddot{y} = -\frac{k_{fty}}{m}\dot{y} + \frac{1}{m}U_y\,U_1$$

$$\ddot{z} = -\frac{k_{ftz}}{m}\dot{z} - g + \frac{\cos\phi\cos\theta}{m}U_1$$

where, $U_x = \cos\phi\cos\psi\sin\theta + \sin\phi\sin\psi$

$U_y = \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi$

Table -1: Parameters used in hexacopter dynamics

Jxx	MOI about body frame's x-axis
Jyy	MOI about body frame's y-axis
Jzz	MOI about body frame's z-axis
I	Moment arm
Jr	Rotor inertia
m	Hexarotor mass
Kf	Aerodynamic force constant
Kt	Aerodynamic translational coefficient

3. PID CONTROLLER

PID (proportional-integral-derivative) is a control loop feedback mechanism, which calculates an error that is the difference between an actual value and a desired value. It tries to remove or minimize this error by adjusting the input. PID controller is required in the hexacopter in order to make stable, operative and autonomous. If a PID is not used then on application of a throttle, the hexacopter will reach the desired position but instead of being stable, it'll keep oscillating around that point in space. The reason for using A PID controller over a PI controller is that a PID gives a better response and the speed of response is also greater than that of a PI.

There are three control algorithms in a PID controller, they are P, I, and D. P depends on the present error and reduces the rise time and the steady state error; I depends on the accumulation of past errors and eliminates the steady state error; while D is a prediction of future errors based on the current rate of change. It increases the stability of the system and also improves the transient response.

The main purpose of the PID controller is to force the Euler angles to follow desired trajectories. The objective in PID controller design is to adjust the gains to arrive at an acceptable degree of tracking performance in Euler angles. [1]

The block diagram for a PID controller is shown below:

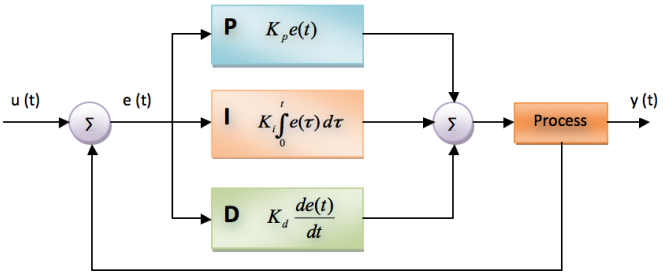

Fig -2: PID Controller Block diagram

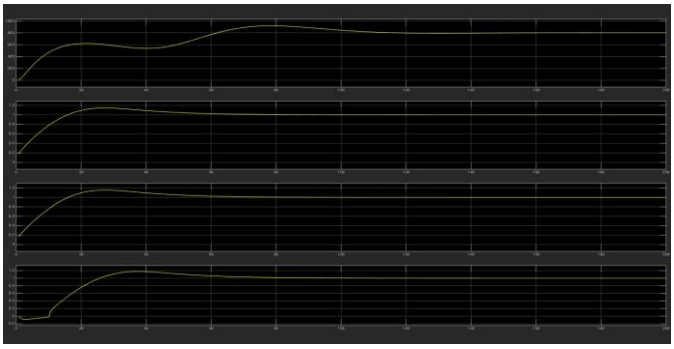

Fig -3: PID Controller Output

Altitude controller:

$$U_1 = k_{p,z}(z - z_d) + k_{d,z}(\dot{z} - \dot{z}_d) + k_{i,z}\int (z - z_d)\,dt \quad (5)$$

Where,

z_d and $\dot{z}_d$: *Desired altitude and rate of change of altitude*

Attitude controller:

The PID controller for the roll (ϕ), pitch (θ) and yaw (ψ) dynamics is given as,

$$U_2 = k_{p,\phi}(\phi_d - \phi) + k_{d,\phi}(\dot{\phi}_d - \dot{\phi}) + k_{i,\phi}\int (\phi_d - \phi)\,dt$$
$$U_3 = k_{p,\theta}(\theta_d - \theta) + k_{d,\theta}(\dot{\theta}_d - \dot{\theta}) + k_{i,\theta}\int (\theta_d - \theta)\,dt \quad (6)$$
$$U_4 = k_{p,\psi}(\psi_d - \psi) + k_{d,\psi}(\dot{\psi}_d - \dot{\psi}) + k_{i,\psi}\int (\psi_d - \psi)\,dt$$

Where,

φ_d and $\dot{\varphi}_d$: *Desired roll angle and rate of change of roll angle*

θ_d and $\dot{\theta}_d$: *Desired pitch angle and rate of change of pitch angle*

ψ_d and $\dot{\psi}_d$: *Desired yaw angle and rate of change of yaw angle*

Position controller:

The control laws are given as,

$$\ddot{x}_d = k_{p,x}(x_d - x) + k_{d,x}(\dot{x}_d - \dot{x}) + k_{i,x}\int (x_d - x)\,dt \quad (7)$$
$$\ddot{y}_d = k_{p,y}(y_d - y) + k_{d,y}(\dot{y}_d - \dot{y}) + k_{i,y}\int (y_d - y)\,dt$$

4. SIMULATION

After tuning the PID and calculating the thrust U1 and control inputs U1, U2, U3 and U4 i.e. yaw, roll and pitch, the next step is to calculate the rotor velocities. This is done using the following matrix (8):

$$\begin{bmatrix} U1 \\ U2 \\ U3 \\ U4 \end{bmatrix} = \begin{bmatrix} b & b & b & b & b & b \\ \frac{-bl}{2} & -bl & \frac{-bl}{2} & \frac{bl}{2} & bl & \frac{bl}{2} \\ \frac{-bl\sqrt{3}}{2} & 0 & \frac{bl\sqrt{3}}{2} & \frac{bl\sqrt{3}}{2} & 0 & \frac{-bl\sqrt{3}}{2} \\ -d & d & -d & d & -d & d \end{bmatrix} \begin{bmatrix} w_1^2 \\ w_2^2 \\ w_3^2 \\ w_4^2 \\ w_5^2 \\ w_6^2 \end{bmatrix} \quad (8)$$

The rotor velocities from to is calculated by taking the inverse of the thrust and control inputs. To obtain the rotor velocities the inverse of the matrix (8) is taken, to give the following set of equations:

$$w_1^2 = \frac{1}{6bl}(lU1 + 2U2 - \frac{bl}{d}U4)$$
$$w_2^2 = \frac{1}{6bl}(lU1 + U2 - \sqrt{3}U3 + \frac{bl}{d}U4)$$
$$w_3^2 = \frac{1}{6bl}(lU1 - U2 - \sqrt{3}U3 - \frac{bl}{d}U4) \quad (9)$$
$$w_4^2 = \frac{1}{6bl}(lU1 - 2U2 + \frac{bl}{d}U4)$$
$$w_5^2 = \frac{1}{6bl}(lU1 - U2 + \sqrt{3}U3 - \frac{bl}{d}U4)$$
$$w_6^2 = \frac{1}{6bl}(lU1 + U2 + \sqrt{3}U3 + \frac{bl}{d}U4)$$

Where b stands for the drag constant, l stands for the length of the arm of the hexacopter. [1]

Table -2: Simulation Parameters and values

Parameters	Values
Jxx	7.5e-3
Jyy	7.5e-3
Jzz	1.3e-2
Kftx	1.5
Kfty	0.1
Kftz	1.9
Kfax	0.1
Kfay	0.1
Kfaz	0.15
b	3.13e-5
d	7.5e-7
Jr	6e-5
Ki	1
Kd	0.1
Kp	1000
Kpphi	0.00139259447453355
Kptheta	0.00139259447453355
Kppsi	0.00241383042252482
Kpz	0.0722761065184604
Kiphi	0.0468148941773645
Kitheta	0.0468148941773645
Kipsi	0.0468148941773645
Kiz	0.0536435875840761
Kdphi	5.24517904929436
Kdtheta	5.24517904929436
Kdpsi	5.24517904929436
Kdz	3.25172322804672

5. EXPERIMENTATION

5.1 Hardware

Arduino- the board used for experimental purposes is Arduino Mega 2560. Arduino Mega 2560 is preferred over

Arduino Uno because experimentation can be done real time using external mode in Matlab Simulink with Mega but not with Uno. Arduino Mega has more number of digital and PWM pins compared to Uno. Six of its PWM pins give the input control signal to start or stop the rotor and to control the speed of the same. It processes the data coming in from the MPU sensor, based on the control algorithm and updates the control signals accordingly.

5.2 MPU

MPU 6050 – It is 6-axis, consisting of a 3-aixs gyroscope and a 3-aixs magnetometer. It is a motion-tracking device.

Raw values:
The angles (degrees) from the accelerometer are calculated using the equations:

```
AccXangle=(float)
(atan2(accRaw(1),accRaw(2))+M_PI)*RAD_TO_DEG
AccYangle= (float)                                    (10)
(atan2(accRaw(2),accRaw(0))+M_PI)*RAD_TO_DEG(conve
rts accelerometer raw values to degrees)
```

The angles (degrees) from the gyroscope are calculated using the equations:

```
rate_gyr_x = (float) gyrRaw(0) * G_GAIN
rate_gyr_y = (float) gyrRaw(1)* G_GAIN                (11)
rate_gyr_z = (float) gyrRaw(2) * G_GAIN
(Converts gyro raw values to degrees per second)
```

```
gyroXangle = gyroXangle + rate_gyr_x*DT
gyroYangle = gyroYangle + rate_gyr_y*DT               (12)
gyroZangle = gyroZangle + rate_gyr_z*DT
```

(gyroXangle is the current X angle calculated from the gyroscope, gyroYangle is the current Y angle calculated from the gyroscope and gyroZangle is the current Z angle calculated from the gyroscope)

Table -3: MPU Parameters

Parameters	Values
DT	20ms
G_GAIN	0.07
M_PI	3.14159265358979323846
RAD_TO_DEG	57.29578
AA	0.98

These are raw values obtained from the MPU. They need to be filtered to overcome gyro drift and accelerometer noise. This filtration can be done using Complementary filter or Kalman filter.
Complementary filter:

Current angle = 98% x (current angle + gyro rotation rate) + (2% * Accelerometer angle)
CFangleX = AA * (CFangleX+rate_gyr_x*DT) + (1 - AA) * AccXangle
CFangleY = AA * (CFangleY+rate_gyr_y*DT) + (1 - AA) * AccYangle
CFangleX & CFangleY are the final angles to use.
The values obtained after filteration are stable values.

5.3 Implementation

The yaw, roll and pitch values are given from the MATLAB Simulink software to the Arduino board. This is done in external mode and not normal mode, since it allows us to change the real time values. Once the real time changes are made in the simulation the MPU senses the actual position of the hexacopter and gives the feedback to the user.
After this the error value between the desired values and the practical values is calculated and the required corrections can be made. Tuning of the 3 control inputs by PID tuning does this correction. At the same time 'z' value is incremented step-by-step using a step input. This sensing and correction is necessary to make a stable hexacopter.

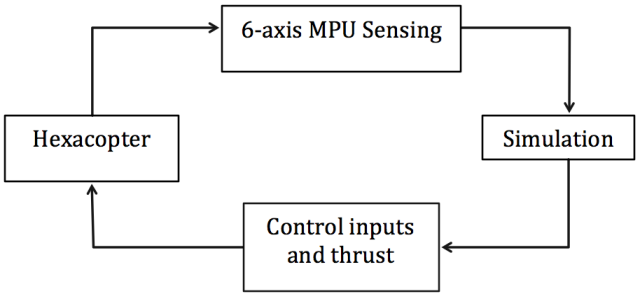

Fig -4: Implementation block diagram

5.4 Result

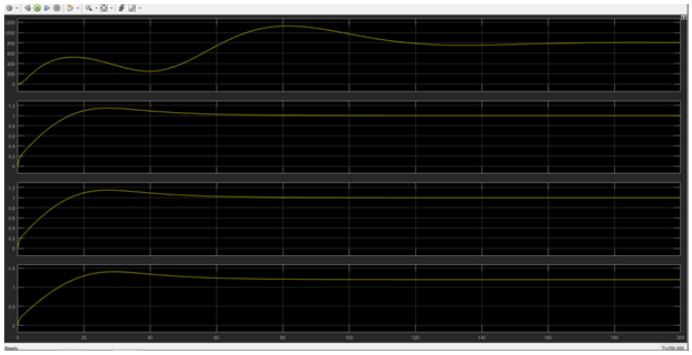

Fig -5: Implementation block diagram

The graphs above show the thrust signal (first graph) that stabilizes after a time delay. Further development is possible to stabilize this signal with a lesser time delay. The

second, third and fourth graphs show phi (yaw), theta (pitch) and psi (roll) signals respectively, that stabilize faster than the thrust signal. Six control inputs are generated using three torques (phi, theta, psi) and one thrust (z) signal. Four PID controllers (each for the thrust and torque signals) used for feedback are tuned to give stable values. The 6-axis MPU mounted on the hexacopter senses its current location, which is required to determine the roll, yaw and pitch to reach the desired location.

6. CONCLUSIONS

In this paper we have presented the mathematical modeling of a highly unstable hexacopter using non-linear dynamics. The main advantage of this paper is that the MPU senses the real time position of the hexacopter, which helps us to gain more stable values of the control inputs. Another advantage is that PID tuning is used to control the control inputs and thrust which allows us to do fine tuning of the inputs for the desired control. When we started off with this project we had an objective of creating a simple and practical UAV. We therefore zeroed in the hexacopter to meet our objective. Hence the initial objective was achieved through this project.

Future students can use our projects as a baseline create advanced version of aerodynamic modeling, which can has more accurate attitude and altitude stability control. This can be done through exhaustive flight controlling methods.

REFERENCES

[1] Mostafa Moussid, Adil Sayouti and Hicham Medromi, "Dynamic Modeling and Control of a HexaRotor using Linear and Nonlinear Methods," in *International Journal of Applied Information Systems (IJAIS),* Foundation of Computer Science FCS, New York, USA, Volume 9 – No.5, August 2015.

[2] Sidea, A-G., Brogaard, R. Y., Andersen, N. A., & Ravn, O. (2015), "General model and control of an n rotor helicopter," Journal of Physics: Conference Series (Online), 570, [052004], doi: 10.1088/1742-6596/570/5/052004.

[3] V. Artale, C. L. R. Milazzo and A. Ricciardello, "Mathematical Modeling of Hexacopter," in *Applied Mathematical Sciences*, Vol. 7, 2013.

[4] Mostafa Moussid, Asmaa Idalene, Adil Sayouti and Hicham Medromi, "Autonomous HexaRotor Arial Dynamic Modeling and a Review of Control Algorithms," in International Research Journal of Engineering and Technology (IRJET), Vol. 2, Iss. 05, Aug 2015.