Republic of Iraq
Ministry of Higher Education
and Scientific Research
University of Babylon
College of Engineering
Mechanical Engineering Department

Quadrotor Adaptive Control Made Easy using LabView and Matlab Software

A Thesis Submitted to the
College of Engineering / University of Babylon in
Partial Fulfillment of the Requirements for the Degree of
Master of Science in / Engineering /Mechanical Engineering
/Applied Mechanics

By

Fatimah Adil Raheemah Ali

Supervised by

Asst. Prof. Mustafa T. Hussein

2022 A.D. **1442 A.H.**

بسم الله الرحمن الرحيم

(إِنَّا فَتَحْنَا لَكَ فَتْحًا مُبِينًا)

صدق الله العظيم

سورة الفتح آية ﴿1﴾

Certification

We certify that this thesis entitled "**Quadrotor Adaptive Control Made Easy using LabView and Matlab Software**" was prepared by **FATIMAH ADIL RAHEEMAH ….** under our supervision at the University of Babylon in a partial fulfillment of the requirements for the degree of Master of Science in Mechanical Engineering (Applied Mechanics).

We recommend that this thesis be forwarded for examination in accordance with the regulation of the University of Babylon.

Signature

Asst. Prof. Mustafa T. Hussein

(Supervisor)

Dedication

To myself

Fatimah 2022

Acknowledgement

I would like to thank everyone at Babylon University. In particular, Dr. Samer, Dr. Bassem, and Dr. Ossama. Thank you to my advisor **Asst. Prof. Mustafa T. Hussein** for the freedom you have enjoyed in my research and for constantly pushing me towards new innovation. Thank you to my family for providing the means for me. To continue my education for their support. Finally, thanks to all the faculty, staff, and graduate students. She taught me how to become an engineer and made me wish I could stay another year.

Thanks, my dad, mam, and my brothers

Thanks my dear Zainab

thanks my family student housing

TABLE OF CONTACTS

LIST OF FIGURES

LIST OF TABLES

Nomenclature

Symbol	Definition
Γ^E	linear position
I_x	Moment of inertia around the x-axis in the body frame (N.m.s^2)
I_y	Moment of inertia around the y-axis in the body frame (N.m.s^2)
I_z	Moment of inertia around the z-axis in the body frame (N.m.s^2)
J_P	Moment of inertia of rotor (N.m.s^2)
K_d	Derivative gain (seconds)
K_p	Proportional gain
R_θ	Rotation matrix
T_θ	Matrix of angular transformations
V^B	Linear velocity
W^B	Angular velocity
Θ^E	Euler angle
$\emptyset$	The orientations of quadrotor in inertial frame along Euler angle; roll (rad)
T	The total thrust (N)
b	Lift coefficient (N.m.s^2)
d	Drag coefficient (N.m.s^2)
g	Gravitational acceleration (m/s^2)
m	The mass of the quadrotor (Kg)
p, q, r	Body angular velocities
u, v, w	Body velocities
x	The position of quadrotor in inertial frame along x-axis (m)
y	The position of quadrotor in inertial frame along y-axis (m)
z	The position of quadrotor in inertial frame along z-axis (m)

Ω	The overall propeller speed [rad. s^{-1}].
μ	Control input
θ	The orientation of quadrotor in inertial frame along Euler angle; pitch (rad)
ξ	The damping factor of the system
φ	The orientation of quadrotor in inertial frame along Euler angle; yaw (rad)
ω	The natural frequency of the system (rad/s)

List of Abbreviation

3D	Three Dimensions
AR	Augmented Reality
CAD	Computer Aided Design
CAE	Computer-Aided Engineering
CAM	Computer-Aided Manufacturing
CATIA	Computer Add Three Interval Application
CMOS	Complementary Metal Oxide Semiconductor
CPU	Central Processing Unit
DOF	Degree of Freedom
GPS	Global Positioning System
GUI	Graphical User Interface
IMC	Internal Model Controllers
IMU	Inertial Measurement Unit
LED	Light Emitting Diode
LQR	Linear Quadrotor Regulation
MRAC	Model Reference Adaptive Control
PID	Proportional Integral Derivative

PLM	Product Lifecycle Management
PTC	Parametric Technology Corporation
ROS	Robot Operating System
UAV	Unmanned Aerial Vehicle
VTOL	Vertical Take-Off and Landing

ABSTRACT

This thesis focused on the study of a quadrotor helicopter. The dynamic system modeling and the control algorithm evaluation are carried out. The dynamic system was modeled using the Newton-Euler formalism. The tests were carried out on a simulated model where the performance could be easily evaluated using a mathematical approach. Also, the work will focus on the advantages of several engineering software to present a case study for quadrotor systems from basic design to motion control. The CAD software is a very powerful tool for model design of dynamical systems, which no needs the derivation of mathematical equations. MATLAB software can receive the CAD model and turn it into a simulated physical model. The CAD model for the quadrotor system in this study is used to show the capability of the 3D model implementation in modeling and control. The control algorithms sliding mode control with Adaptive control and sliding mode control with PID control were also compared. This thesis outlines the technique of identifying and controlling a Parrot AR. Drone. The thesis also is to establish position control movement of the quadrotor (AR. Drone) by using the program LabVIEW. Here, the quadrotor will be controlled in two ways by using the joystick and the use of the keyboard. AR. Drone has been shaped in the hover path and straight path. The simulation results in the dynamic model showed the correct path tracking and the success of the proposed controller unit by using MATLAB/Simulink. Also, it shows the 3D CAD model works fine with the controllers and it preserves the desired position and attitude along the desired predefined path. As well as, prove that sliding mode control with PID control is more efficient than sliding mode control with Adaptive control. The error rate when implementing a helix path of the dynamic model was proved to

be better than CAD model with a difference of 12% for position. While the laboratory results showed the efficiency of using the LabView program in controlling this type of quadrotor .It has been proven that controlling the drone using the keyboard and the joystick gives similar results when comparing the error ratio. And it was concluded when comparing the results of simulation and experiment that the application of the path is better in simulation. Although there is a slight difference in the error ratio, this indicates the success of the experiments.

Chapter One:

Introduction

CHAPTER ONE

INTRIDUCTION

Quadrotor is a term that refers to a helicopter with multiple rotors that is driven and lifted by four rotors. Quadrotors are classified as vertical planes. It is a four-engine aircraft capable that can fly up or down vertically. Also, it has four propellers, where the thrust force is formed by the propellers. The efficiency and technical advantages of quadrotor vehicles are even greater. Instead of a single huge rotor, it has four little propellers. This reduces system torque, allowing the blades to be driven at faster speeds without causing additional mechanical vibrations, while also increasing motor efficiency [1].

During the last two decades an interest in an unmanned aerial vehicle (UAV) increased due to its high maneuverability and flight ability. The quadrotor is gaining popularity and utility in the majority of military and civilian fields. It is utilized for land and sea search work, pipelines, constructions, and flame piers and for aerial mapping, surveying, and mining exploration in the mining sector. In addition, it is utilized to conduct tests on power transmission lines. This system keeps workers away from dangerous circumstances and assists in monitoring the process. The quadrotor is used in danger areas or hard-to-reach areas, in the military fields for espionage purposes. Also, quadrotors now include a variety of equipment on board, including GPS, which enables the quadrotor to correctly determine its location. Application development has also been done to make it easier to use quadrotors, such as setting up waypoints, hovering with position control, and flight trajectory following, among other things [2].

The quadrotor is a dynamically unstable system that has to be stabilized by a suitable control system. However, it is operated autonomously or remotely operated. Basically, an onboard computer or microcontroller and a variety of sensors including accelerometers, gyroscopes, and magnetometers maintain the quadrotor's stability.

1.1. AR. Drone Search Platform

AR. Drone 2.0 model from Parrot, a French manufacturer was used in this study as shown in **figure 1.1.** This type is available to the general public and was chosen for its basic structure. Ample sensory equipment, and low cost of upkeep. These characteristics demonstrate that drone drive is a good subject for both academic and practical purposes. AR. Drone is a UAV (a rotorcraft) that has been sold as a high-tech toy. It was created to be controlled by tablets or smartphones connected to a wi-fi network. Toy quadrotors can be purchased for a few hundred dollars, but they cannot communicate with a computer and lack sophisticated sensors [3]**.**

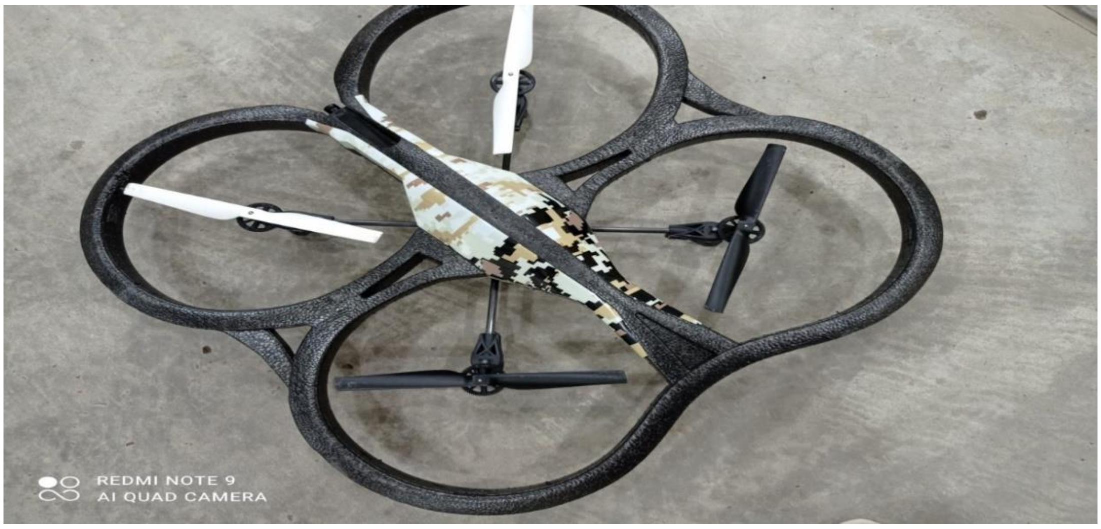


Figure 1.1: AR. Drone 2.0 quadrotor

1.2. Quadrotor VS Other Airplanes

In comparison to other forms of aircraft or copters, quadrotors have a number of advantages. Unlike fixed-wing aircraft, quadrotors do not require a large amount of room to generate lift. Due to the fact that a quadrotor uses four engines rather than a single main rotor, it consumes less kinetic energy per engine for the same amount of thrust as a helicopter. The quadrotor requires fewer repairs than traditional aircraft since it is more maneuverable, has fewer mechanical complications, and can carry a huge payload. Additionally, quadrotors provide advantages such as high mobility, low cost, and simplicity of manufacture, making them an excellent candidate for usage as robotic autonomous devices [4].

1.3. Software used in the thesis.

1- MATLAB 2017 is a high-level programming language for technical computing that can be used for algorithm development, data visualization, data analysis, and numeric computation. Because of its simple interface and strong commands, it’s commonly utilized in research and engineering.

2- LabVIEW 2015 is a graphical programming environment, used to create automated research, validation, and production test systems.

3-Soildwork 2017 provides powerful engineering tools for designing and can simulate a model for analyzing forces and torque in mechanical links with acceleration and displacement mapping of every part of the system.

1.4. Objective of Thesis

AR. Drone has been selected for this research due to its low cost, maneuverability, simple mechanics, and payload capacity. The objectives of this thesis include:

1- Quadrotor identification, system modeling, evaluation of control algorithm design.
2- Simulator design, optimizing the behavior of a model.
3- Study the advantages of several engineering software.
4- Achieve Stabilization of the quadrotor by using program MATLAB and LabView.
5- Achieve that a quadrotor follows a specified path using LabVIEW and MATLAB programs.

Chapter Two:
Literature Review

LITERATURE REVIEW

CHAPTER TWO

The proposed work was compared with the previous studies by reviewing a body of literature. The survey of literature will be discussed as follows:

- Quadrotor modeling
- Control methods

2.1. Quadrotor Modeling

A quadrotor was considered a rigid body in three-dimensional space, it has six degrees of freedom, three of which are translational and three of which are rotational. The first three translational degrees describe the UAV position, which was a simple work, whereas the remaining three degrees describe the UAV orientation, which was a more difficult task with ramifications for the derived model. There are a few methods for determining the orientation of a rigid body in space. The quaternions and Euler angles are the most extensively utilized approaches in aeronautical applications **Michael et al(2010)** [5] .

Muller et al (2011) [6] , demonstrate how a quadrotor can hit a ping-pong ball at a target with the help of an attached paddle. A single quadrotor can juggle a ball, back and forth. A Kalman filter was utilized to calculate the ball state necessary to predict impact situations. Three independent experiments used the algorithm: one drone replicating a ball thrown by a human, two drones striking the ball back and forth, and one drone trying to match on its own. The performance of two vehicles cooperating was demonstrated to be significantly superior to that of single-vehicle acting alone, owing to the vehicles having more time to adapt and begin each trajectory in a more advantageous position.

Hernandez et al (2013) [7], stated in their research the method of identifying and controlling a Parrot AR. Drone was described in closed-loop mode by using program MATLAB. Here, the transfer functions for pitch and altitude movements specified and a comparison between the performance of both simulation and practice implemented PID and internal model controllers were made. The result showed that the IMC overshoot provides a much faster settling time of T < 9s as compared to the PID T ≈ 14s. The comparison shows an increase in performance with internal model control use.

In some literature, instead of searching for parameters, the researchers identified the mathematical model and verified its validity. **Koszewnik (2014)** [8] , stated in this research that the method of modeling and designing control laws for the Parrot UAV's four-rotor form presented. A mathematical model was derived by Newton-Euler's method, then simplification the mathematical model and rewritten. The platform stability was achieved by designing four separate PID controllers that control pitch, roll, yaw, and elevation position, respectively. Here, it was found that the parrot's movement was correctly stabilized by the whole closed-loop system.

Prayitno et al (2014) [9] , stated in their research that AR. Drone 2.0 Elite Edition's fuzzy logic controller designed and installed to follow a certain reference path in space coordinates. LabVIEW software will be used to implement the controller. This algorithm was used to control autonomous flight using a variety of moved for entertainment reasons. The fuzzy logic controller can be utilized to tackle the challenge of trajectory tracking in AR. Drone. As well, the algorithm tested under the variety of paths, including straight lines, straight lines with vertical inflection, rectangular paths, and curved paths.

Civan et al (2015) [10] performances LabVIEW and MATLAB /Simulink on control system simulation will be studied and compared at modeling dc motor by using P, PI, and PID. The influence of the type of controller affects simulation time was examined. As expected, the amount of time spent on simulation increases as the controller's complexity develops. Despite the fact that simulation time increases on both platforms, the speed of increase in LabVIEW is faster. Total simulation time was divided by the number of loops for all controllers to calculate time per loop. It was clear that LabVIEW's simulation time performance outperforms MATLAB/ Simulink's for a relatively small number of loops. When the number of loops was increased, however, MATLAB/Simulink outperforms in terms of simulation time.

The way to control the trajectory was of great importance, **Guo et al (2016)** [11] , the problem of trajectory tracking was addressed, and a control system based on disturbance observer was developed to deal with the impacts produced by time-varying mass, inertia, and another uncertainties. The control diagram was composed of two loops: one for internal attitude control and one for external position control. The simulation results demonstrate the accuracy and speed with which tracking issues may be solved in the presence of a variety of unpredictable parameters, such as time-varying mass and inertia. The disturbance observer-based control scheme was practical in many plants because of its outstanding performance and simple structure.

To achieve stability, quadrotor controller based on a (PID) control method designed. This vehicle's dynamic model will also be explained using the Euler Newton method. MATLAB /Simulink was used to test and analyze the proposed control strategy's performance. The experimental results show that the quadrotor can currently "take-off, hover, and land."**Khusheef et al (2016)**[12]**.**

Holonec et al (2016) [13], stated in their research a self-guided AR. Drone to locate and follow an object inside a limited area. Algorithms can be enhanced and expanded to get more responsive and accurate drone behavior. The applications were developed in LabVIEW utilizing a preconfigured pallet for communicating between a PC and an AR Drone 2.0. The algorithms were created using data from the quadrotor. They denote the angle of rotation about the x-, y-, and z-axes. LabVIEW was used to collect and process data. The drone's photographs were sent to the laptop, where they were analyzed and decoded using AR Drone Toolkit.

Gouasmi et al (2012) [14] stated that SolidWorks and MATLAB /Simulink software were used to verify the theory and simulate robot motion by comparing two robots using equal trajectory and the same time duration, finding dynamic and kinematic parameters by generating computer code. The results obtained, whether using SolidWorks or MATLAB/Simulink, are exactly the same. This similarity of results confirms the reliability of the kinematic model.

CAD designs can greatly accelerate the process of construction and updating the model of a plant used in simulation studies. This technique can help to reduce the number of bugs when design changes happen, as they are routinely implemented rather than by hand. For this reason, CAD data is used to study quadrotors. The model builted through converting the CAD data into XML data so that it can be read by using a physical modeling tool, in which the quadrotor system dynamics defined instead of using the derivation of differential equations **Ryan et al (2013)** [15]**.**

Elsamanty et al (2013) [16], stated in their research that using SolidWorks to create a CAD model that calculated the moment of the inertial block as well as all missing geometrical parameters. The stability of the quadrotor position tested by designing a control system based on the technique of linear feedback. controllers executed on Arduino Mega 2560 MCU connected to MATLAB. Two distinct test rigs are built to verify the quadrotor's identified parameters and attitude estimation. The experiments show that the quadrotor-identified parameters were accurate enough that the quadrotor can be stabilized efficiently using the feedback linearization approach.

Modeling and testing programs for quadrotor drones was used, automated navigation systems developed and the UAV trajectory planned. A simulation model was built using Solidworks as a CAD program, As examples of simulation models, a laboratory truck crane and a forest crane have been shown. then the CAD model was executed and exported to Matlab/Simulink **Cekus et al (2014)** [17] by using Simscape multibody shown in **figure 2.1**.

Shaqura et al (2017) [18] a user-friendly interface used to create quadrotor design, modeling, and simulation. For model validation, simulation, and control creation, the tool based on the SolidWorks application program interface and MATLAB Simscape toolbox. Developers can use the SolidWorks application program interface to construct a standalone application that takes advantage of their engineering capabilities and used CAD that strengthens the relationship between simulations and the actual system. The application was built with a user-friendly interface to attenuate the difference between quadrotor simulations and actual experimentation The software generates the 3D model, identifies its properties, and creates the MATLAB simulation environment files based on specified specifications.

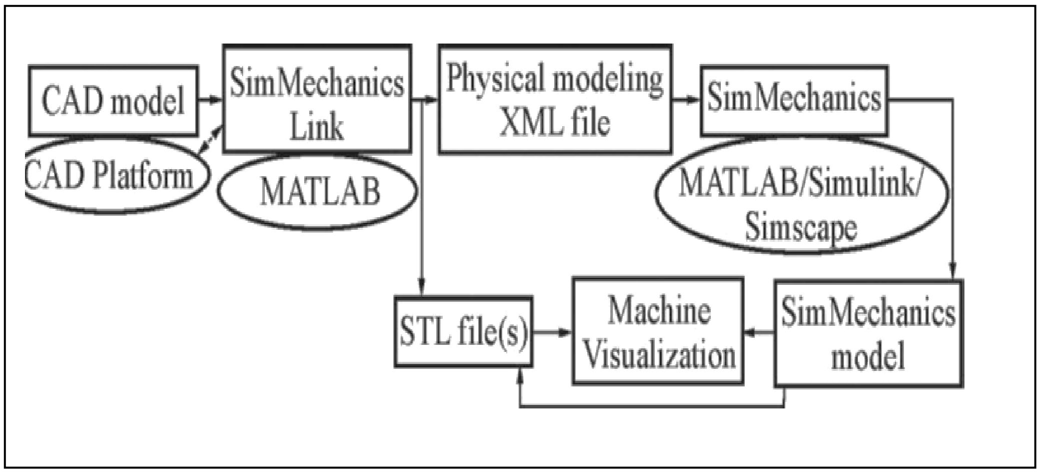


Figure 2.1: Matlab/Simulink implementation process for the CAD model [17].

Simone et al 2017 [19]**,** a simplified model of the quadrotor was built using the Simscape Multibody environment. The simplified quadrotor model was an underacting, tightly interconnected nonlinear system. The dynamics take into account the aerodynamic effect of drag, the push owing to a propeller rotation, and external wind fields. Linear PID controllers were then constructed in the presence of the wind fields for feedback control.

Zátopek et al (2018) [20] stated that a CAD model created for two main purposes. First: developing a mathematical model for the design of a particular controller. second: creating a model that can be exported to MATLAB/Simulink (Simscape/SimMechanics) where the modeling process was facilitated by taking advantage of the properties of the combination of MATLAB and SolidWorks.

Grau et al (2019) [21]**,** due to the system's nonlinearity, a linear control model suggested to create a global model that operates in all operating ranges and used a linear time-invariant controller. CAD techniques applied to the parameters of these linear models. Thus the geometric and physical qualities of the components obtain the dynamic parameters. The linear model obtained has enough "quality" and can be used to improve the quadrotor controller.

Ali et al (2020) [22] stated in their research that a dynamic model and control simulation of a quadrotor aircraft were created in order to create a control algorithm for quadrotor flying and low-speed flight circumstances, as well as to evaluate the controller's performance. The quadrotor model implemented in the X-Plane flying simulator, as well as the LabVIEW console was implemented. Furthermore, it was determined that the developed controllers required further fine-tuning of the gain values in order to decrease rise and settling time and overshoot.

Santana et al (2020) [23] researchers present a system for implementing autonomous landing on a drone by using program MATLAB . The image frames are processed by an indicator recognition package, which detects the AR marker that serves as a landing pad and computes the position. Also using PID controller to control the speed of AR. Drone, and using A laptop computer running Ubuntu and ROS. The resilience of this control system was demonstrated in practice, with challenges such as large wind gusts and nonmodeled external forces successfully overcome.

Velasco et al 2020 [24] , stated in their research that a simulation technique developed for designing and testing of control devices that would coordinate and interact with unmanned vehicles .V-REP and MATLAB/Simulink were used in the simulation strategy because they offer a wide range of features, modules, and cross-platform interoperability. V-REP was based upon Gazebo and has a variety of advantages for implementation. It can import CAD models, grid processing, and editing tools that can quickly be converted to effective and useful robotic models. The results reveal that by offering an off-the-shelf simulation environment and step-by-step implementation directions, this strategy dramatically decreases development and delivery timelines.

Idrissi et al 2021 [25] stated in their research that increasing the flight performance of the quadrotor, a quadrotor platform with replaceable chassis was developed. Two plates containing the block of electronic components were added at the same time as the quadrotor. The overall structure of the vehicle is similar to a conventional quadrotor engine in terms of characteristics and features. During the flight, the two plates can move horizontally back and forth. To test performance capabilities and assure stability, PID controllers were designed and implemented on the quadrotor model. The behavior demonstrates that the quadrotor was able to reach the end position faster while being stable than the conventional model.

CAD program distinguished by its capacity to produce powerful results that mimic any dynamic system while taking into account its geometry, mass, and inertia. So by implementing a CAD model that was previously produced in SolidWorks and exported to MATLAB/Simulink, it has been described the integration process of SolidWorks and MATLAB/Simulink environments. Also, implemented an algorithm based on PID control strategies in order to evaluate the efficiency of SolidWorks and MATLAB/Simulink **Jatsun et al (2021)** [26]**.**

2.2. Control methods

A backstepping control was offered with the Lyapunov stability theory to guide the design process to steady the entire system. So **Madani et al (2006)** [27] creates a dynamic, non-linear quadrotor model for the rear control design. Diverse model simulations show that a quadrotor with adequate tracking is stabilized in the control law.

Lee et al (2009) [28], stated in their research that the linear feedback controller was derived to simplify the dynamics of quadrotors and in a traditional way. The linear feedback controller was not robust to uncertainty as well as sensor noise. Then the adaptive sliding mode controller was used which is more robust in facing the noise. so that the uncertainty here was caused by the ground effect and was compensated.

Das et al (2009) [29], proposed the use of dynamic inversion to overcome the ineffective system of coupling in a quadrotor dynamic. Dynamic reflection applied to the inner loop and which results in therefore not necessarily stable inner dynamics. The internal dynamics were stabilized through a robust control method (feedback linearized) rather than altering the output control variables to achieve stability. This results in an internal ring for dynamic inversion and an external ring for stabilization of internal dynamics. A Lyapunov-type proof is used to ensure stability and tracking performance.

Li et al (2012) [30], The comparison was made using three non-linear control methods, namely backstepping control , sliding mode control ,and feedback linearity control. They were tested on a quadrotor (Qball-X4). It has been observed that the best performance and best control was the sliding mode control due to its ability to handle uncertainty and the feedback linearity control was slightly weaker than the sliding mode control. Backstepping control can be used when axes need to be separated.

Dydek et al (2012) [31] , An internal flight test was conducted using direct and indirect model reference adaptive control depending on stability (Lyapunov), where it was concluded that combined model reference adaptive control (was an adaptive controller that combines direct and indirect adaptive control) was more effective in learning the true value of the parameters and improves more performance quadrotor from notes that the difference between overshoots was 3.7 % for the MRAC controller and 0.1 % for the CMRAC. It was also allowing safe operation in the event accrues of a failure in thrust.

It was observed that quadrotor control was a quite challenging due to its nonlinear properties and unstable dynamics. So in **Khalifa et al (2013)** [32] Specifically, the researchers developed two types of controllers: the Direct Fuzzy Logic controller(DFLC) and the Fuzzy Model Reference Learning Control(FMRLC). Also, evaluated them the algorithm's capacity to stabilize the system and monitor the required pathways while the payload was being picked and placed, and altering the operating system area. The results show the (DFLC) fails in making the system stable while, and (FMRLC) success when operating an area transition.

Ceren et al (2017) [33] ,designed both SMC and PID models are created in MATLAB Simulink. Altitude, roll, pitch, and yaw angle controls are all controlled with sliding mode controllers. Following that, a proportional integral derivative (PID) controller is created. According to simulation result, sliding mode controllers may track desired values faster and with less oscillation than PID controllers.

The most common nonlinear control methods for quadrotors are linearizad feedback, Sliding mode, and backstepping. **Fethalla et al (2018)** [34] stated in their research that the backstepping approach with control of the sliding mode used to track the trajectory of the quadrotor and to offset the impact of the unparalleled uncertainties affecting the system during flight. The stability of the quadrotor will be designed by taking the idea of sliding mode controller with a Nonlinear Disturbance Observer (NDO). The outcome of the laboratory experiment shows that in the presence of external winds, the designed controller provides good traceability.

Control methods have been used to achieve better accuracy and the PID control method is the most popular method. **Miranda et al (2020)** [35] , stated in their research that the proportional integral derivative (PID) was used with the energy reduction methodology to control the quad-rotor, so that, the control scheme was compared with non-linear sliding mode control, and the superiority of the control scheme and its durability against different types of disturbances was proven. In addition, reducing the energy by using the power reduction method.

Cardeira et al (2021)[36], the researchers designed a control structure consisting of a feedback linearization method and an LQR method. The control structure was implemented on a quadrotor. The control solution structured in two loops, an outer ring for horizontal position control, and an inner ring for controlling the attitude and altitude dynamics, and the strategy was validated by the results. It was observed that performance improved and overcame turbulence and uncertainty.

Isaac et al (2021) [37]**,** to increase the control performance, particularly the system's stability, five distinct control systems were presented. that contain a typical PID controller, two Fuzzy-PD controllers, and two controllers autotuned using a genetic algorithm are created based on the proposed dynamic model, and their performance compared. A simulator was also created with the goal of characterizing the quadrotor's motions. The results revealed the PID controller's limitations under non-linear settings, and the Mamdani Fuzzy-PD produced the greatest results with the majority of the difficult trajectories, and it did not necessitate the use of a Genetic Algorithm to improve its performance.

2.4. Closing Remark

- The new contribution in this work is the dynamic model which is especially executed on AR. drone compared to published research that it is executed in general on quadrotor such as. This point was studied in the current work is up to date and has not been studied before in this manner.

- The derived mathematical model is compared with CAD AR. Drone model to verify the CAD model usefulness through simulation.
- Several flight tests experiments have been carried out for the quadrotor using different methods. This thesis offers a point of start for more advanced applications involving automated control of an AR. Drone quadrotor.
- The experiments were carried out on the drone (2018) platform and using one of the modern programs LabVIEW. Based on the literature survey, the use of this program for research is few because of its recentness. Table 2.1 explain comparison between literature review closer to the work presented.

Table2.1: Comparison between literature review

Ref	Control algorithm	Software	Aim
[9] 2016	FLC	LabVIEW	The fuzzy logic controller of drone 2.0 Elite Edition developed and installed to follow a specified reference path in space coordinates.
[13] 2016		LabVIEW	Experiment: Studies the feasibility of implementing a self-guided drone using image processing and data acquired from its onboard sensors
[38] 2018	SMC NDO	SoildWork MATLAB/Simulink	The backstepping approach with sliding mode control be used in conjunction with a nonlinear disturbance observer to offset the impact of unprecedented uncertainties affecting the system during flight, and the quadrotor be designed as a sliding mode controller (SMC) with a nonlinear

			disturbance observer (NDO).
[22] 2020	PID	LabVIEW	a dynamic model and control simulation of a quadrotor aircraft were created in order to create a control algorithm for quadrotor flying and low-speed flight circumstances.
[23] 2020	PID	MATLAB /Simulink	it was using the program MATLAB to control and implement the hover path in AR. Drone.
[25] 2021	PID	MATLAB /Simulink	to increase the flight performance of the quadrotor, a quadrotor platform with replaceable chassis was developed. Two plates containing the block of electronic components were added at the same time as the quadrotor.

Chapter Three:

Theoretical Work

THEORETICALWORK

CHAPTER THREE

An introduction about CAD and its types have been clarified. The derivation of the quadrotor model is presented in this chapter. The four motor speeds, equations can be used to characterize and forecast the positions attained by the quadrotor. The control algorithms used are also designed.

3.1. 3D CAD model for a quadrotor

In this section, 3D CAD and its types are discussed. The physical system is a quadrotor system that has a highly nonlinear and coupled mathematical model. By using the traditional methods of modeling which complicated the modeling process and control design. The CAD model can be designed using the real dimensions and properties of the system. This modeling method will give nearly exact real system parameters and simplify the control design process.

3.1.1. 3D CAD systems

CAD is the abbreviation for computer aided design. It contains all software for creating and editing product geometry, 3D spatial models, and development documents simply and efficiently. For particular purposes such as simulation and force analysis, many CAD programs are extended by different functions. When designing or changing parts, the key benefit of CAD applications is the considerable time savings in processing the drawings themselves. AutoCAD, Inventor, SolidEdge, SolidWorks, CATIA, Pro/Engineer are examples of typical applications [39].

3.1.1.1. Product Dassault Systems

I. Solidworks

SolidWorks provides powerful engineering tools for designing and can simulate a model for analyzing forces and torque in mechanical links with acceleration and displacement mapping of every part of the system. The goal of SolidWorks is to create 3D models drawing from each part including engineering data [39].

II. CATIA

CATIA represents a system of the highest class in the field of CAD / CAM / CAE technologies. It offers the tools to cover the entire product development process. From concept to custom fabrication, through various analyses, simulations, and optimizations, with the ability to make immediate adjustments to the product based on all development documentation programs for self-production [39].

3.1.1.2 Solid Edge

Solid Edge is a Siemens PLM software suite's mid-range 3D CAD device. It has an advanced synchronous technology that enables the processing of parametric models with direct modeling. This functionality allows users to operate concurrently or switch from rival systems and ease interoperability with other CAD/CAM/CAE systems. It is built so that users who are used to only build in a 2D environment can easily learn and adapt to 3D [39].

3.1.1.3 Parametric Technology Corporation (PTC)

Pro / Engineer Wildfire is a modular framework that uses structural elements to create complete associative parametric volumetric modeling. It includes a wide range of parallel development methods, from the design phase of industrial design to concept development, comprehensive plan, analysis and optimization, production documentation development, and testing. It's used to create complex engineering modules and improve and sustain growth in various fields (aerospace, designing machine production, automobile, etc.) [39].

3.1.2 Simmechanics Quadrotor Model

SimMechanics is Simulink physical modeling, which entails modeling and developing structures based on fundamental physical principles. The physical simulation environment and interfaces in Simulink work in conjunction with the rest of Simulink and MATLAB. In contrast to other Simulink blocks that represent mathematical processes or signal processing, physical modeling blocks are used to visually represent physical components or connections [40]. Also, SimMechanics is an environment for designing, modeling, and simulating rigid body machines and their motions that employs standard newton dynamics of forces and torques. SimMechanics enables you to model and simulate mechanical structures using various methods to determine bodies and their mass properties and systems for coordinating and initiating body motions and measuring them [40].

SimMechanics provides two options for machine animation and visualization. One is the integrated handle graphics tool, which uses standard handle graphics equipment known from MatLab with some special functions specific to SimMechanics. The machine bodies can be described as convex hulls. A second choice is to represent the bodies as similar ellipses [41]. Models by SimMechanics allow a range of characteristics to be determined, such as a trajectory, speed, and acceleration, by adding measuring instruments of every element of a complicated system. MATLAB/Simulink environment sometimes does not provide features appearing in many simulation models. In such circumstances, individual scripts and functions might be created to emulate these components [17].

A SimMechanics model describes a system as radically different from other Simulink models.

First, an ordinary model from Simulink is the mathematics of the motion of a machine, that is, of the difference and algebraical equations that predict the future state of the device from its present form. Simulink can simulate the computer with a mathematical model. Second, a SimMechanics model describes a machine's physical structure, its mass properties, and its component geometric. This systemic representation is converted into an inner mathematical model by SimMechanics software [42].

3.2. Basic Concepts

The quadrotor system has three DOF in terms of translation and three DOF in terms of rotation. Additionally, the quadrotor has four propellers so quadrotors do not have a tail rotor because the power of each propeller can be canceled through the distinctive design of the quadrotor. The propellers form the thrust force so an increase or decrease in the speed of the four propellers in the quadrotor will cause a change in the altitude in the quadrotor, velocity, and position [43].

The difference in the left and right rotor speed creates the rolling movement in the opposite direction. The disparity between front and rear speeds results in a pitch movement in the opposite direction. The change of pace in the opposite direction of each rotating pair results in a yaw movement. This enables left and right turning [3]. Where follows are a description of each movement.

1- Throttle:

Throttle movement is accomplished by uniformly increasing (or reducing) the speeds of all propellers. This provides a vertical force along the z-axis relative to the B-frame, which results in the quadrotor being raised or lowered. The throttle movement is depicted in **figure 3.1(a)** [3].

2- Roll :

Roll movement is accomplished by increasing (or reducing) the left propeller's speed while decreasing (or increasing) the right propeller's speed. This speed differential generates torque in the x-axis relative to the B-frame, which causes the quadrotor to revolve and accelerates the roll angle. The roll movement is depicted in **figure 3.1(b)** [3].

3- Pitch:

Similar to Roll movements, pitch changes are accomplished by increasing (or decreasing) the speed of the back propeller and decreasing (or increasing) the speed of the front propeller. This difference in speed provides torque in the y-axis relative to the B-frame, which causes the quadrotor to turn and accelerates the pitch angle. The pitch movement is depicted in **figure 3.1(c)** [3].

4-Yaw :

Yaw is accomplished by increasing (or decreasing) the speeds of the front and back propellers while decreasing (or increasing) the speeds of the right and left pairs. This generates torque in the z-axis relative to the B frame, which causes the quadrotor to revolve and accelerates the yaw angle. The yaw movement is depicted in **figure 3.1(d)** [3].

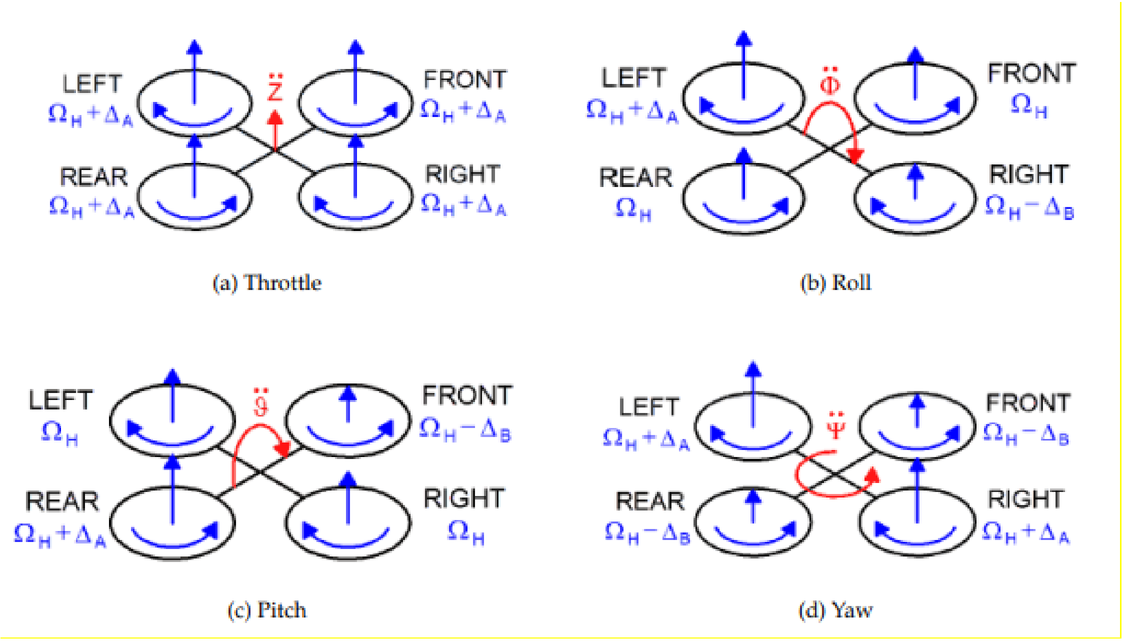


Figure 3.1: Motions of quadrotor[3]

3.2.1. Newton-Euler model

Movement of a 6-DOF rigid body is characterized by two reference frames, first Earth inertial (E-frame); a second, Body-fixed reference(B-frame) [44]. As described in **figure 3.2**

The kinematics of 6-DoF quadrotor

$$[\,\Gamma^E \quad \Theta^E\,]^T = J_B^E [V^B \quad W^B]^T \tag{3.1}$$

Where Γ^E is the linear position, Θ^E is Euler angle. The quadrotor's linear position Γ^E and Euler angles (Θ^E) are specified using the E-frame in equation [44].

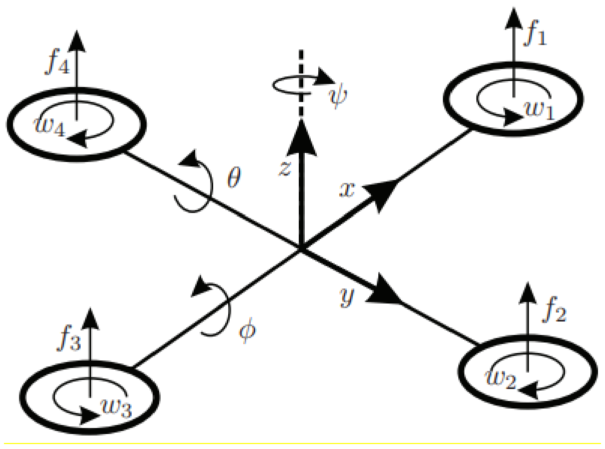


Figure 3.2: Quadrotor system.

$$\Gamma^E = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \quad \Theta^E = \begin{bmatrix} \emptyset \\ \theta \\ \varphi \end{bmatrix} \tag{3.2}$$

Where the center of gravity location in the E-frame is x, y, z , while $\emptyset, \theta, \varphi$ are Euler angles in the E-frame that represent roll, pitch, yaw angles in the E-frame. The quadrotor's linear velocity V^B and angular velocity W^B are specified using the B-frame [44].

$$V^B = \begin{bmatrix} u \\ v \\ w \end{bmatrix}, W^B = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3.3}$$

Where u, v and w are the center of gravity's velocity in B-frame, while p, q and r are the angular velocity in B-frame around x, y and z axes, respectively. And direction of the quadrotor is determined by the rotation matrix, which is specified by the following equation[44]

$$R_\theta = \begin{bmatrix} C_\varphi C_\theta & -S_\varphi C_\emptyset + C_\varphi S_\theta & S_\varphi \, S_\emptyset + C_\varphi \, S_\theta \, C_\emptyset \\ S_\varphi \, C_\theta & C_\varphi C_\emptyset + \; S_\varphi S_\theta \; S_\emptyset & -C_\varphi \, S_\emptyset + S_\varphi S_\theta \, C_\emptyset \\ -S_\theta & C_\theta \; S_\emptyset & C_\theta C_\emptyset \end{bmatrix}, \tag{3.4}$$

$$T_\theta = \begin{bmatrix} 1 & S_\emptyset t_\theta & C_\emptyset t_\theta \\ 0 & C_\emptyset & -S_\emptyset \\ 0 & \frac{S_\emptyset}{C_\theta} & \frac{C_\emptyset}{C_\theta} \end{bmatrix} \qquad (3.5)$$

where T_θ is a matrix of angular transformations, where S_φ refers to $sin(\varphi)$ and C_φ refers to cos (φ) t_θ refers to $\tan(\theta)$[44].

I. Quadrotor equation of Motion

Newton's second law of motion can be used to express the translational motion equation as follows:

$$m\ddot{\Gamma}^E = F^E \qquad (3.6)$$

And the rotational motion equation:

$$I\dot{\omega}^B + \omega^B \times (I\omega^B) = \tau^B \qquad (3.7)$$

Here m is the mass of quadrotor in kg , I is the inertia matrix in [N. m. s^2]. The acceleration in translation and rotation is represented by $\ddot{\Gamma}^E$, and $\dot{\omega}^B$ respectively [44].

II. Forces

With respect to the body E-frame, the thrust and gravity forces acting on the quadrotor body are combined[44].

$$F^E = \begin{bmatrix} s_\psi s_\phi + c_\psi s_\theta c_\phi \\ -c_\psi s_\phi + s_\psi s_\theta c_\phi \\ c_\theta c_\phi \end{bmatrix} \mu - \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} \qquad (3.8)$$

and

$$\mu = k(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_3^2) \qquad (3.9)$$

Here $F^E$is the total of all forces produced by the four motors. Where k is thrust factor in [N.s^2], Ω_1 is the front propeller's speed, Ω_2 is the right propeller's speed, Ω_3 is the rear propeller's speed and Ω_4 is the left propeller's speed [45].

III. Torque

There are two major sources of torque in a quadrotor. Where torques are produced by the principal movement inputs and gyroscopic effects, where $\boldsymbol{\tau}^B$ is defined as[44]

$$\boldsymbol{\tau}^B = -J_P\,\Omega \begin{bmatrix} -\mathrm{q} \\ \mathrm{p} \\ 0 \end{bmatrix} + \begin{bmatrix} \tau_\emptyset \\ \tau_\theta \\ \tau_\varphi \end{bmatrix} = \begin{bmatrix} -J_P\,\Omega\mathrm{q} + \tau_\emptyset \\ J_P\,\Omega\,\mathrm{p} + \tau_\theta \\ \tau_\theta \end{bmatrix} \tag{3.10}$$

Where J_P is the entire rotating moment of inertia in [N. m. s^2] around the propeller axis. And

$$\begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} kl(\Omega_2^2 - \Omega_4^2) \\ kl(\Omega_3^2 - \Omega_1^2) \\ d(\Omega_2^2 + \Omega_4{}^2 - \Omega_1{}^2 - \Omega_3{}^2) \end{bmatrix} \tag{3.11}$$

Where d is the drag factor in [N.m.s^2] and l is the distance between the propeller's center and the motor quadrotor's center [m] [45].

V. Combine dynamic equations

It is possible to represent the linear and angular dynamical system of the quadrotor traveling in frame E as follows, using equations (3.6-3.11). [44]

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \frac{1}{m} \begin{bmatrix} s_\psi s_\phi + c_\psi s_\theta c_\phi \\ -c_\psi s_\phi + s_\psi s_\theta c_\phi \\ c_\theta c_\phi \end{bmatrix} \boldsymbol{u} - \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{3.12}$$

$$\begin{bmatrix} \ddot{\emptyset} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} \left(\frac{I_y - I_z}{I_x}\right)\dot{\theta}\dot{\psi} \\ \left(\frac{I_z - I_y}{I_y}\right)\dot{\phi}\dot{\psi} \\ \left(\frac{I_x - I_y}{I_z}\right)\dot{\phi}\dot{\theta} \end{bmatrix} + \begin{bmatrix} \frac{J_p}{I_x}\dot{\theta} \\ \frac{J_p}{I_y}\dot{\phi} \\ 0 \end{bmatrix}\Omega + \begin{bmatrix} \frac{k}{I_x} & 0 & 0 \\ 0 & \frac{k}{I_y} & 0 \\ 0 & 0 & \frac{d}{I_z} \end{bmatrix}\begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} \tag{3.13}$$

where the inputs are defined as:

$$\begin{bmatrix} u \\ \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ -l & 0 & l & 0 \\ 0 & l & 0 & -l \\ 1 & -1 & 1 & -1 \end{bmatrix}\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} \tag{3.14}$$

The overall propeller speed is defined by the equation below [rad. s^{-1}].

$$\Omega = \Omega_1 - \Omega_2 + \Omega_3 - \Omega_4 \tag{3.15}$$

3.3. Controller design

I. Trajectory tracking control design methodology

Equations (3.12) and (3.13) are part of the quadrotor dynamic discussed in the previous section, it stands for quadrotor motion equations. The equations that described quadrotor dynamics, variables affecting quadrotor motion are $[x \quad y \quad z \quad \phi \quad \theta \quad \psi]^T$ and four input variables for the control system are $[u_c \quad \tau_\phi \quad \tau_\theta \quad \tau_\psi]^T$. The dynamic equations show a tight link between the attitude variables and the state of the system under actuation. The inner-loop and outer-loop structure are designed to be flexible for under-actuation. The term "outer-loop" indicates the position control loop in this thesis, whereas "inner-loop" indicates the attitude control loop [45]. The outer control loop's position control error is defined as

$$p_e = p_c - p \tag{3.16}$$

Where $p_c = (x_c, y_c, z_c)^T$ is the commanded vector's position. Then create the position closed loop equation in the following manner [46]:

$$\ddot{p}_e + K_d \dot{p}_e + K_p p_e = 0 \tag{3.17}$$

Here, K_d and K_p are positive definite controller gain matrices. Assuming the input vector for the position control has form $\ddot{p} = (U_1, U_2, U_3)^T$ [46]. Equation (3.12) is rewrite as:

$$\begin{pmatrix} c_\theta c_\psi & c_\theta s_\psi & -s_\theta \\ s_\theta c_\psi s_\phi - s_\psi c_\phi & s_\theta s_\psi s_\phi + c_\psi c_\phi & c_\theta s_\phi \\ s_\theta c_\psi c_\phi + s_\psi s_\phi & s_\theta s_\psi c_\phi - c_\psi s_\phi & c_\theta c_\phi \end{pmatrix} \begin{pmatrix} U_1 \\ U_2 \\ U_3 + g \end{pmatrix} = \frac{1}{m} \begin{pmatrix} 0 \\ 0 \\ T \end{pmatrix} \tag{3.18}$$

The pitch and roll angles can be modeled as a function of the position controller inputs after a few simplifications and assuming that the yaw angle remains constant [46].

$$\begin{aligned} \theta_c &= \arctan\left(\frac{U_1 \cos\psi_c + U_2 \sin\psi_c}{U_3 + g}\right) \\ \phi_c &= \arcsin\left(\frac{U_1 \sin\psi_c - U_2 \cos\psi_c}{\sqrt{U_1^2 + U_2^2 + (U_3 + g)^2}}\right) \end{aligned} \tag{3.19}$$

The quadrotor system's position is controlled by the outer-loop controller, it incorporates a PID controller with relationships equations for inverse kinematics (3.16-3.19). The quadrotor's attitude is then controlled with the help of a second PID as shown in [46] .Continuously, the required total thrust T can be determined as follows using equations. (3.12 and 3.18):

$$\begin{aligned} T = m[&U_1 (\sin\theta \cos\psi \cos\phi + \sin\psi \sin\phi) \\ &+ U_2 (\sin\theta \sin\psi \cos\phi - \cos\psi \sin\phi) + (U_3 \\ &+ g)\cos\theta \cos\phi] \end{aligned} \tag{3.20}$$

Moreover, based on the regulated values from (3.13), the needed torques may be defined as [45].

$$\begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix}_d = \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix} \begin{bmatrix} 1 & 0 & -S_\theta \\ 0 & C_\phi & S_\phi C_\theta \\ 0 & -S_\phi & C_\phi C_\theta \end{bmatrix} \begin{bmatrix} \ddot{\phi}_d \\ \ddot{\theta}_d \\ \ddot{\psi}_d \end{bmatrix} \tag{3.21}$$

II. Adaptive Sliding Mode Control

$$\dot{x} = f(x) + g(x)u + \Delta \tag{3.22}$$

here: $x \in R^n$ is the state vector, $u \in R^m$ is the system control vector, $m = n$,with $\Delta \in R^n$ is the unidentified vector that represents the modelling error. $f(x)$ and $g(x)$ are continuously differentiable [47] . Where the vector of the error state is $e = x - x_r$. The sliding surface's first derivative can be defined as:

$$\dot{s} = \dot{e} = \dot{x} - \dot{x}_r = f(x) + g(x)u + \Delta - \dot{x}_r \tag{3.23}$$

In order to let the $\dot{s} = 0$, let us propose the following control input,

$$u = g^{-1}(x)(-k_n s - f(x) - \hat{\Delta}) \tag{3.24}$$

Where k_n is a positive definite diagonal matrix. Substitute the proposed control input vector with the following shape for the first derivative of the sliding surface[47] .

$$\dot{s} = -k_n s + \Delta - \dot{x}_r - \hat{\Delta} = -k_n s + \tilde{\Delta}, \tag{3.25}$$

Where $\hat{\Delta}$ is online estimated vector., assume the $\tilde{\Delta} = \Delta - \dot{x}_r - \hat{\Delta}$, then construct the adaptive dynamic vector as follows:

$$\dot{\tilde{\Delta}} = \dot{\Delta} - \ddot{x}_r - \dot{\hat{\Delta}} = -\lambda s, \tag{3.26}$$

where λ is the definite diagonal gain matrix that is positive.

If the appropriate adaptation rules (3.26) are applied in conjunction with the suggested control input (3.22), the affine nonlinear system's sliding surface stated in (3.25) will converge to zero as time increases.

Lyapunov candidate function is defined as

$$V = \frac{1}{2}\, s^T s + \frac{1}{2}\, \tilde{\Delta}^T \lambda^{-1} \tilde{\Delta}. \tag{3.27}$$

The first derivative of the Lyapunov candidate function is defined as

$$\dot{V} = s^T \dot{s} + \tilde{\Delta}^T\, \lambda^{-1} \dot{\tilde{\Delta}}, \tag{3.28}$$

By expanding the preceding equation, we obtain,

$$\dot{V} = s^T \dot{s} + \tilde{\Delta}^T\, \lambda^{-1} \dot{\tilde{\Delta}} = s^T(-ks + \tilde{\Delta}\,) + \tilde{\Delta}^T \lambda^{-1}(-\lambda s) = -s^T k s < 0 \tag{3.29}$$

So it can be stated that the sliding surface converges to zero as time increases, yield the error states goes to zero [47].

The trajectory and yaw directives are included in the command signals. The trajectory command is equivalent to the positions x_c, y_c and z_c. $\psi_c$is the yaw command. The relationship between the attitude angle and the position derivative can be expressed as follows:

$$\begin{aligned} \varphi_c &= (\ddot{x}\sin\,\psi - \ddot{y}\cos\,\psi)m/F_z \\ \theta_c &= (\ddot{x}\cos\,\psi + \ddot{y}\sin\,\psi)m/F_z \\ F_{zc} &= (\ddot{z} + g)m + \hat{\Delta}_1 \end{aligned} \tag{3.30}$$

Where the predicted $\hat{\Delta}_1$ is online estimated to adapt the simplified model inversion deficiency according to formula (3.26). The second derivative of variable z can be treated as the first derivative of variable $\dot{z}$ in equations (3.30) [47] .

The sliding surface:

$$s_1 = \ddot{z}, \tag{3.31}$$

The adaptive sliding mode control is

$$\dot{\hat{\Delta}}_1 = \lambda_1 s_1 \tag{3.32}$$

Trajectory control is considered to drive all three-waypoint error components exponentially to stabilize. We would then like to order the acceleration to satisfy the following:

$$\begin{aligned}\ddot{x} &= -2\xi\omega\dot{x} - \omega^2(x - x_c) = -k_u\dot{x} - k_uk_x(x - x_c)\\ \ddot{y} &= -2\xi\omega\dot{y} - \omega^2(y - y_c) = -k_v\dot{y} - k_vk_y(y - y_c)\\ \ddot{z} &= -2\xi\omega\dot{z} - \omega^2(z - z_c) = -k_w\dot{z} - k_wk_z(z - z_c)\end{aligned} \tag{3.33}$$

Where is ξ the damping factor of the system, ω is the natural frequency of the system. From the above formula (3.30), (3.32), and (3.33), The slow-loop position control provides the attitude set points for the fast loop attitude controller. The control factor can be defined [47].

$$k_x = k_y = k_z = \frac{\omega}{2\xi} \ ,k_u = k_v = k_w = 2\omega\xi. \tag{3.34}$$

Chapter Four: Simulation and Experimental Work

SIMULATION AND EXPERIMENTAL WORK

CHAPTER FOUR

In this chapter, two programs are used to control and achieve stability of the quadrotor. First: MATLAB 2017 is a high-level programming language for technical computing that can be used for algorithm development, data visualization, data analysis, and numeric computation. Because of its simple interface and strong commands, it’s commonly utilized in research and engineering. Second: LabVIEW 2015 is a graphical programming environment, used to create automated research, validation, and production test systems.

4.1. Simulation Work

This section, first: Introduces a dynamic model of quadrotor developed and implemented at Simulink. The derivation of the quadrotor model is presented it's significant because it clarifies how the quadrotor responds to its input. Second: The possibility of modeling and controlling complex engineering systems is discussed. The modeling approach is based on a 3D CAD model built using SolidWorks and SimMechanics/MATLAB 2017 software. A real quadrotor system is transformed into a 3D computer model.

4.1.1. Mathematical Model of Quadrotor

The dynamic model of mechanical systems is traditionally derived using mathematical relations. Quadrotor modeling is useful for understanding dynamics for quadrotor control and stability. The regulation of a roll, pitch and, yaw of a quadrotor is done with PID control system. A mathematical model of a quadrotor is designed and implemented in Simulink. Parameters of the quadrotor as shown in **table 4.1**and **4.2**.

Table 4.1 parameters of a Mathematical Model.[45]

Parameter	**Units**	**Value**
J_P	N.m. s^2	70.8×10^{-6}
m	kg	1
d	N.m. s^2	1.1×10^{-6}
b	m	54.2×10^{-6}
I_{xx}	N.m. s^2	8.1×10^{-3}
I_{yy}	N.m. s^2	8.1×10^{-3}
I_{zz}	N.m. s^2	14.1×10^{-3}
g	m/s^2	9.81
l	m	0.24

Table 4.2 parameters of PID control.

For position	**Units**	**Value**
k_p	-	5
k_i	s^{-1}	0
k_d	s	10
For motor		
k_p	-	1
k_i	s^{-1}	10
k_d	s	0

I. System structure

Simulink is a multidomain simulation and Model-Based Design environment for dynamic and embedded systems. It offers an interactive graphical environment as well as a customizable set of block libraries for designing, simulating, implementing, and testing a wide range of time-varying systems. Simulink was chosen for this project because of its simple and clear graphic interface [48].

The dynamic model of mechanical systems is consisting of several blocks." **Discrete trajectory generator**” represents the quadrotor's mechanics and delivers the desired position and yaw. "**Euler** " provides the attitude desired of the quadrotor.” **torque**” and "**force**". It sends its information directly to the “**omega**," while the "Discrete trajectory generator" and "Euler" provide information on quadrotor motion. "**Quadrotor Dynamics**" denotes the quadrotor's physics and provides velocity, position, and attitude. **Figure 4.1** shows dynamic model of quadrotor in Simulink.

II. Blocks implementation

" Discrete trajectory generator” represents the quadrotor's physics and provides the desired position and yaw. **Figure 4.2** shows implementation of" Discrete trajectory generator” in Simulink

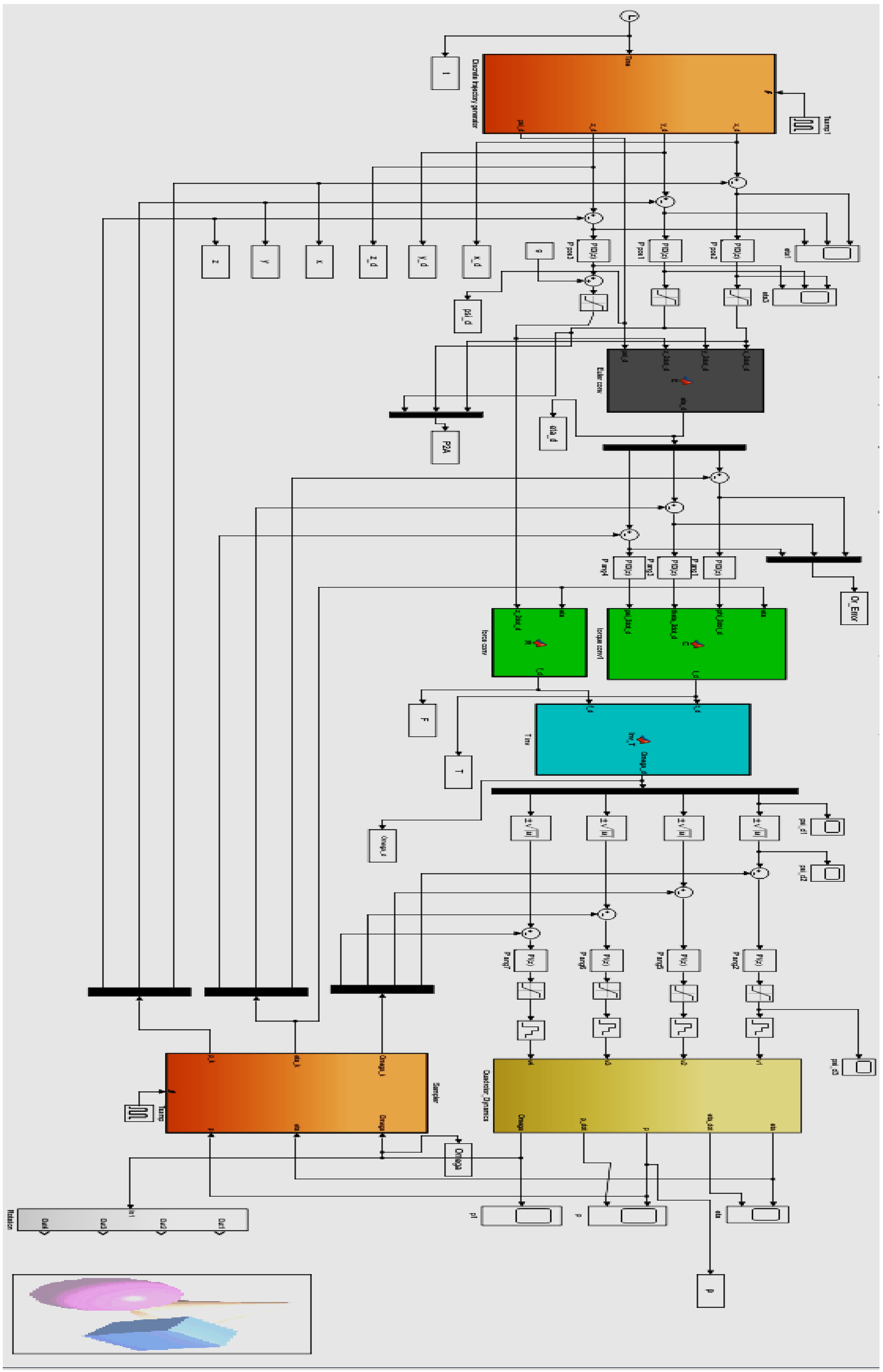


Figure 4.1: Dynamic model of quadrotor in Simulink.

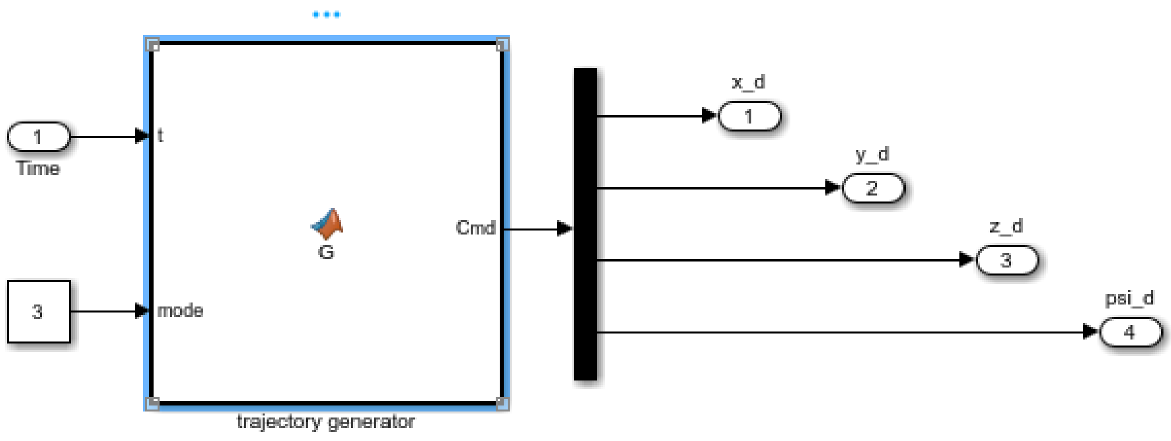


Figure 4.2: "Discrete trajectory generator" implementation

This block is implemented by writing software code. The desired direction is calculated according to the theory presented in chapter 3. Also "Euler" blocks provide the attitude desired of the quadrotor." It is implemented by writing software code by using equation 3.19 as shown in **figure 4.3**.

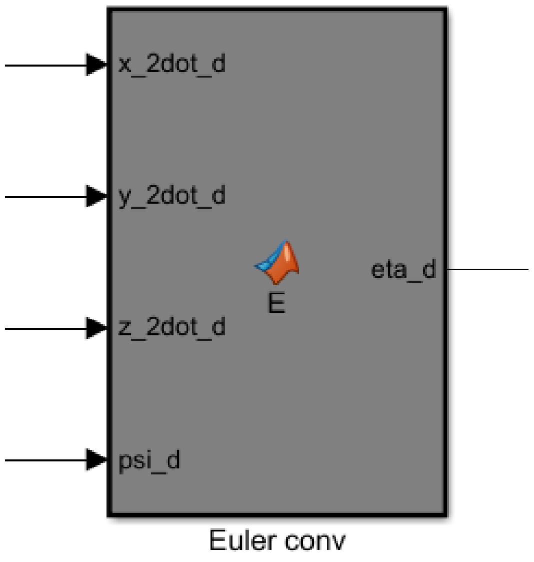


Figure 4.3: "Euler" implementation

"Quadrotor Dynamics" represents the quadrotor's physics that provides position, velocity, and attitude as shown in **figure 4.4**.

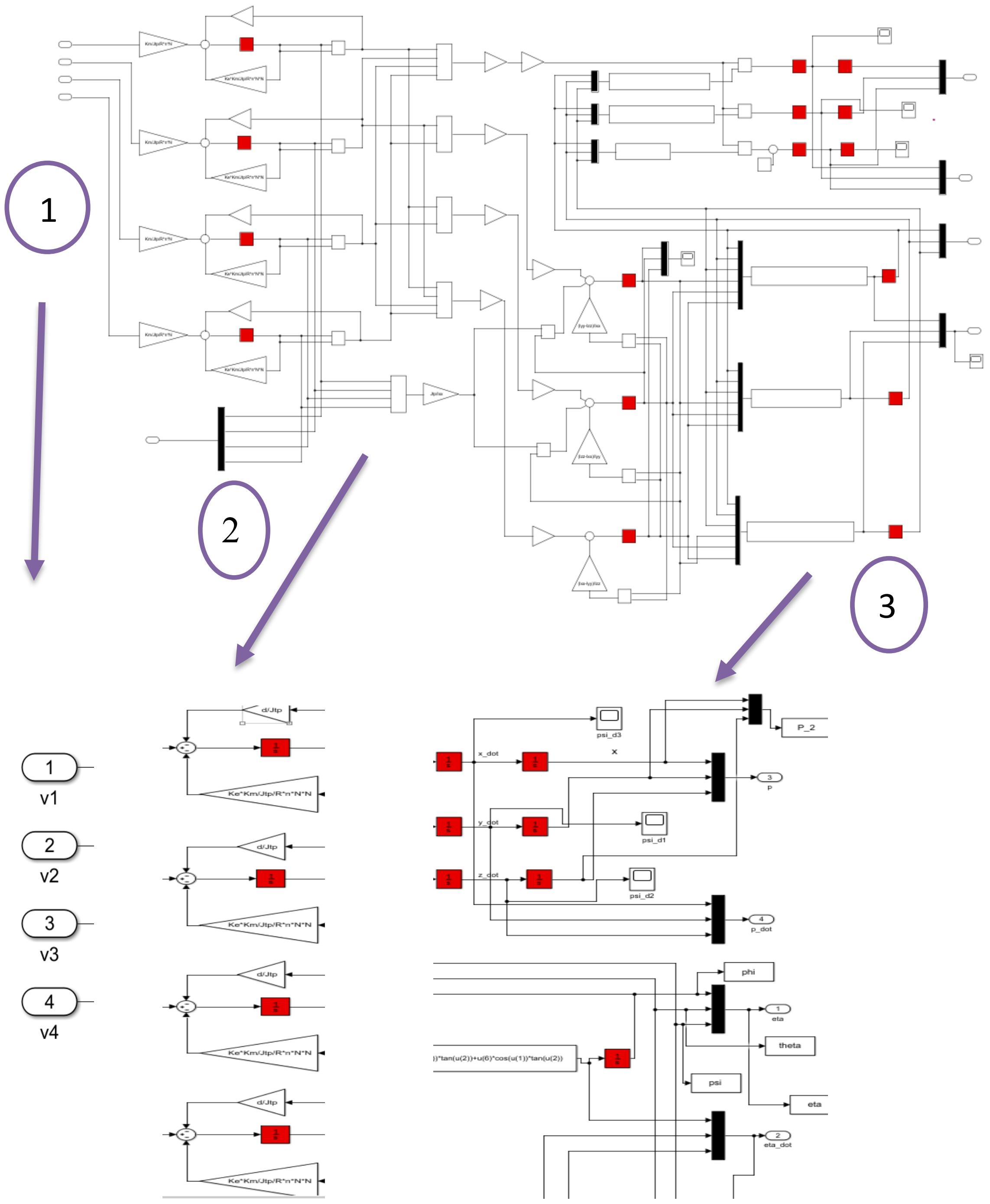


Figure 4.4: " Quadrotor Dynamics" implementation

4.1.2. CAD Model of Quadrotor

The CAD model for the quadrotor system in this thesis is used to show the capability of the 3D model implementation in modeling and control. These models echo the real-time models to improve behavior in the real world. The actual quadrotor device was converted from the real system to a 3D model. The goal of SolidWorks is to create 3D models drawing from each part including engineering data. The parts are connected correctly to form an assembly, as shown in the **figure 4.5**, which shows the 3D model designed using CAD of the SolidWorks system. Parameters of CAD model is shown in **table 4.3**

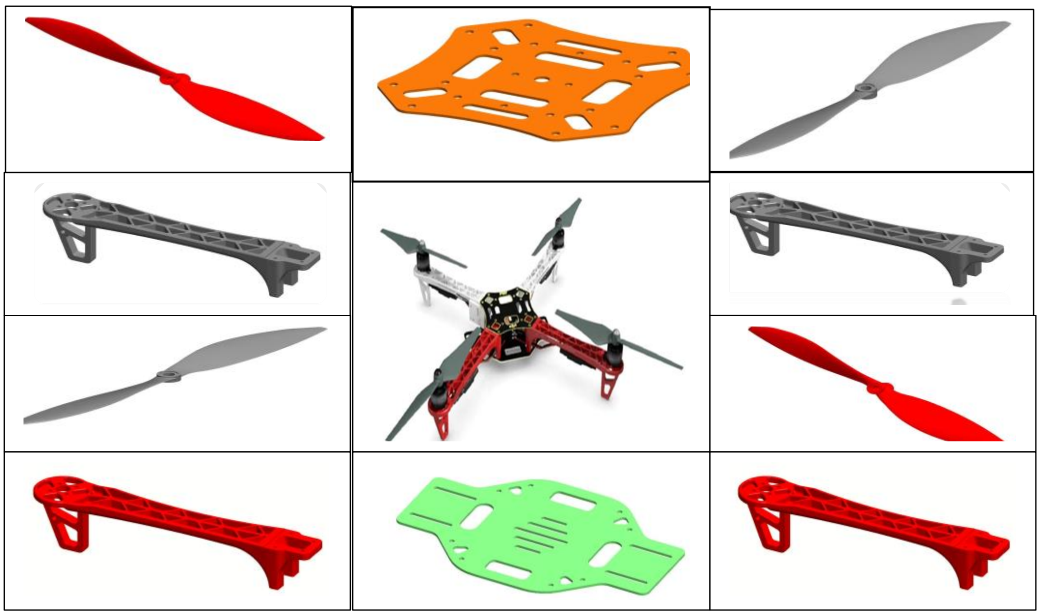


Figure 4.5: 3D model of a quadrotor system.

Table 4.3 parameters of CAD Model.

Parameter	Units	Value
L	m	0.1813
m	kg	1
d	N.m. s^2	2.5×10^{-7}
b	m	8.1×10^{-6}
I_{xx}	N.m. s^2	0.005161
I_{yy}	N.m. s^2	0.0051571
I_{zz}	N.m. s^2	0.01005
g	m/s^2	9.81
J_P	N.m. s^2	2.03×10^{-5}

I. System Description

The CAD model of a quadrotor is used in this simulator. The quadrotor's dynamics are recorded by a CAD model built in SolidWorks and then imported in Simulink using the Simscape Multibody, which is a unique feature of this simulator. To properly evaluate the controllers' performances, the trajectories of quadrotors using two alternative control strategies (PID control and Adaptive control) can be compared to the ideal reference trajectory.

The model of the whole system is composed of several interconnected blocks in a classic feedback structure as shown in both **figures 4.6 - 4.7**. which consists of the "**Reference Path**" block that produces the path to follow for the quadrotor. This simulator allows you to hover in place or follow a straight or conical path. The vector $[x_c, y_c, z_c, \psi_c]^T$is the block's output.

The block "**Sliding Mode**" generates the required roll and pitch angle to ensure the required quadrotor position in the x-y plane by using equation 3.30

"**Adaptive Controller**" block uses an architecture from model reference adaptive control to simulate and monitor the quadrotor's ideal dynamics. **"PID Controller"** block applies the classic PID control.

"Quadrotor and Environment Model" blocks combine the CAD models of the quadrotors and the environment to perform numerical simulations and generate 3D animations. **"Create State Vector"** blocks determine the position of the quadrotor, and the attitude determined by the 'Quadrotor and Environment block. **Plots and Signal Logging"** block allows the modeling to be saved.

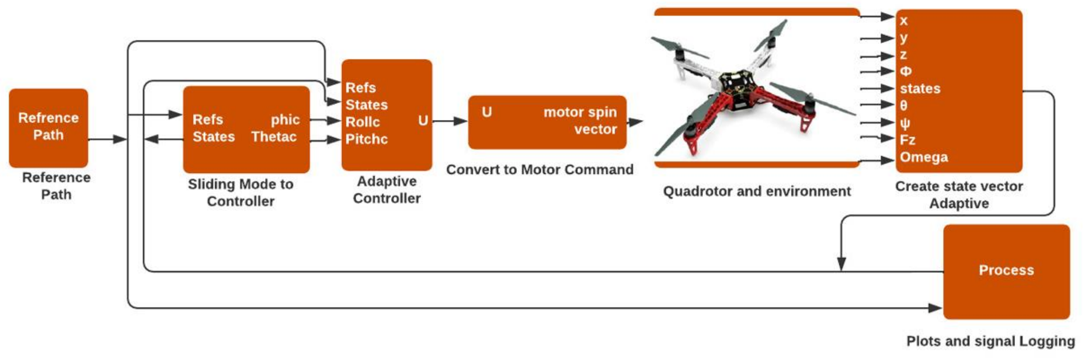


Figure 4.6: CAD-simulator and controller (sliding mode controller with adaptive controller).

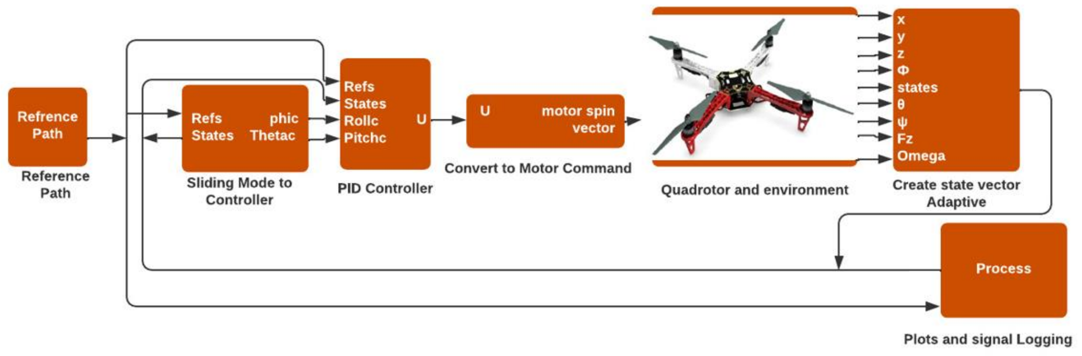


Figure 4.7: CAD-simulator and controller (sliding mode controller with PID Controller).

II. Implementation of paths

Here reference path block produces the path to follow for the quadrotor. This simulator allows you to hover in place or follow a straight or conical path. the vector $[x_c, y_c, z_c, \psi_c]^{\mathrm{T}}$is the block's output can be seen in **figure 4.8.** First, the conical trajectory was created using equations as shown ($4.1a$, $4.1b$, $4.1c$, *and* $4.1d$) below. So that conical path block that implement is shown in **figure 4.9**. Also, it can be seen in **figure 4.10** pictures showing that the movement of the path for the quadrotor at a different time.

$$x = r\cos\theta \text{ m} \tag{4.1a}$$

$$y = r\sin\theta \text{ m} \tag{4.1b}$$

$$z = \frac{\theta}{4} \text{ m} \tag{4.1c}$$

$$\varphi_d = 0 \text{ deg} \tag{4.1d}$$

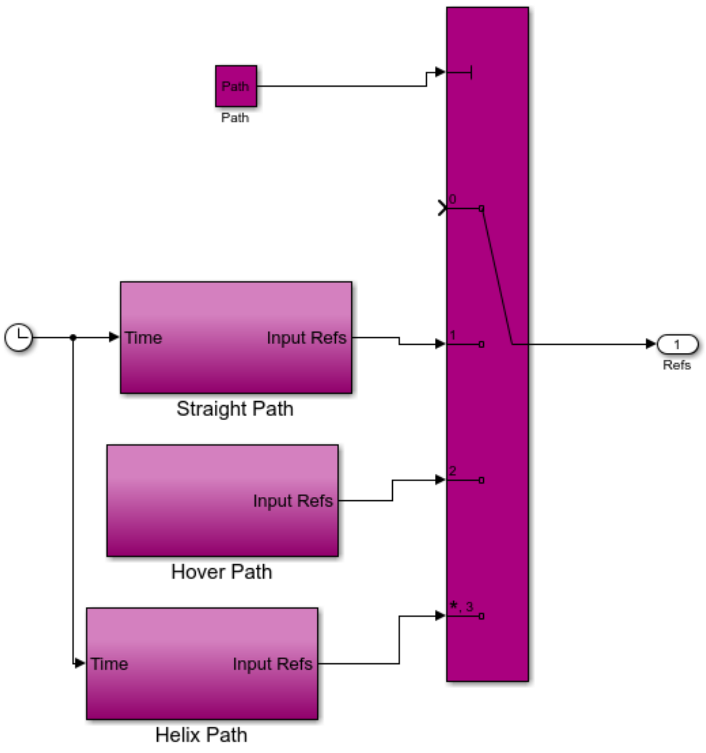


Figure 4.8: Reference path block in Simulink

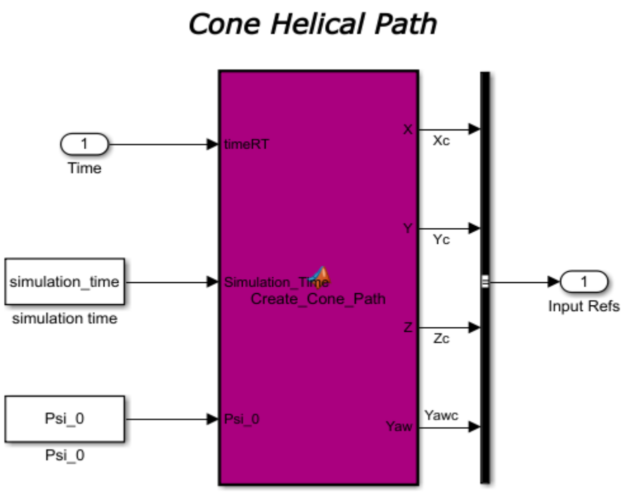


Figure 4.9: Conical path block in Simulink

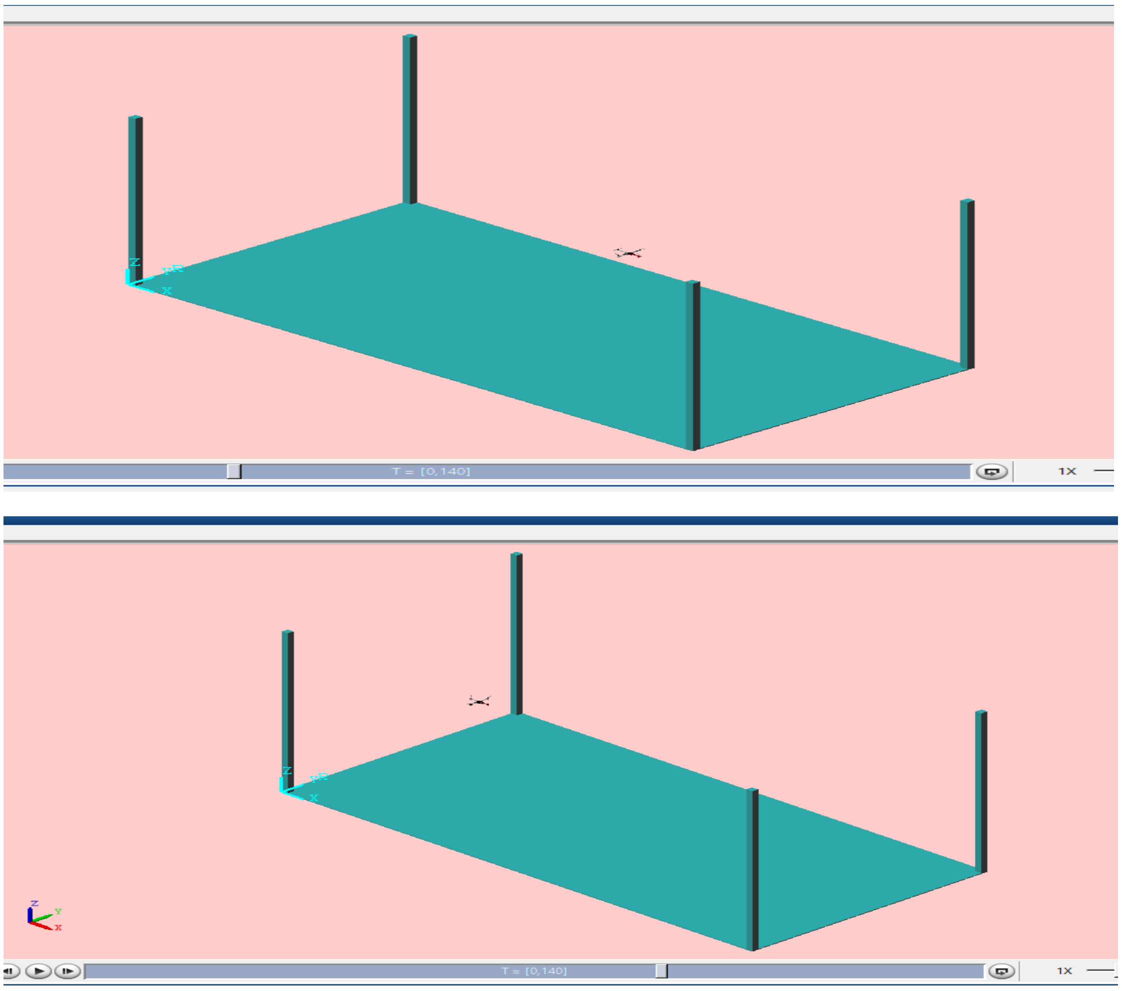


Figure 4.10: 3D image to conical path of quadrotor at (43-89) sec

Second, the straight trajectory was created using equations as shown (4.2a, 4.2b, 4.2c, *and* 4.2d) below. So that straight path block that implement is shown in **figure 4.11**. Also, it can be seen in **figure 4.12** the pictures that showing the movement of the path for the quadrotor at a different time.

$$x_d = 0.05 * r \quad \text{m} \tag{4.2a}$$

$$y_d = 0 \quad \text{m} \tag{4.2b}$$

$$z_d = 0 \quad \text{m} \tag{4.2c}$$

$$\varphi_d = 0 \quad \text{deg} \tag{4.2d}$$

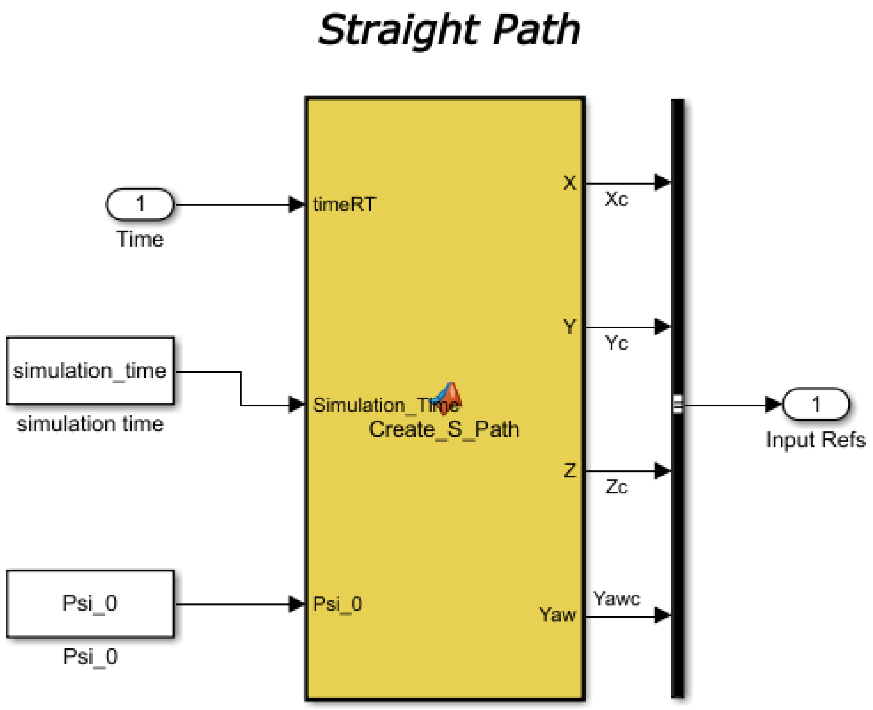


Figure 4.11: Straight path block in Simulink

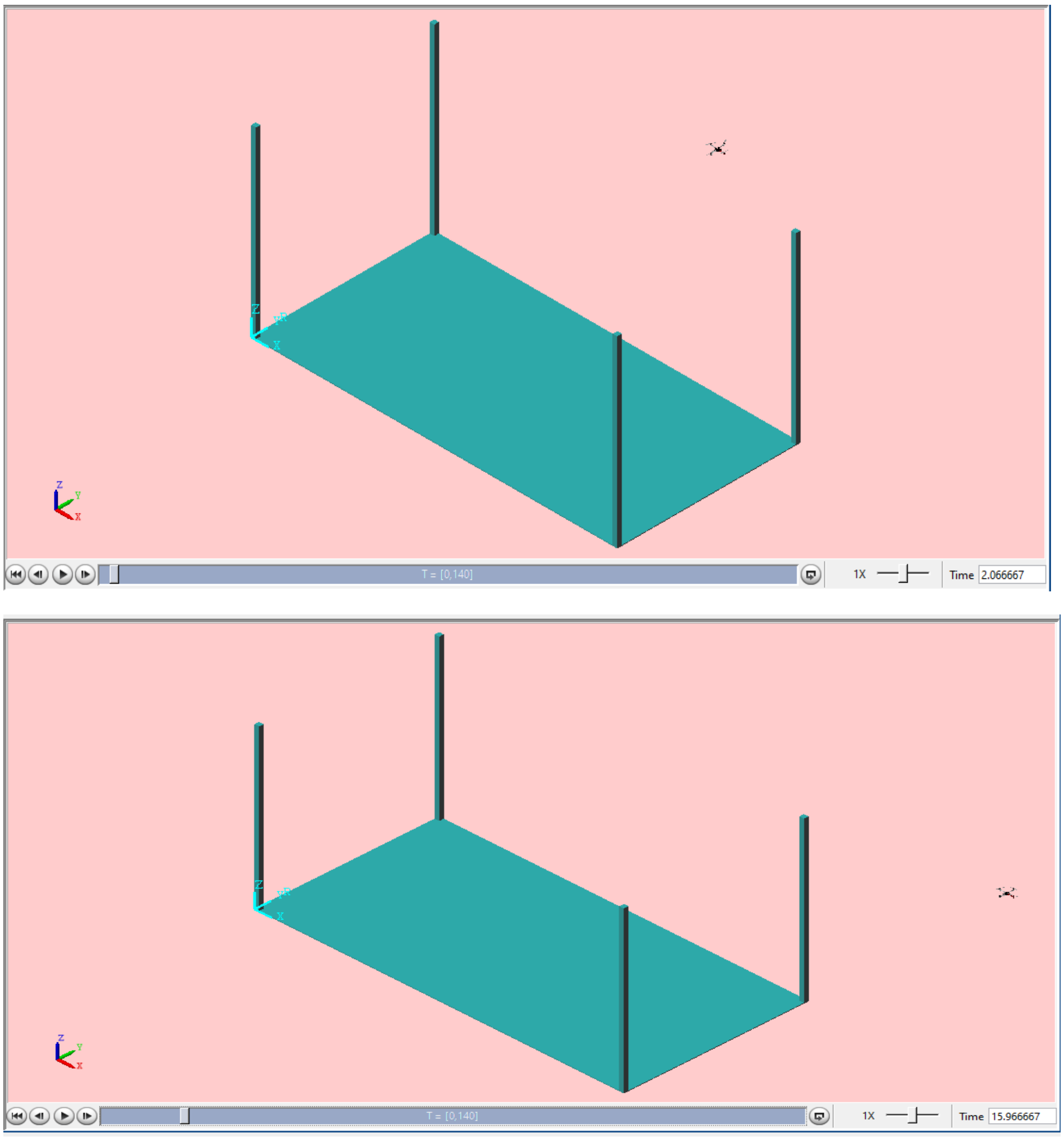


Figure 4.12: 3D image to straight path of quadrotor at (2-15) sec

Third, the hover trajectory was created using equations as shown ($4.3a$, $4.3b$, $4.3c$, *and* $4.3d$) below. Also, a comparison is made between hover path of the original model **figure 4.13** and hover path which was created in Simulink **figure 4.14**. So that, it can be seen in **figure 4.15** the pictures that showing the movement of the path for the quadrotor at a different time.

$$x = 0 \text{ m} \tag{4.3a}$$

$$y = 0 \;\; \text{m} \tag{4.3b}$$

$$z = 2.5 \quad \text{m} \tag{4.3c}$$

$$\varphi_d = 0 \;\; \text{deg} \tag{4.3d}$$

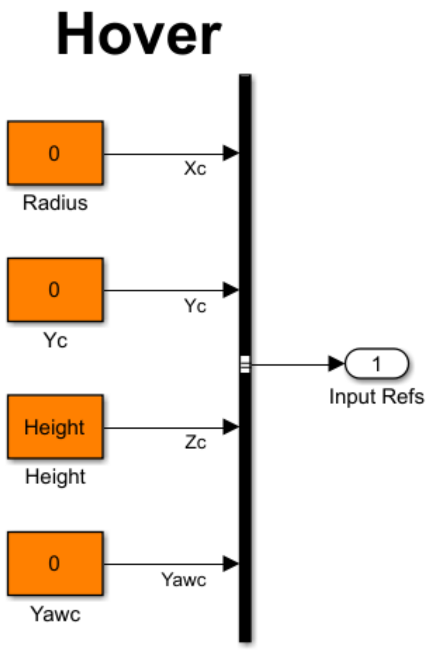


Figure 4.13: Hover path of the original model

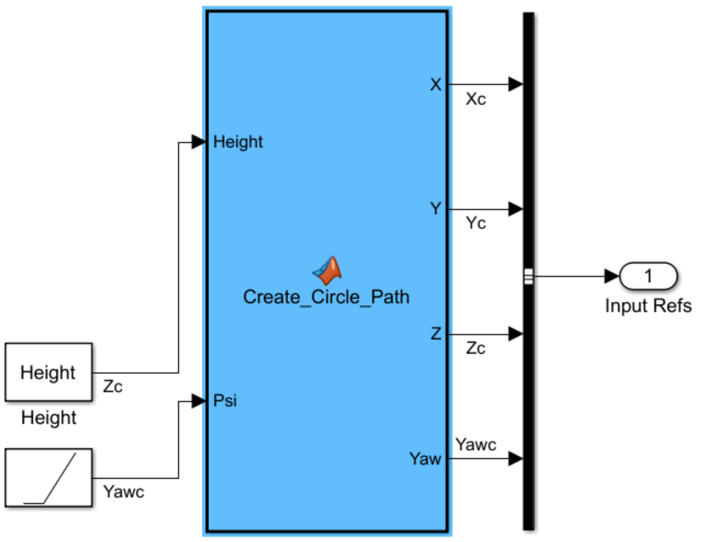


Figure 4.14: Hover path block in simulation

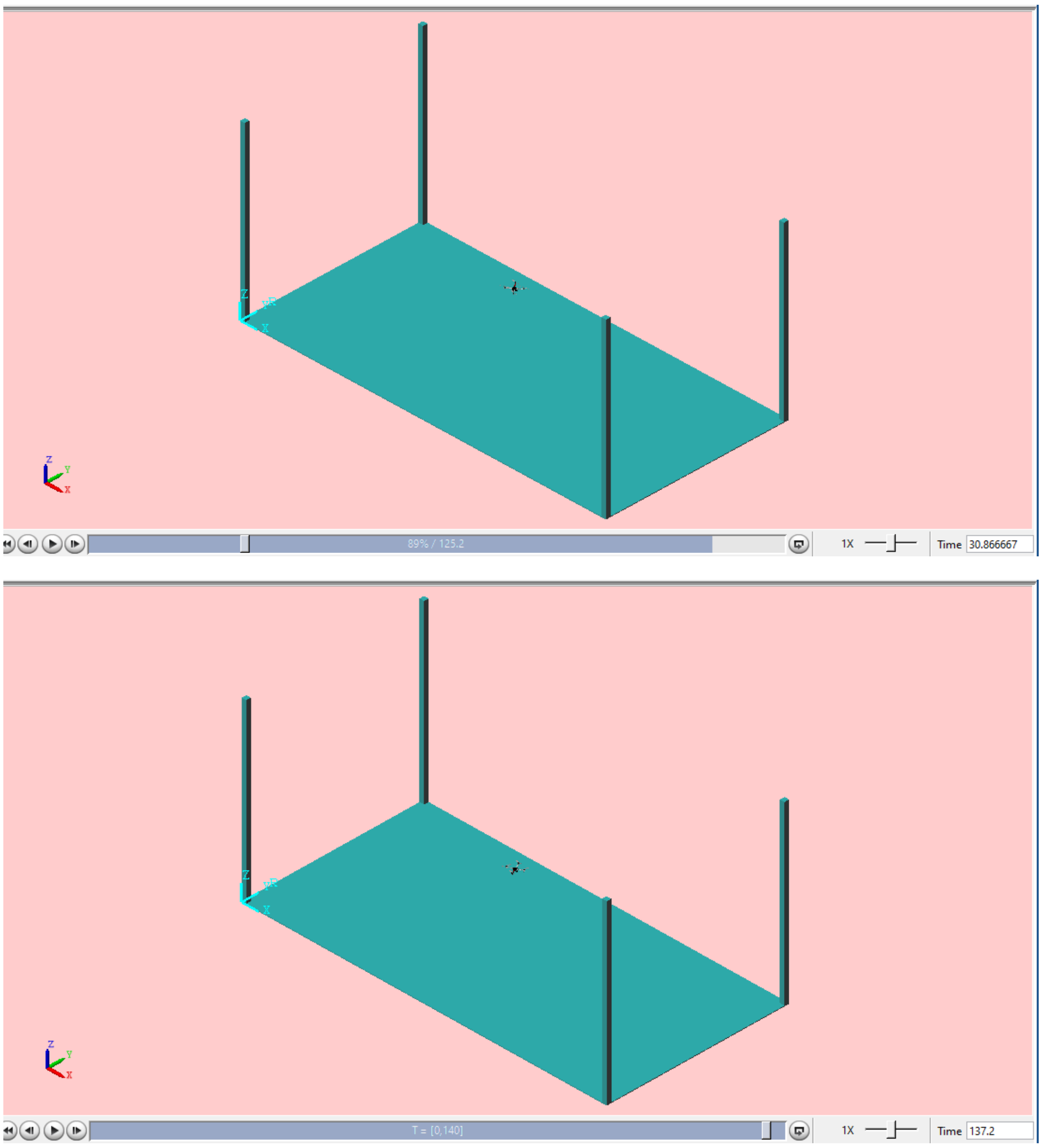


Figure 4.15: 3D image to hover path of quadrotor at (30-137) sec

4.2. Experiment Work

The platform used in this thesis is Parrot AR. Drone. AR. Drone 2.0 is a quadrotor with four rotors. The structure of the mechanical is made up of four rotors that are connected to the four ends of the junction where the battery and radio-frequency gadget are connected. A computer connection is required so that users may develop code to control the AR

Drone and monitor it while in flight. If the AR Drone could be linked to a computer, the user could read sensors and live video streams, command and maneuver AR. Drone (for more details about AR. Drone in Appendix A) AR. Drone is examined as a robotic platform in this section. Also, experiments that were carried out on the plane are discussed where the plane was programmed and controlled using two programs, MATLAB and LabView.

4.2.1. programing AR. Drone with LabVIEW

LabVIEW programming environment is composed of functions called VI's (for virtual instruments). VI shows graphically as a block with inputs and outputs, as shown in **figure 4.16.** By virtually wiring two VIs together, they might be joined. This method can be used to generate entire programs. Each program contains a Block Diagram that shows where the VIs is located and a Front Panel that shows and manipulates graphical user interface (GUI) objects. AR. Drone was programmed using two methods: joystick controller, and keyboard controller.

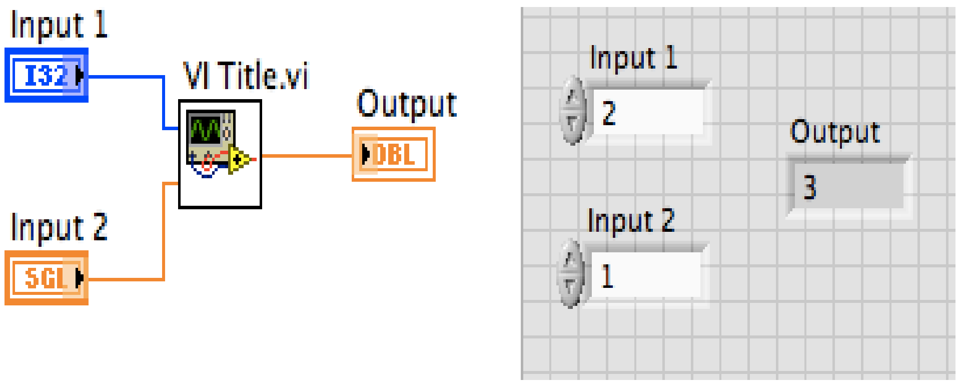


Figure 4.16: The Block Diagram (left) with a VI that has inputs and output, also shown on the Front Panel (right) [49]

I. System Description

Two tools used in this work to control by using a joystick, and a keyboard, which consist is made up of high-level VIs that handle things like flying the AR Drone, reading and decoding video streams, and gathering sensor data as shown in **figure 4.17**.

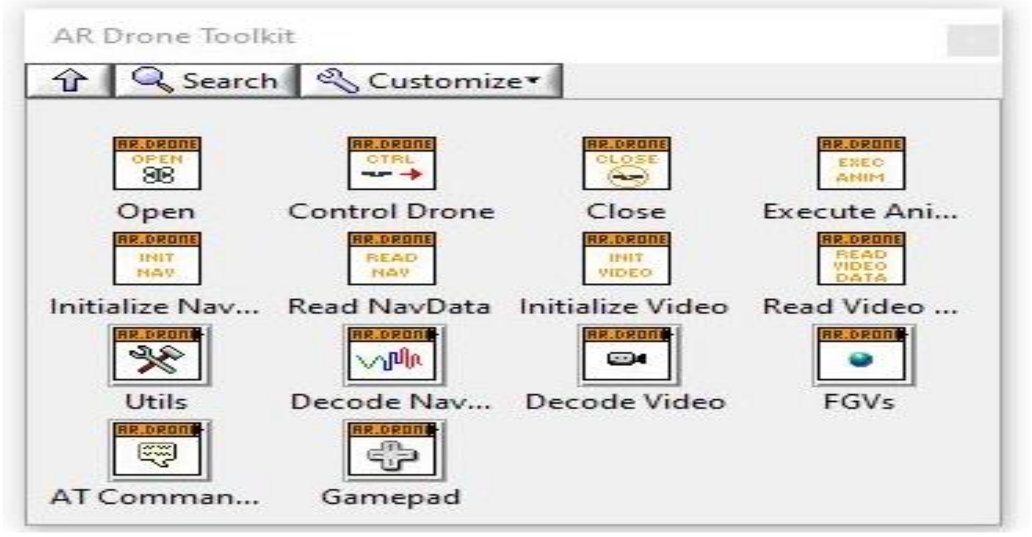


Figure 4.17: Palette of VI's comprising the AR Drone LabVIEW Toolkit

There are VIs in **figure 4.17** to execute the primary operations over the AR. Drone. **Open VI** enables us to begin accessing data and video; this VI has an input cluster for specifying the IP address of both the AR. Drone and the PC.

The connection with the drone is finished by **Close VI**. There is a VI in the toolkit called **Control Drone** that transmits basic control commands to the AR. Drone, such as take-off and landing, emergency landing, hover mode, and movement directives as show in **figure 4.18**.

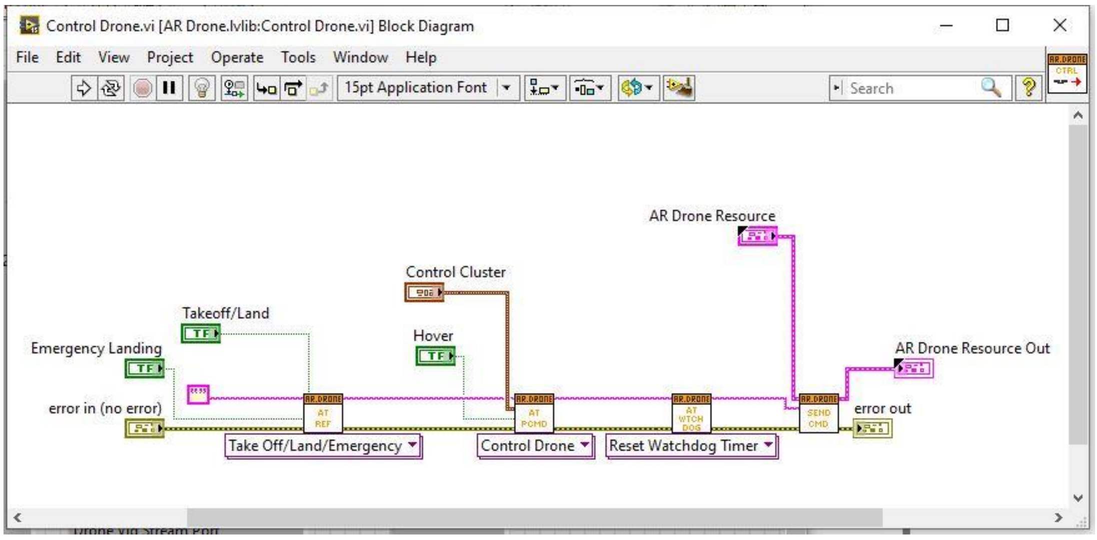


Figure 4.18: Block digram of control Drone of the AR Drone LabVIEW Toolkit

The use of tool "write to measurement" to save the data related to the AR. Drone. " Font panel " for a joystick, and keyboard shown in **figures 4.19-4.21.** A Block diagram was also presented to control the joystick and the keyboard in the **figure 4.20-4.22.**

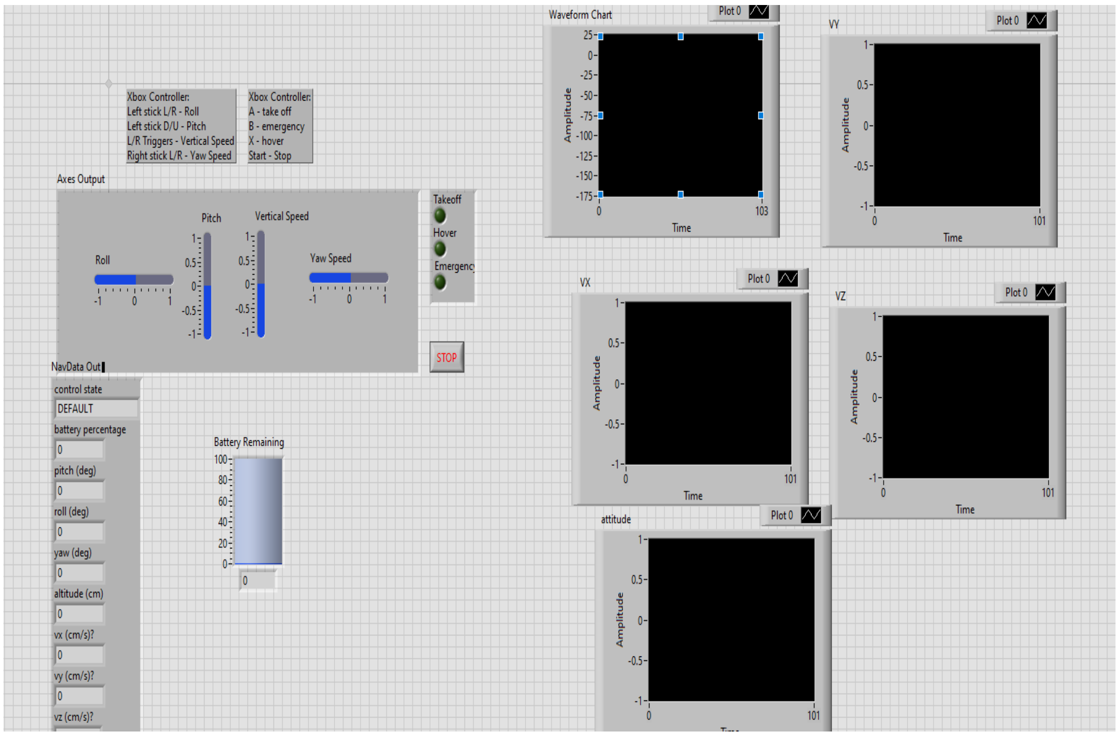


Figure 4.19: Font panel of joystick controller program in LABVIEW

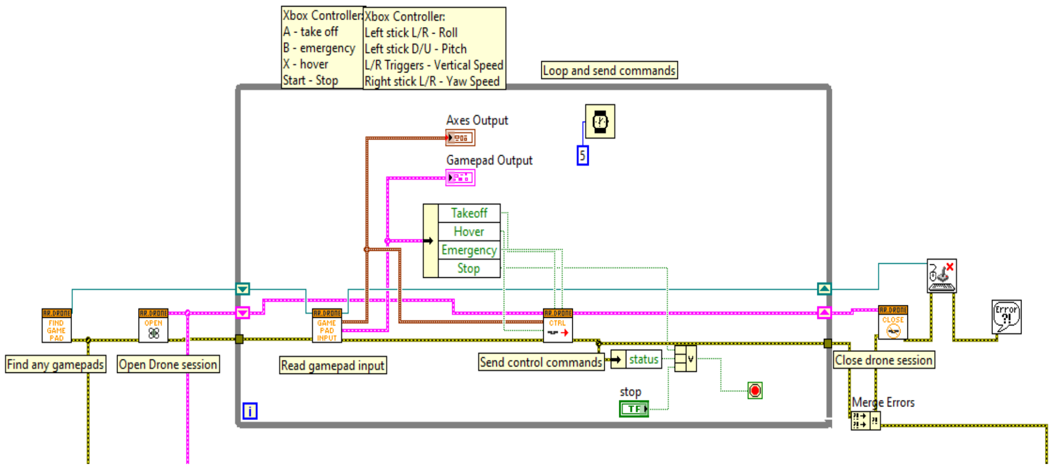


Figure 4.20: Block diagram of joystick controller program in LABVIEW

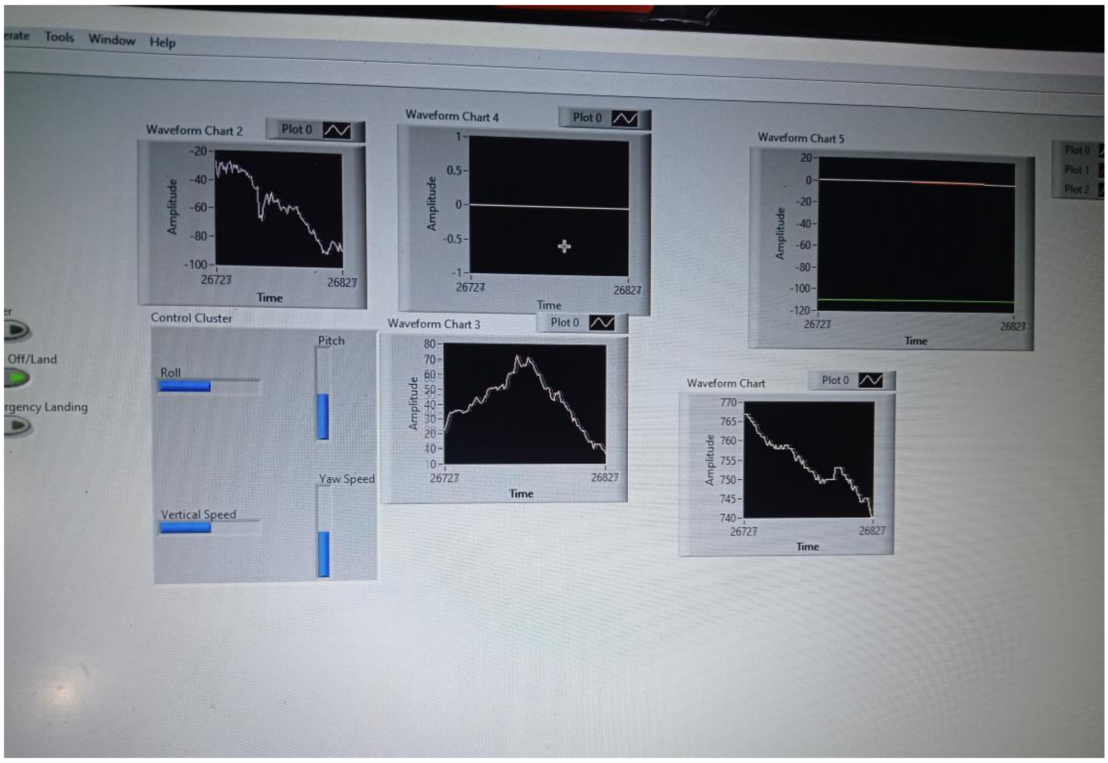


Figure 4.21: Font panel of keyboard controller program in LABVIEW

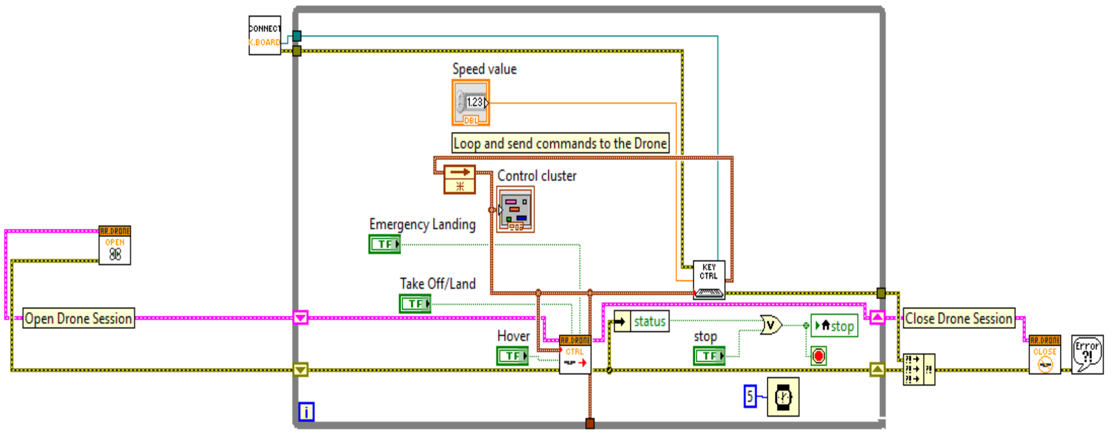


Figure 4.22: Font panel of keyboard controller program in LABVIEW

II. Experiment setup

The experiment is shown in **figure 4.23**, To perform any experiment using this setup

1- At the start of the experiment, the drone's battery is attached to the chassis where the lights will flash red with a modest movement of the propellers to indicate its operations as shown in **figure 4.24**
2- Using the laptop, the drone is connected via wi-Fi **(see figure 4.25),** the LED changes from red to green as shown in **figure 4.26** (however, the color of the light does not always change, possibly due to a low battery) indicating that the connection has been established correctly.
3- Open the LabVIEW program that have been installed on the laptops and select types control.
4- Finally, the drone's communication procedure has been completed, and it is now ready to take off and perform any trial as shown in **figure 4.27.**
5- Stop LabVIEW program.

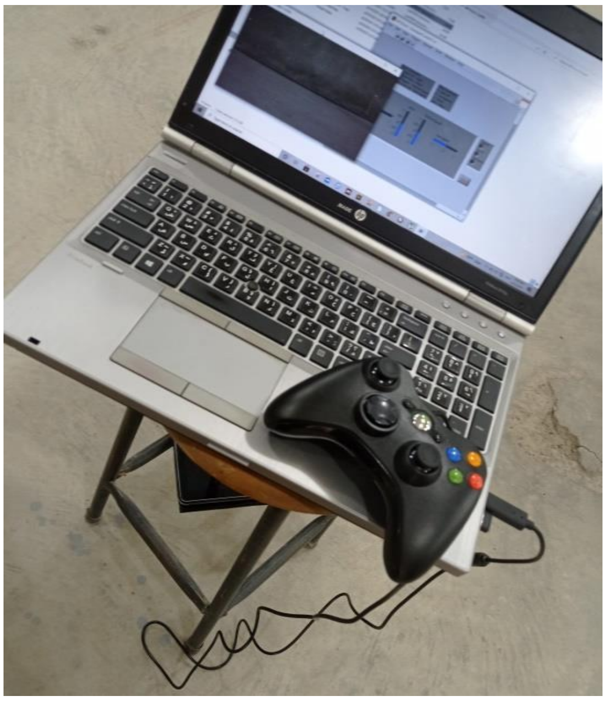
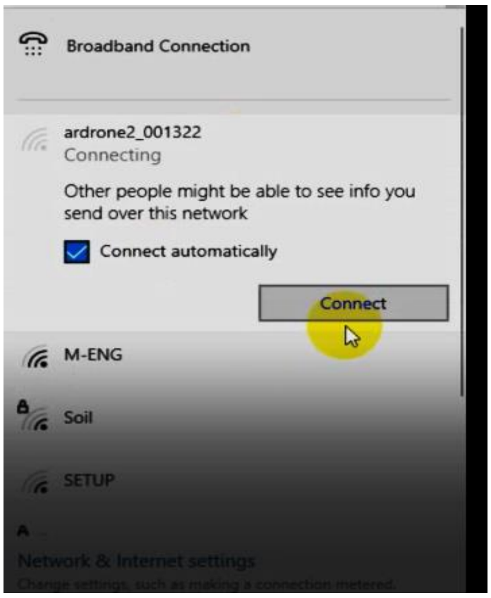


Figure 4.23: Experiment setup

Figure 4.24: Before connecting the AR Drone with computer



Broadband Connection

ardrone2_001322
Connecting

Other people might be able to see info you send over this network

☑ Connect automatically

Figure 4.25: AR. Drone Wi-Fi network

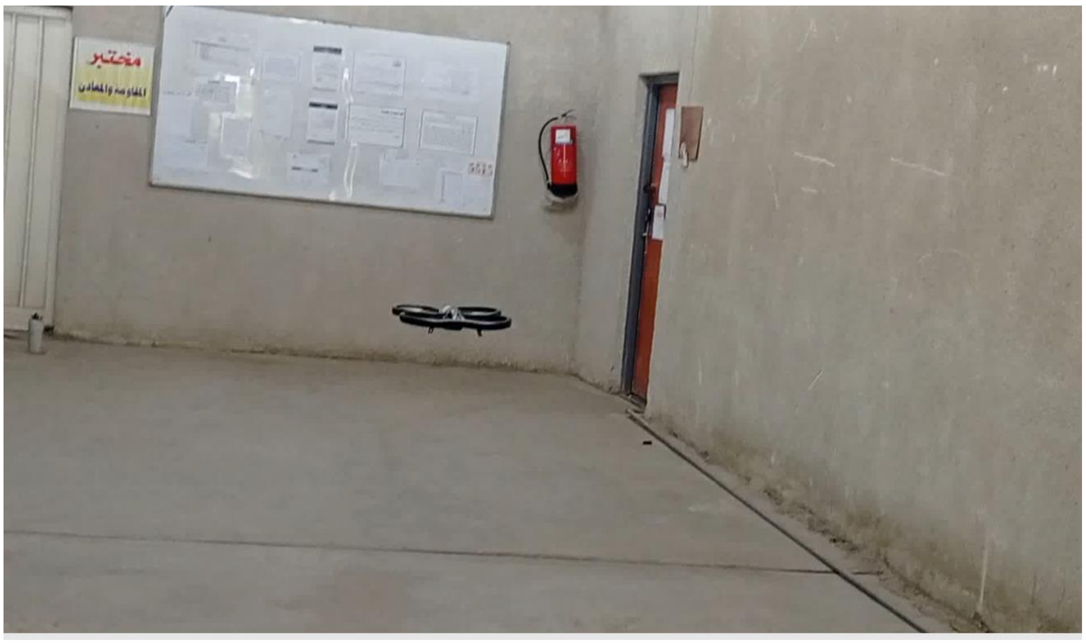


Figure 4.26: after connecting the AR Drone with computer

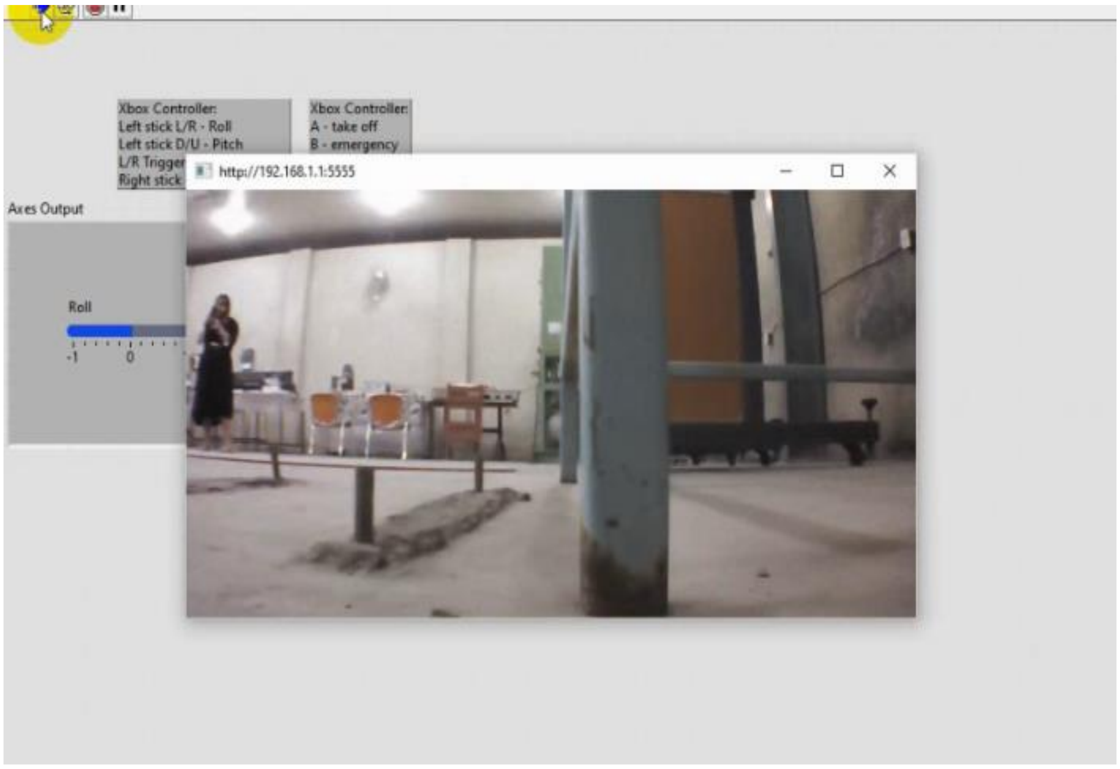


Figure 4.27: Some pictures taken by drone during the experiment

III. Experiment Execution

When the drone receives the takeoff signal, it will read the data from the sensors and send it to the computer. The application measured the speed on the x and y axes, which were named v_x and v_y . So that, it is using program to convert the resulting velocity to displacement as shown in the **figure (4.28).**

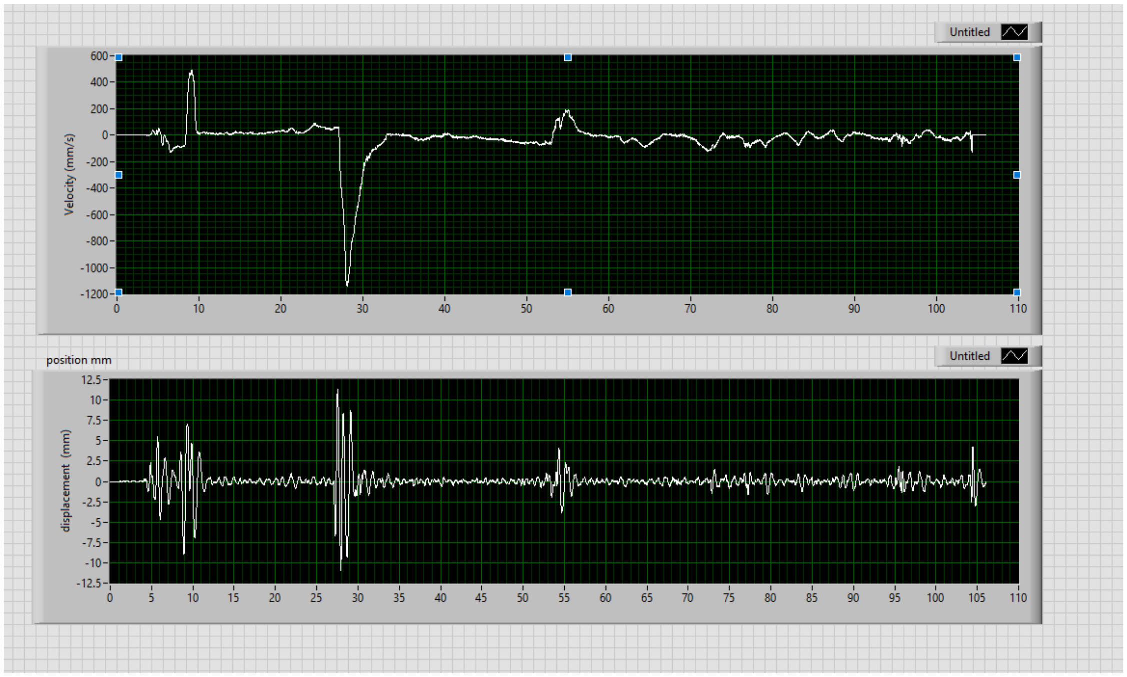


Figure 4.28: Convert velocity to displacement program in LABVIEW

First: to control the drone using the keyboard. Key T is for Take-off and Landing (press it once briefly for Take-off, press it again to Land). When takeoff, the keys W, A, S, D are used to regulate the drone's tilting Letter A for left movement, D for right movement, W for forwarding movement, and S for backward movement (in the program these two axes appear as Pitch and Roll). The arrow keys LEFT and RIGHT control the rotation in one place (in the program it is called Yaw Speed), the arrow keys UP and DOWN control the altitude (in the program it is called Vertical Speed), key H is for Hovering (press and hold to enter Hover mode, and finally, when the drone has landed, the program is stopped.

Second: The drone is controlled using a joystick. Left joystick is used to control the tilting of the drone (in the program these two axes appear as Pitch and Roll). The right joystick controls the rotation in one place (in the program it is called Yaw Speed). The left and right triggers (1 and 2) are used to control the altitude (in the program Vertical Speed). button A is for Take-off and Landing (press it once briefly for Take-off, press it again to Land). button B is for Hovering (press and hold to enter Hover mode.

Experiments conducted with the keyboard :Hovering ,and straight path

Experiments conducted with the joystick: Hovering

V. Experiment Hovering

Quadrotors, like helicopters, are capable of hovering. Environmental conditions (wind, subsurface, aerodynamic interactions) and control software play a role in hovering stability. AR. Drone hovered for 110 seconds as part of an experiment. This experiment was done ten times, resulting in 5306 movement samples collected during 120 seconds by joystick, and hovered for 90 seconds resulting in 5053 movement samples collected during 90 seconds by the keyboard.

VI. Experiment Horizontal movement (straight path)

In this experiment the AR. Drone is flown in a straight line. AR. Drone flown for 120 seconds as part of an experiment. This experiment was done ten times, resulting in 6000 movement samples collected during 120 seconds by keyboard as shown in **figure 4.29**, where it shows the movement of the drone in a straight path and its pass through a rotating disk.

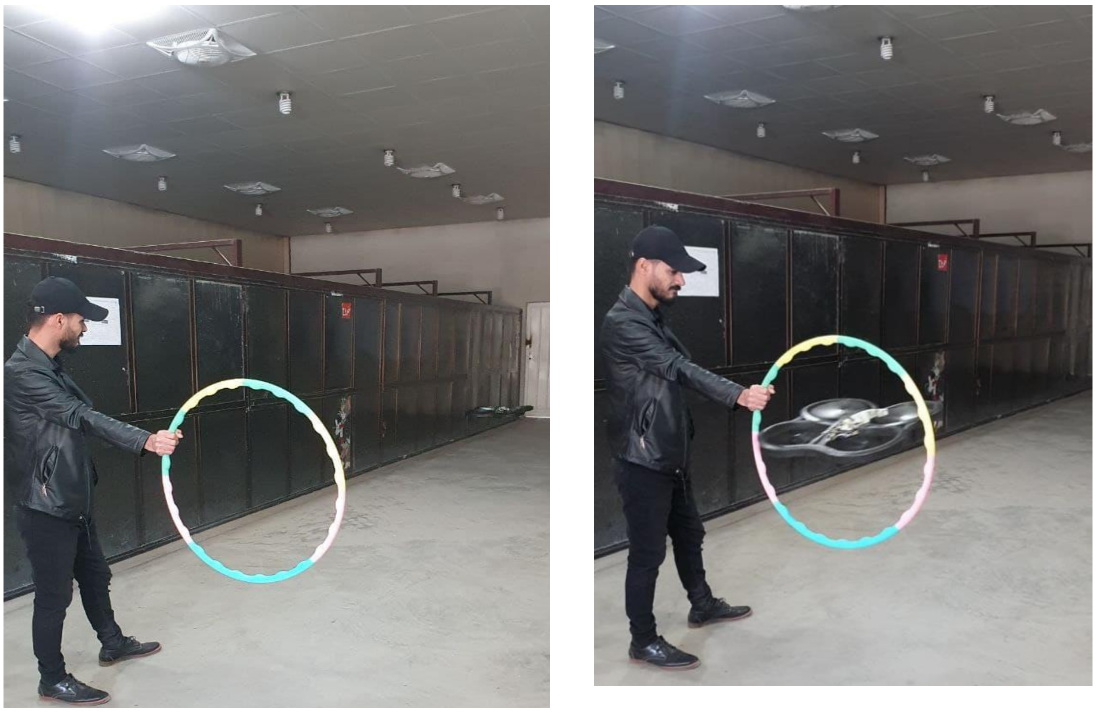


Figure 4.29: Experiment horizontal movement

4.2.2. Programming AR. Drone with MATLAB

The drone was programmed using MATLAB and the tools (AR Drone 2.0 MATLAB / Simulink Target), where a MATLAB project provides a different approach to establishing communication between Simulink and AR Drone 2.0 using an embedded coder, this tool is referred in [50] see figure **4.31** that explains example uses , and the drone was programmed using another tool, developed a Simulink model based on the AR Drone 2.0 SDK for the purpose of establishing communication between the Simulink platform and the AR Drone 2.0 , this tool is referred in [51].

At the beginning of the experiment, a connection is made between an aircraft and a computer by using an example sample in **figure 4.30**.

The error that appears despite the existence of the function is shown in figure 4.31. An attempt was made to correct this error for 9 months, the owners of the tools were reached but no response was obtained. It was concluded that these two tools cannot be used unless licenses are obtained from the company, where the connection between the MATLAB software and the drone is not done through Wi-Fi.

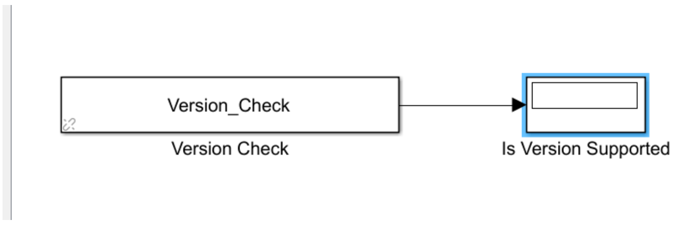
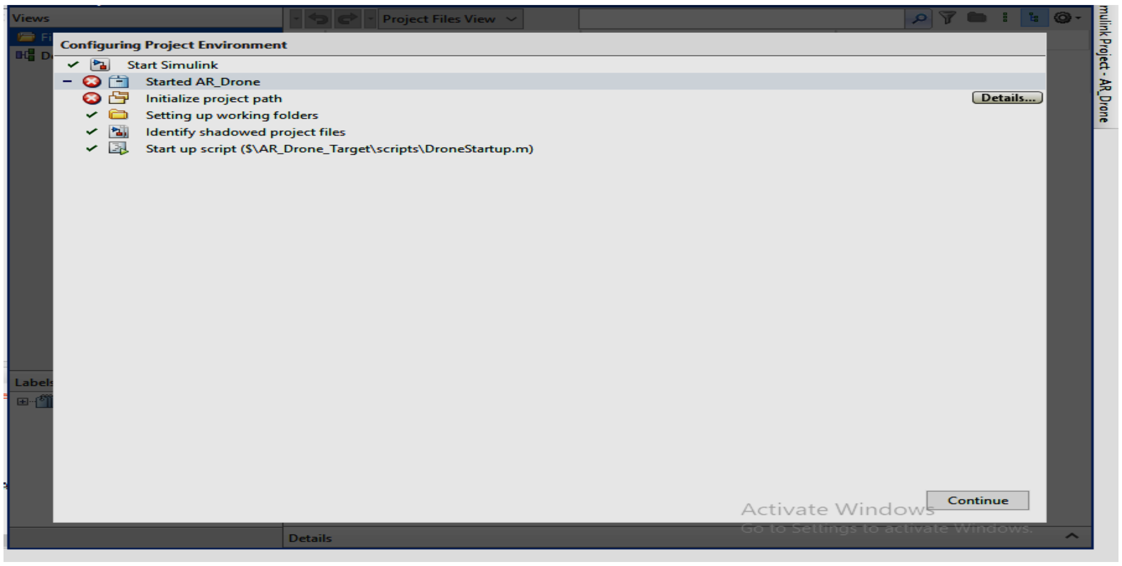
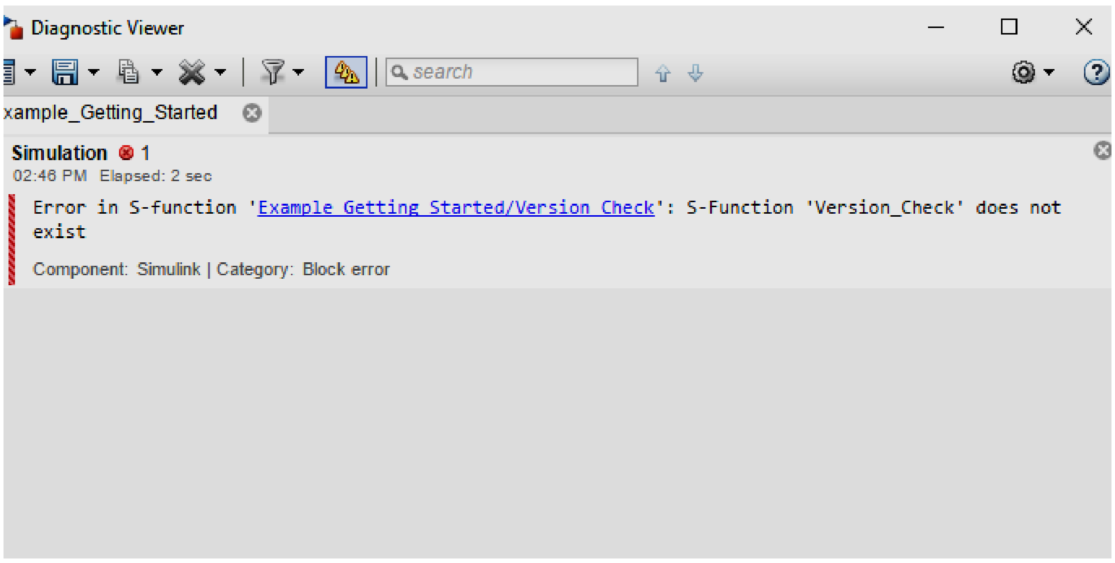


Figure 4.30: example getting started in MATLAB



Configuring Project Environment

- Start Simulink
- Started AR_Drone
- Initialize project path
- Setting up working folders
- Identify shadowed project files
- Start up script ($\AR_Drone_Target\scripts\DroneStartup.m)



```
Error in S-function 'Example Getting Started/Version Check': S-Function 'Version_Check' does not
exist
```

Component: Simulink | Category: Block error

Figure 4.31: Error in example getting started

Chapter Five:
Results and discussions

RESULTS AND DISCUSSIONS

CHAPTER FIVE

In this chapter, simulation results and experiments of the drone are presented and discussed. First, the simulation results are divided into two parts: The mathematical model and 3D CAD model. In a mathematical model, the mathematical model of the UAV was modeled using MATLAB software and using mathematical equations. And in the CAD Model, CAD was improved and a helical path, hover path, and straight path was added using Simulink. Second, the drone was controlled, and several different experiments were conducted by using the program LabVIEW. Then a comparative study between theoretical and experimental results are presented.

5.1. Simulation Results of Mathematical Model

In this section, Simulink was utilized to represent the quadrotor system's mathematical model, and in addition to the position, attitude, and motor controllers. The goal of this section is to see how effectively the quadrotor system's performance can be stabilized by the controllers, and to analyze the vehicle's dynamic behavior. So that, the performance of the quadrotor is displayed as a virtual model which will provide a platform for researchers to perform an improved analysis of the UAV, cost saving and physical development time. The parameters of dynamic model of quadrotor are illustrated in **table 4.1** that mentioned previously. The results are explained from **figures 5.1 to 5.7.**

Figure 5.1 explains 3D trajectory of the quadrotor. **Figure 5.2** displays the positions of the quadrotor system in three different directions (x, y, z). Where it is observed that position z different from position x and y ,and the quadrotor is in position z start zero and continues to increase to 30 m in 80 sec and then stabilize at 80sec. **Figure 5.3** also shows the attitude of the quadrotor system in three different directions $(\emptyset, \theta, \varphi)$. And it can be noted that the yaw response is different from the pitch and roll response. **Figures 5.2, 5.3** show the comparison between the desired path and actual, where the red line indicates the desired, and the blue line shows the actual.

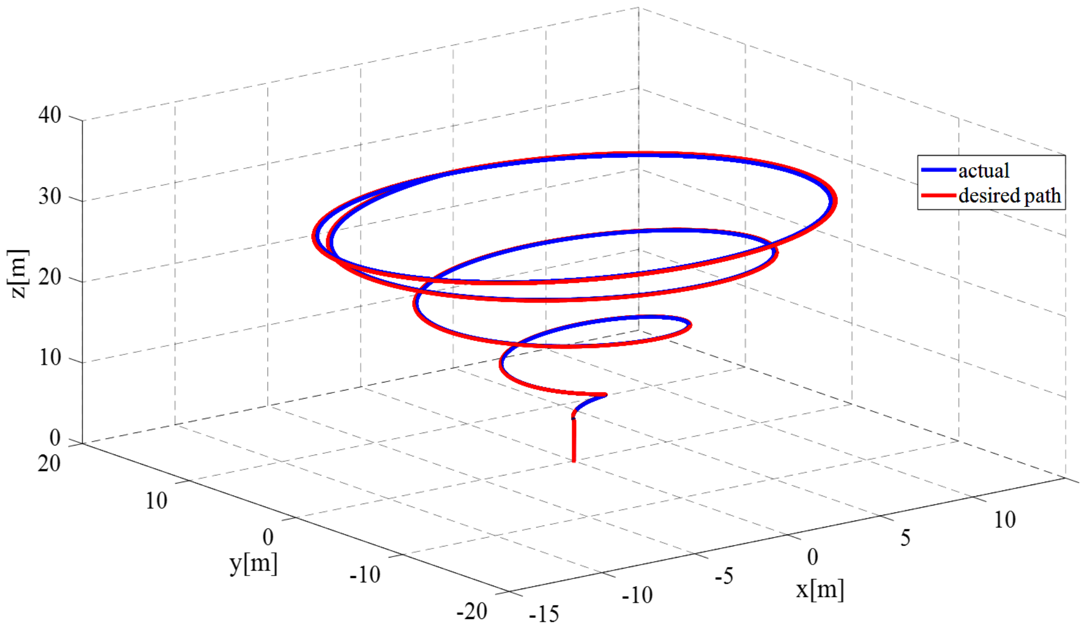


Figure 5.1.Trajectory of the quadrotor.

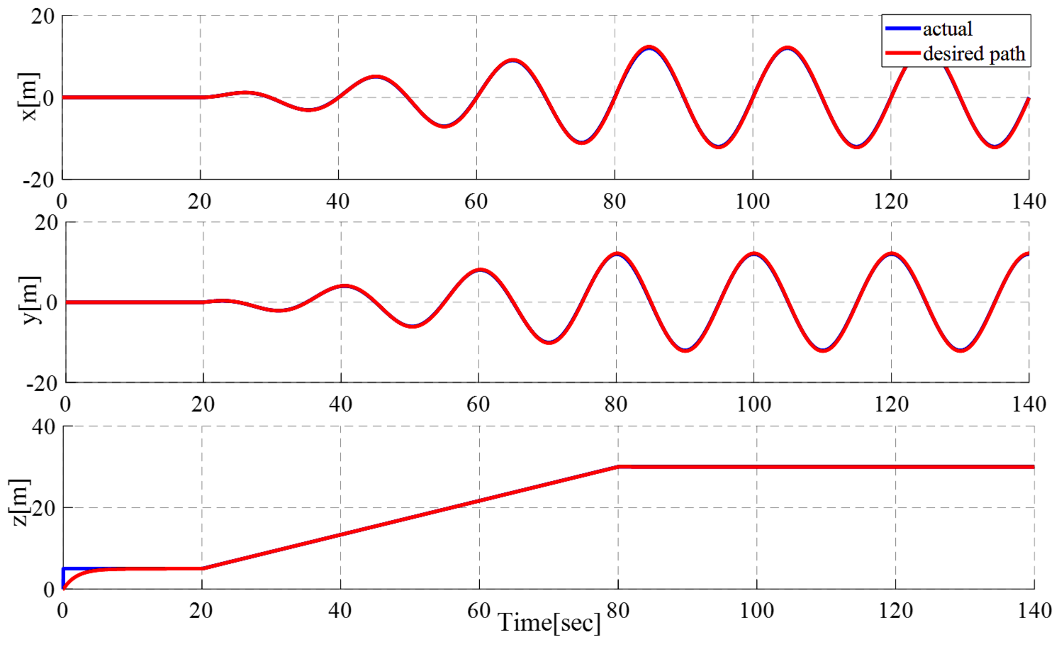


Figure 5.2. Position of the quadrotor in (x, y, z).

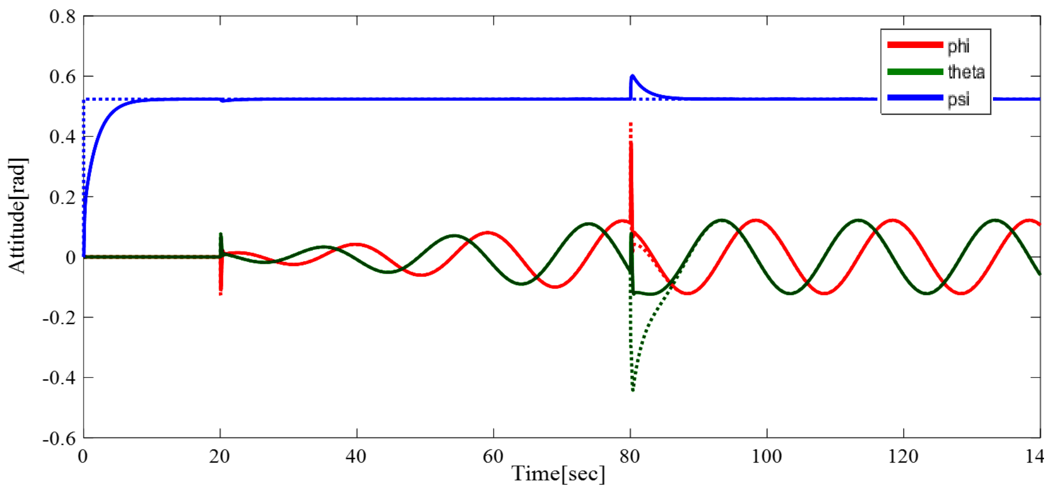


Figure 5.3 Attitude angles of the quadrotor $(\emptyset, \theta, \varphi)$. ——— Actual , ...desired.

Figure 5.4 states that the force decreases from 30- 8 N and increases to 10 and stable from 0-20 sec, and then increases to 14 N and decrease to 10 N in 20 sec then will stable from at 80 sec. Also, it will decrease the force and increase to 10 N again and will be stable from 80 sec to 140 sec. **Figure 5.5** is demonstrated the torque of the quadrotor which produced by the principal movement inputs. Figure **5.6** shows the angular velocity of the propellers, where the four propellers' angular velocity is required to balance the acceleration due to gravity which allowing the quadrotor to hover instantaneously. **Figure 5.7** shows the error for positions and it can be that magnitude of root mean square for position (x, y, z). is 0.1258, 0.1142, and 0.4172.

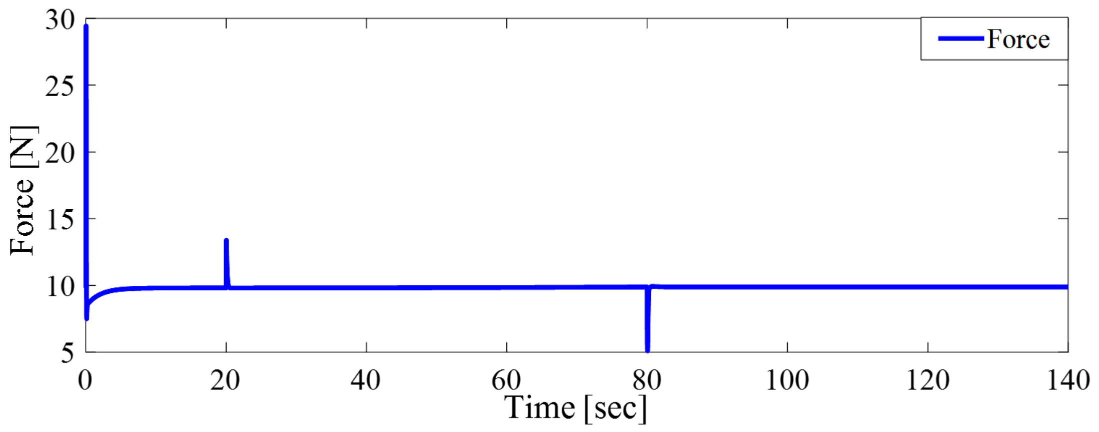


Figure 5.4.Force of the propeller.

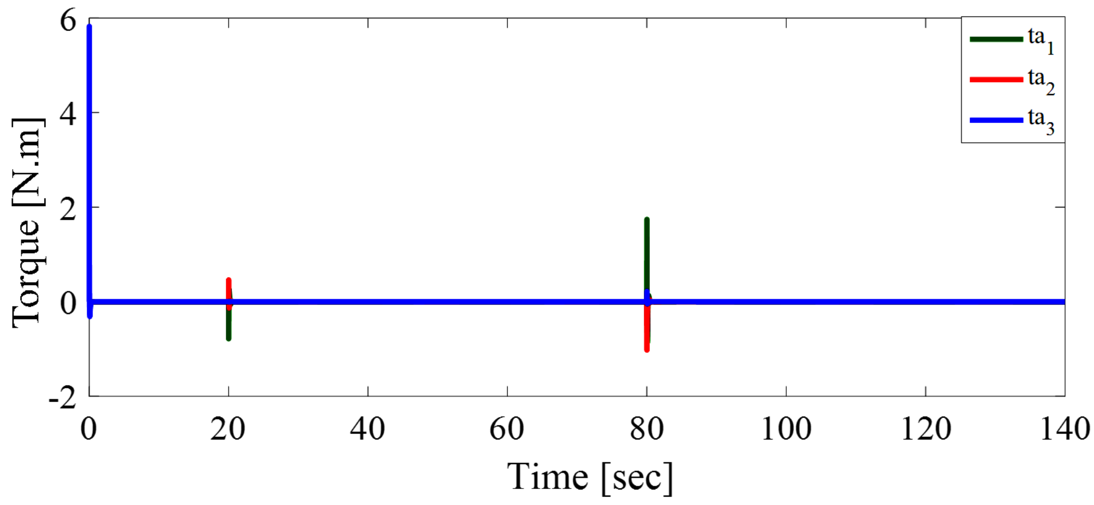


Figure 5.5. Torque of the quadrotor.

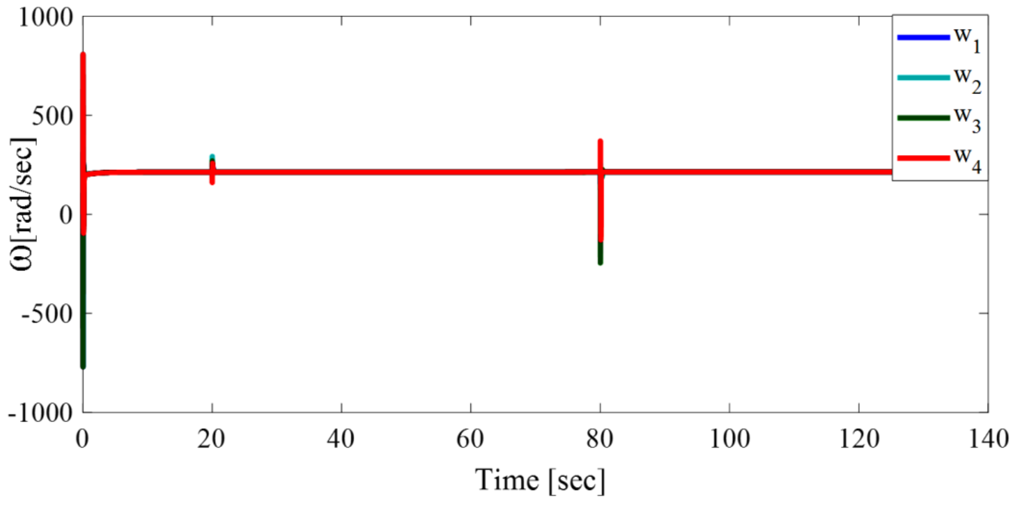


Figure 5.6. Angular velocity of the propellers.

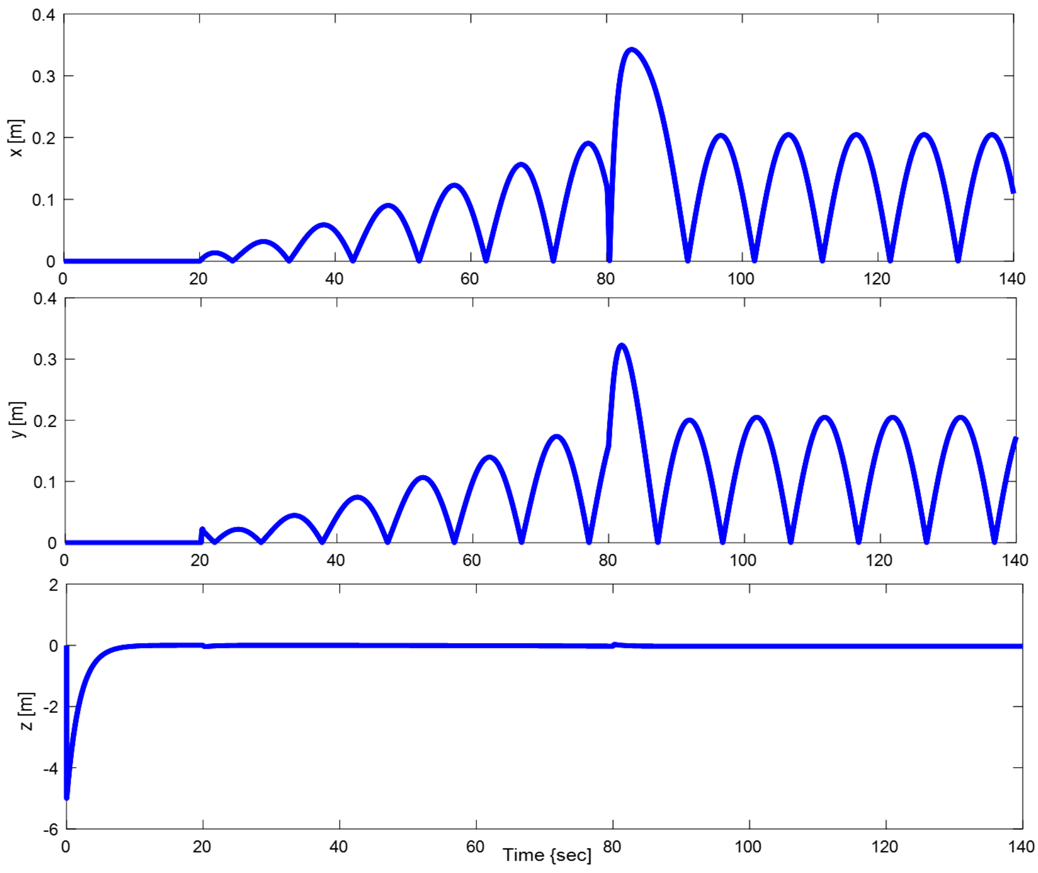


Figure 5.7. Error position of the quadrotor.

5.2. Simulation Result of CAD model for a quadrotor

In this section, thanks to the SimMechanics environment, the 3D quadrotor model is completely simulated as a real one, and the model has been validated using Simscape tools by creating a control algorithm in terms of stability and the ability of the quadrotor to fly at a certain altitude. Where a helix path, hover path, and straight path was created in the CAD model. Also, a comparison was made in the case of the adaptive controller and PID controller for paths. Two distinct control approaches are employed in this work to stabilize the quadrotor system's motion. First sliding mode controller with the adaptive controller. The second is the sliding mode controller with the PID controller. Parameters of 3D CAD of quadrotor illustrated in the **table 4.2** mentioned previously.

5.2.1 Simulation result helix path of 3D CAD model of quadrotor

In this section, the helix path generated in Simulink is presented using equation 4.1that expresses the path and also using sliding mode control with adaptive control and sliding mode control with PID control. Where a comparison is made between the above two control methods.

The simulation results are display in **figure (5.8 – 5.19). Figure 5.8** shows the trajectory of the helix, and it was observed that the quadrotor drive is able to follow the required flight path correctly. **Figure 5.9**, shows the change in quadrotor position (x, y, z) values with acceptable desired response time. **Figure 5.10**, also demonstrates excellent tracking of the attitude reference trajectory, and it can be noted that the yaw response is different from the pitch and roll response, where the yaw response approaches zero.

Furthermore, **figures 5.9 and 5.10** illustrate a comparison of tracking results in the position and attitude subsystems produced with the adaptive sliding mode controller with the adaptive controller. It can be seen from **figure 5.11** that the force starts from zero then increases till it is fixed at 100s. **Figure 5.12** shows the angular velocity where the four propellers' angular velocity is required to balance the acceleration due to gravity, allowing the quad to hover instantaneously.

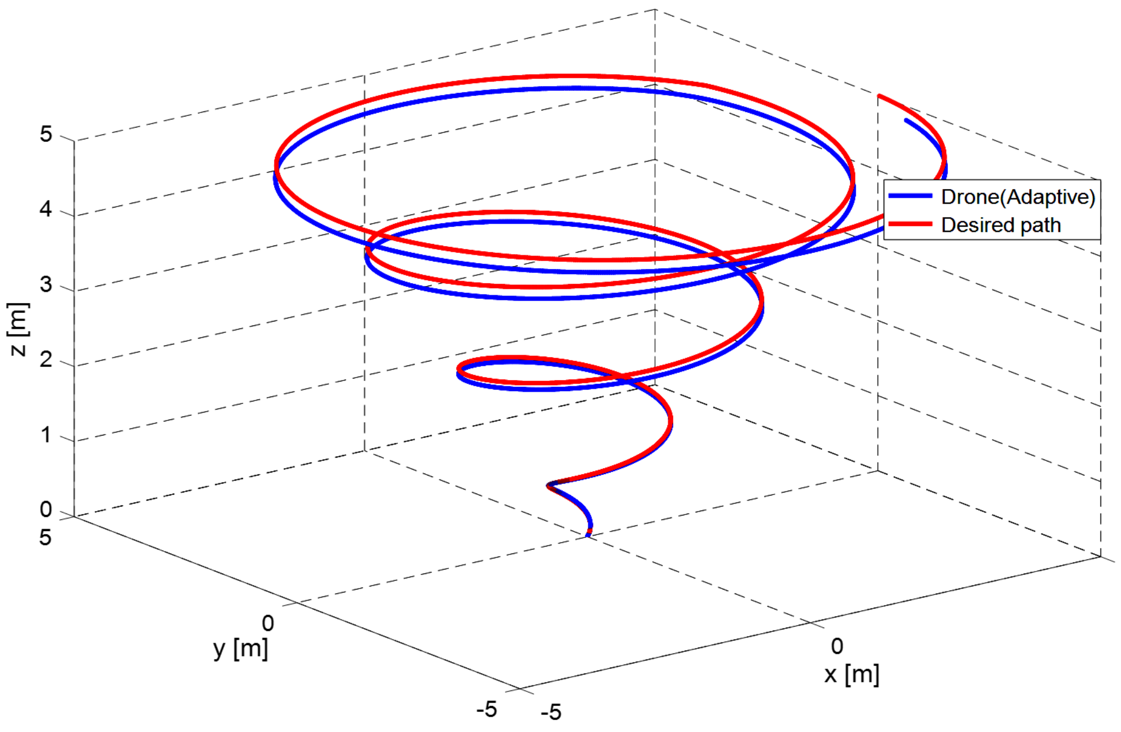


Figure 5.8.3D Conical helix trajectory with the adaptive controller.

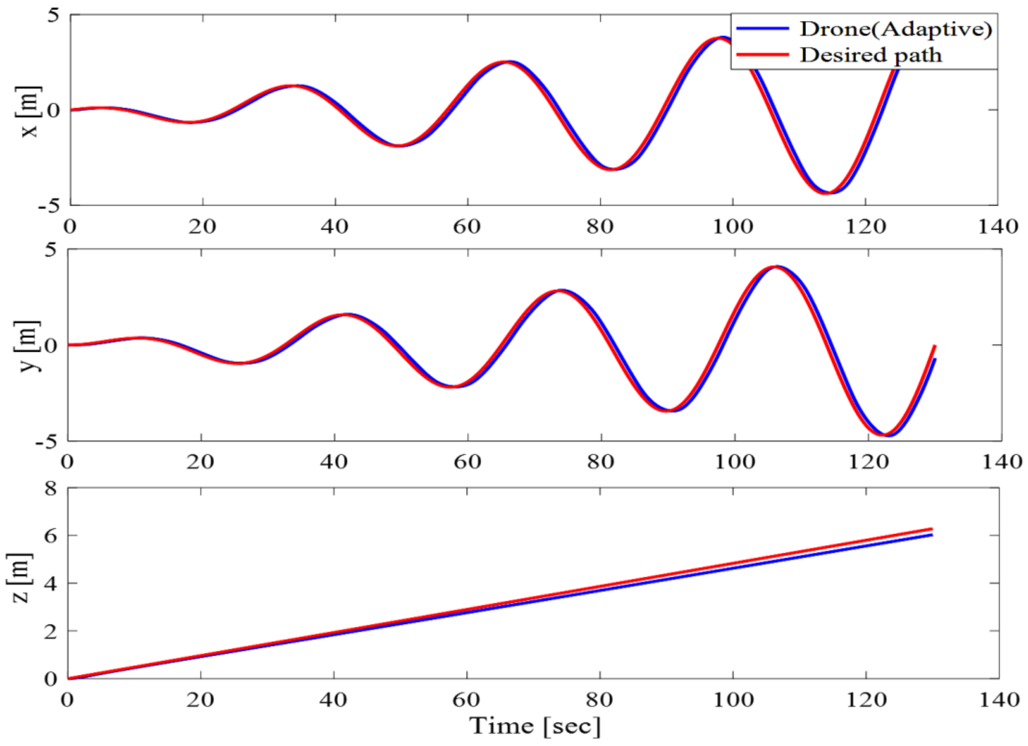


Figure 5.9.Quadrotor position of helix path in$(x, y\,, \mathrm{z})$with adaptive controller.

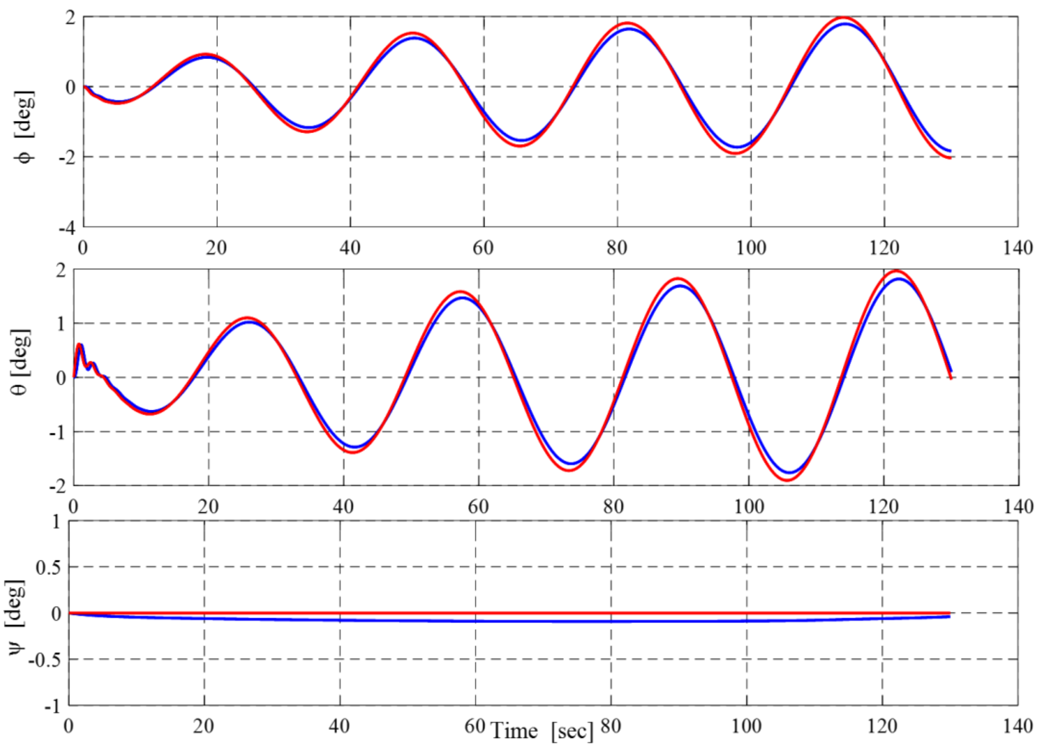


Figure 5.10. Attitude angles of helix path in (ϕ, θ, Ψ) with the adaptiv controller.

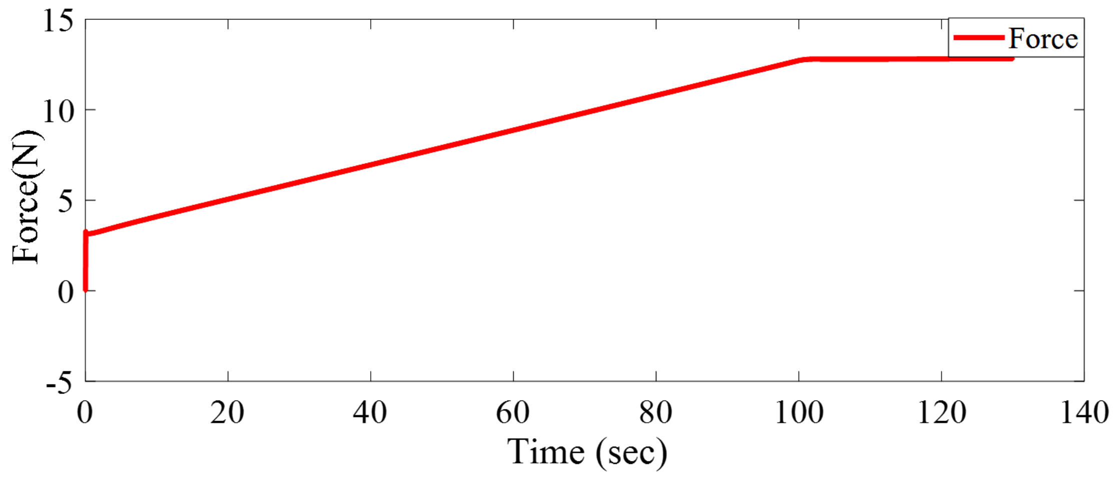


Figure 5.11. Force of helix path with the adaptive controller.

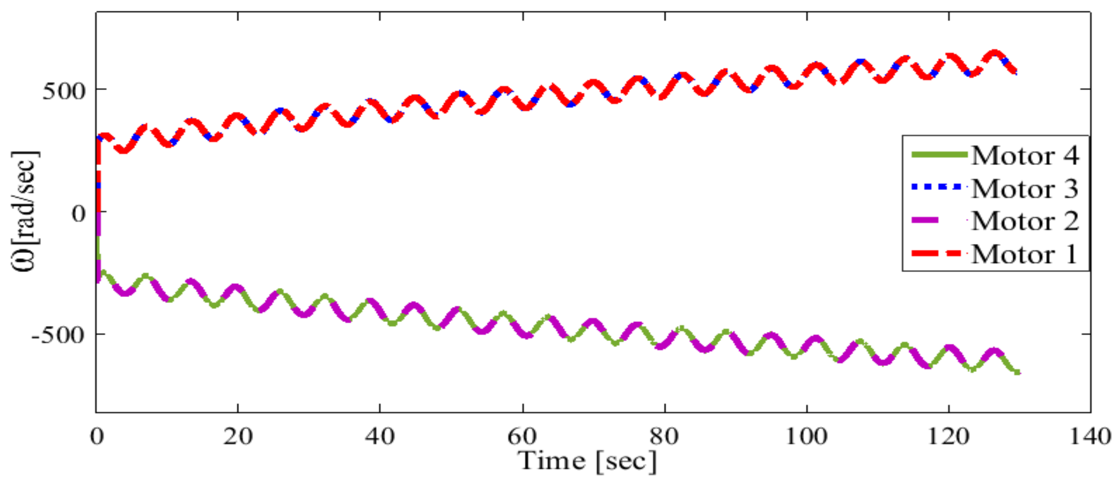


Figure 5.12. The angular velocity of helix pathwith the adaptive controller.

Figure 5.13 shows the trajectory of the quadrotor by using the sliding mode controller with the PID controller. Quadrotor position in (x, y, z) is shown in **figure 5.14** for the helix trajectory. It can be observed from **figure 5.13 and figure 5.14** that the quadrotor is capable of monitoring the appropriate fly path. **Figure 5.15** demonstrates good tracking of the conical helix trajectory's attitude. **Figure 5.14** and **Figure 5.15** provide a comparison between the tracking results in the position and attitude obtained by using the sliding mode controller with a PID controller and the desired path. **Figure 5.16** expresses also angular velocity with the PID controller, where it can be depicted that motor 4 and motor 2 rotate contrariwise to motor 3 and motor 1.

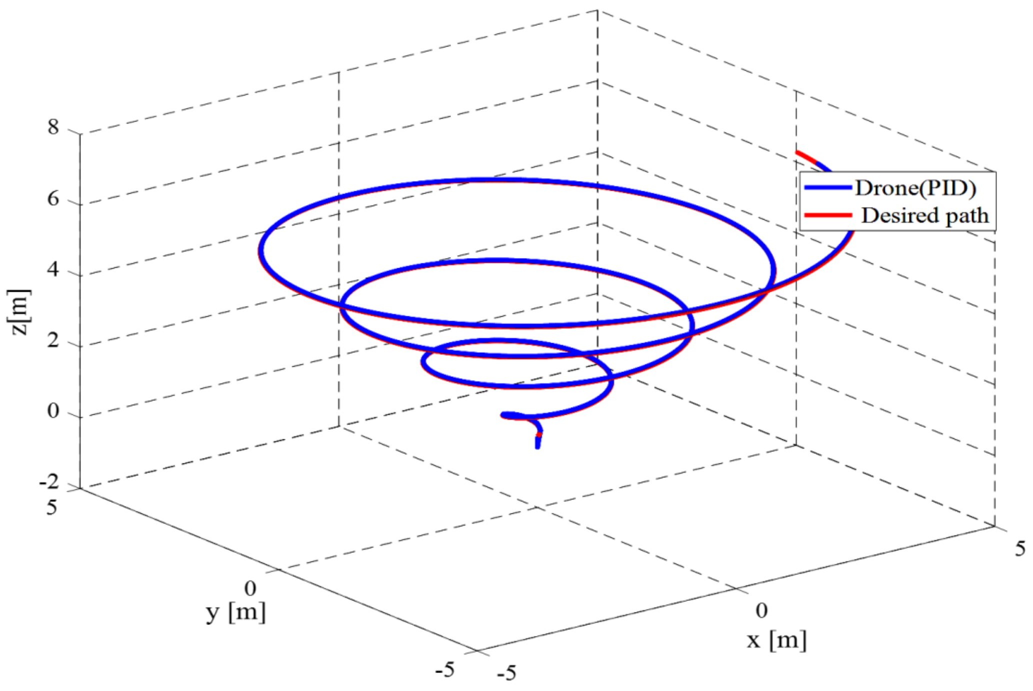


Figure 5.13. 3D Conical helix trajectory with PID controller.

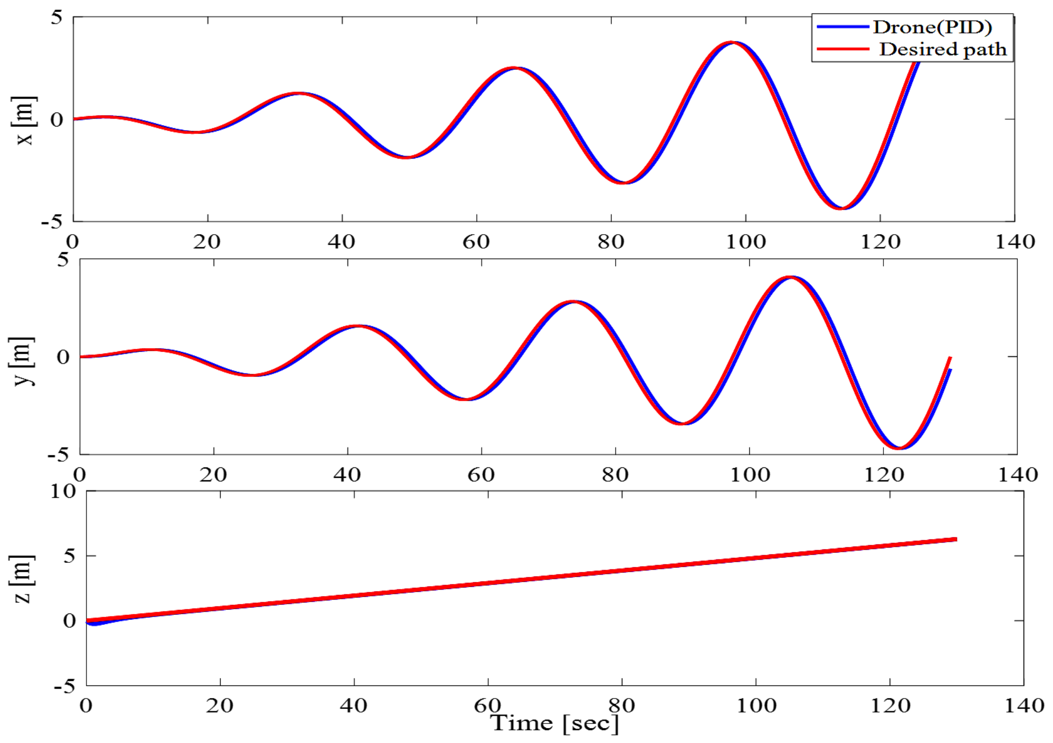


Figure 5.14. Quadrotor position of helix trajectory in$(x, y, z)$with PID controller .

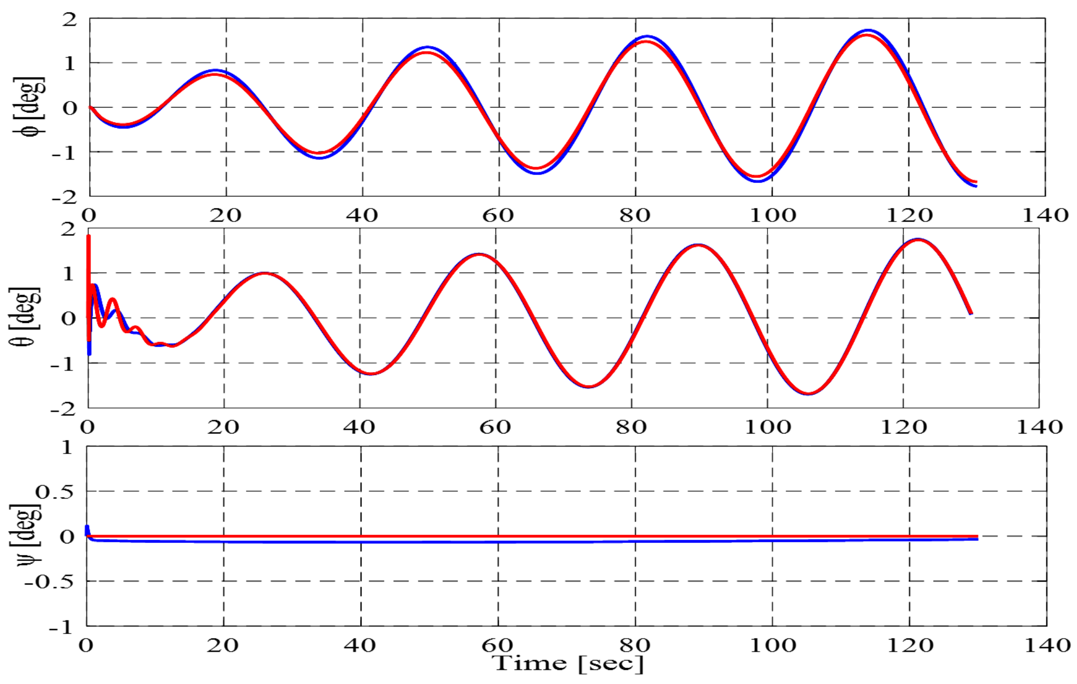


Figure 5.15.Attitude angles of of helix trajectory in (ϕ, θ, Ψ) with PID controller.

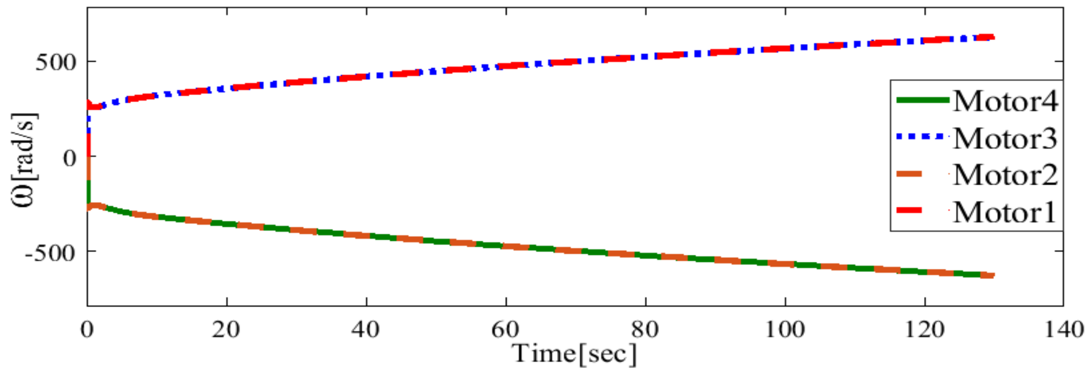


Figure 5.16.The angular velocity of helix trajectory with PID controller.

It can be seen from **figure 5.17-5.19**, that shows error when the adaptive controller and PID are used. The PID controller outperforms the adaptive controller. Also, it can be noted that the magnitude of mean square root for position (x, y, z) is 0.266,0.262,0.1512. When comparing with the mathematical model, it was noticed that the error rate of the mathematical model is less as explain in **table 5.1**

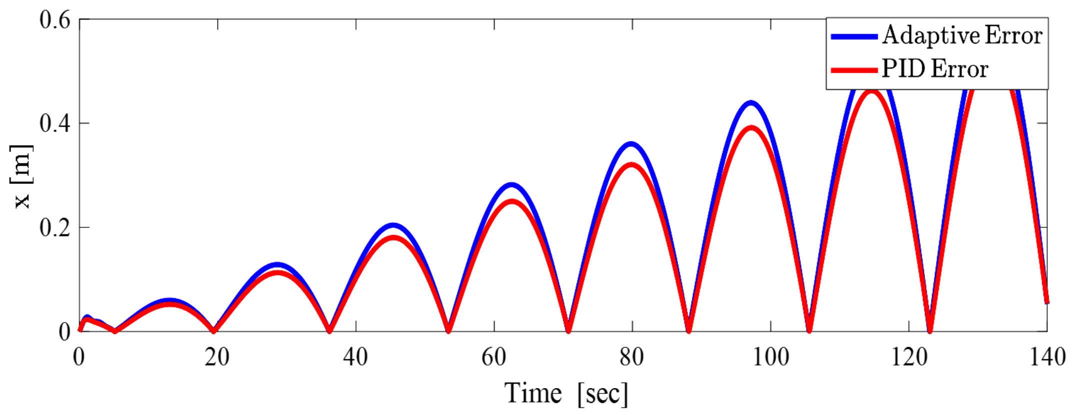


Figure 5.17. Error position of helix trajectory x between Adaptive and PID.

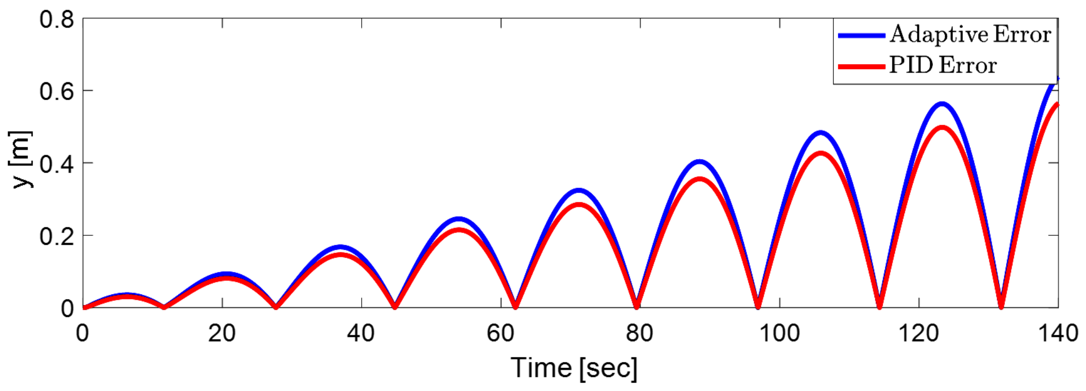


Figure 5.18. Error position of helix trajectory y between Adaptive and PID.

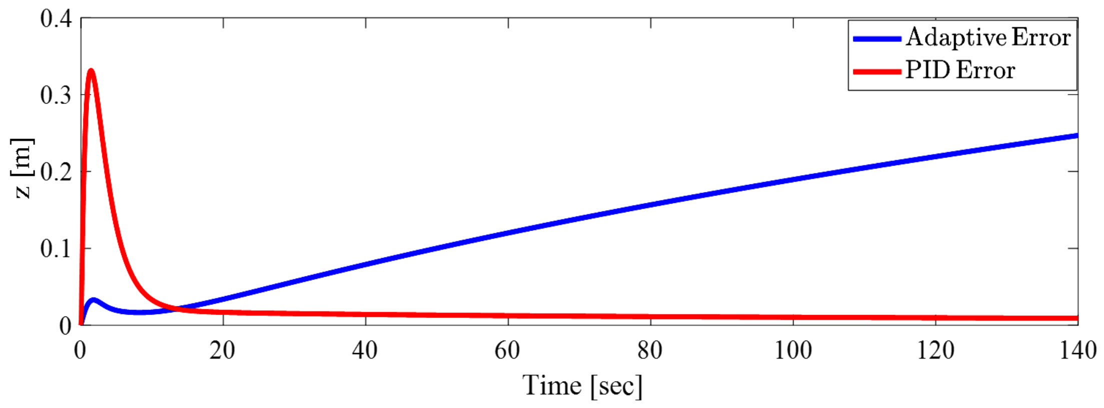


Figure 5.19. Error position of helix trajectory z between Adaptive and PID.

5.2.2. Simulation results straight path of 3D CAD model of quadrotor

In this section, the straight path generated in Simulink is presented using an equation 4.2 that expresses the path and also using sliding mode control with adaptive control and sliding mode control with PID control. where a comparison is made between the above two control methods.

The simulation results are displayed in **figure (5.20 – 5.32). Figure 5.20** shows the trajectory of the straight, and It is apparent that the actual trajectory clearly follows the reference trajectory despite the poorly modelled rotation of the robotic manipulator.

Figure 5.21, shows the change in quadrotor position (x, y, z) values with acceptable desired response time .**Figure 5.22**, also demonstrates excellent tracking of the attitude reference trajectory. Furthermore, **figures 5.21 and 5.22** illustrate a comparison of tracking results in the position and attitude subsystems produced by the adaptive sliding mode controller with the adaptive controller. It can be seen from **figure 5.23** that the force starts from zero then increase to 2sec and remains fixed at 100s . **Figure 5.24** shows the angular velocity where the four propellers' angular velocity is required to balance the acceleration due to gravity which allowing the quad to hover instantaneously.

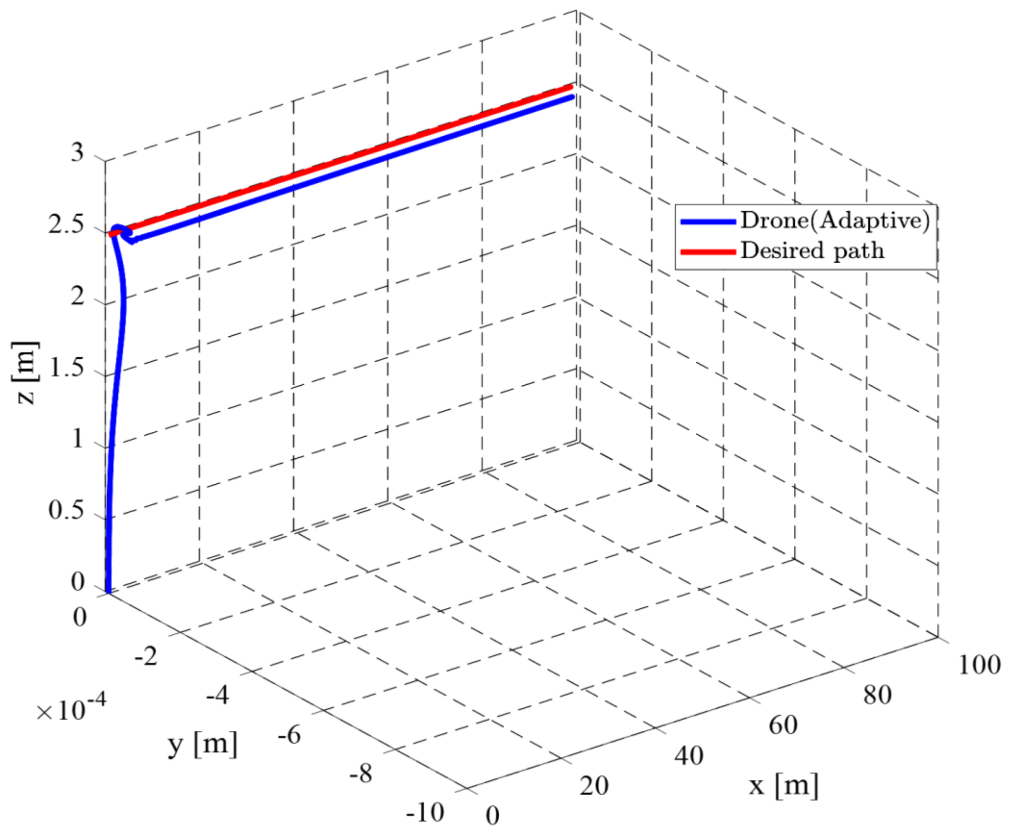


Figure 5.20.3D straight trajectory with the adaptive controller.

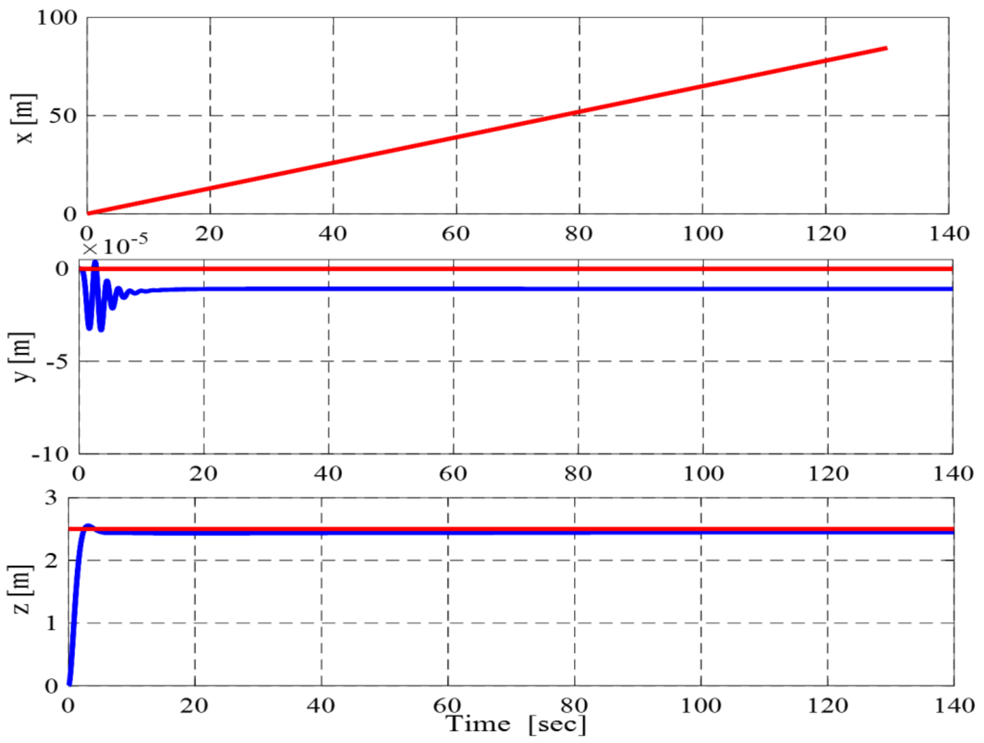


Figure 5.21. Quadrotor position of straight trajectory in (x, y, z) with adaptive controller.

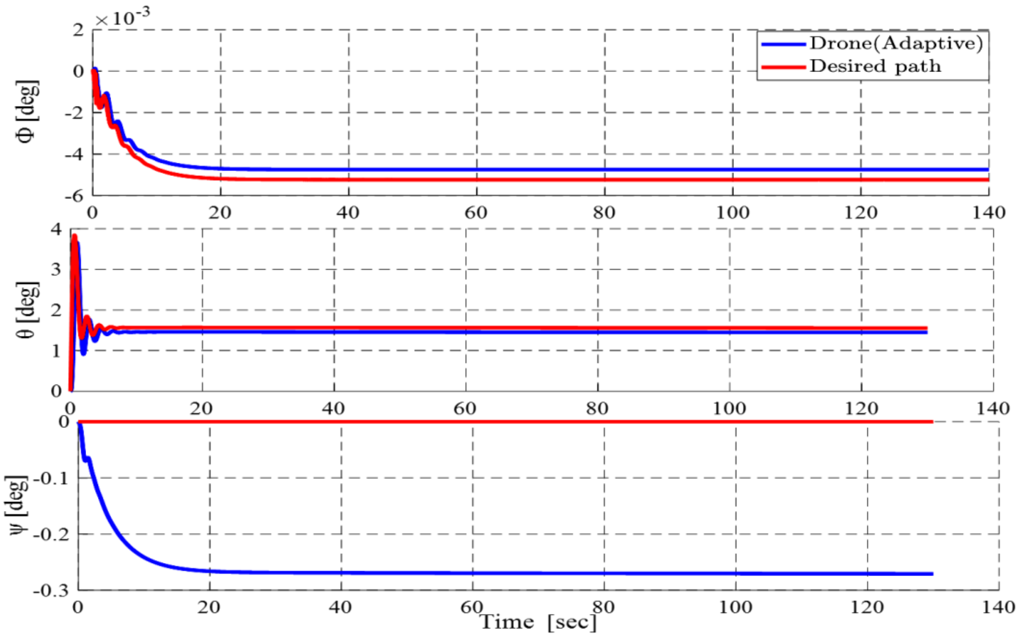


Figure 5.22.Attitude angles of straight trajectory with the adaptive controller.

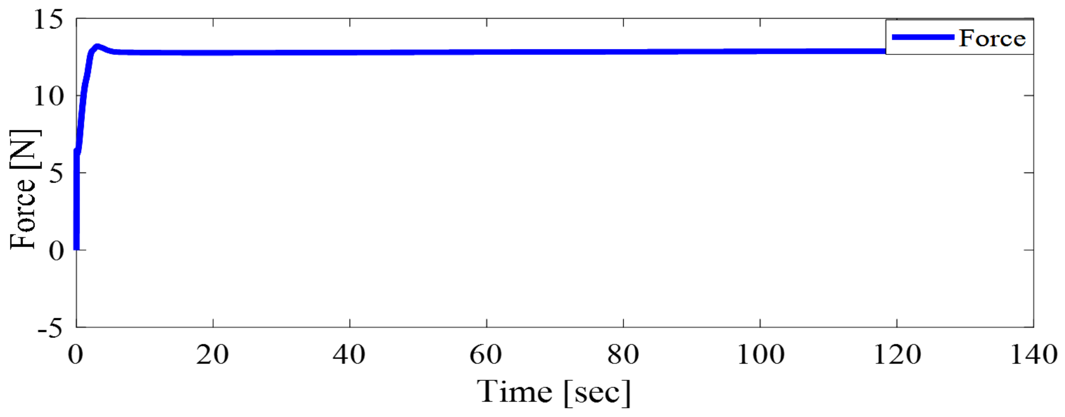


Figure 5.23. Force straight trajectory with the adaptive controller.

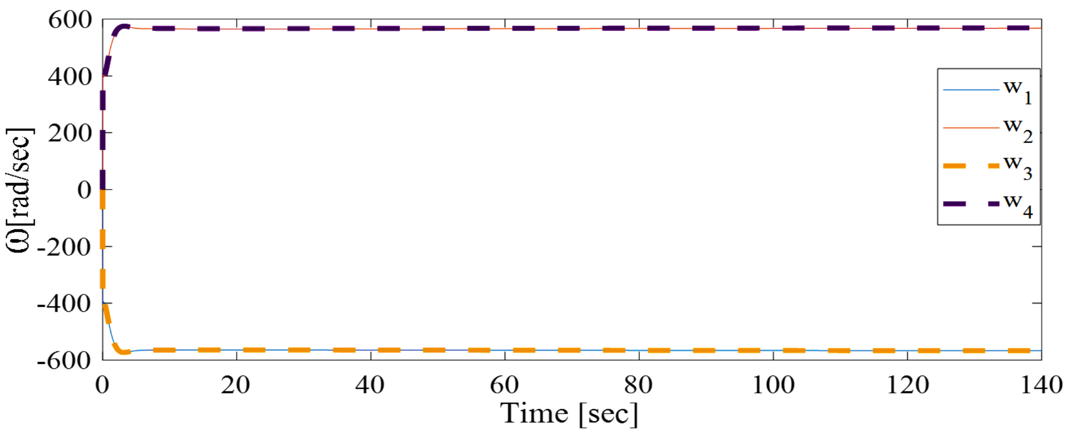


Figure 5.24.The angular velocity straight trajectory with the adaptive controller.

Figure 5.25 shows the trajectory of the quadrotor by using the sliding mode controller with the PID controller .Quadrotor position in (x, y, z) is shown in **figure 5.26** for the straight path. It can be observed from **figure 5.25** and **figure 5.26** that the quadrotor is capable of properly monitoring the appropriate fly path. **Figure 5.27** demonstrates good tracking of the straight trajectory's attitude. **Figure 5.26** and **Figure 5.27** provide a comparison between the tracking results in the position and attitude obtained by using the adaptive sliding mode controller with a PID controller and the desired path. **Figure 5.28** shows the force starts increase to 2sec and remains fixed at 100s.**Figure 5.29** expresses also angular velocity with the PID controller.

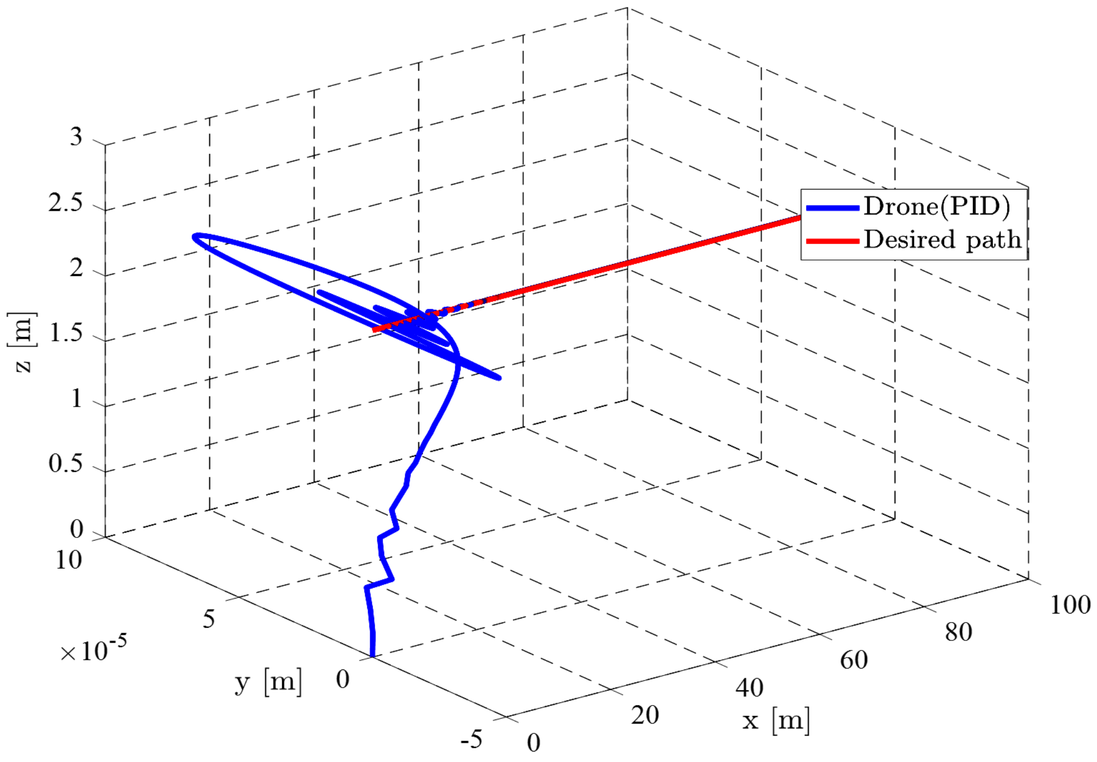


Figure 5.25. 3D straight trajectory of quadrotor with the PID controller.

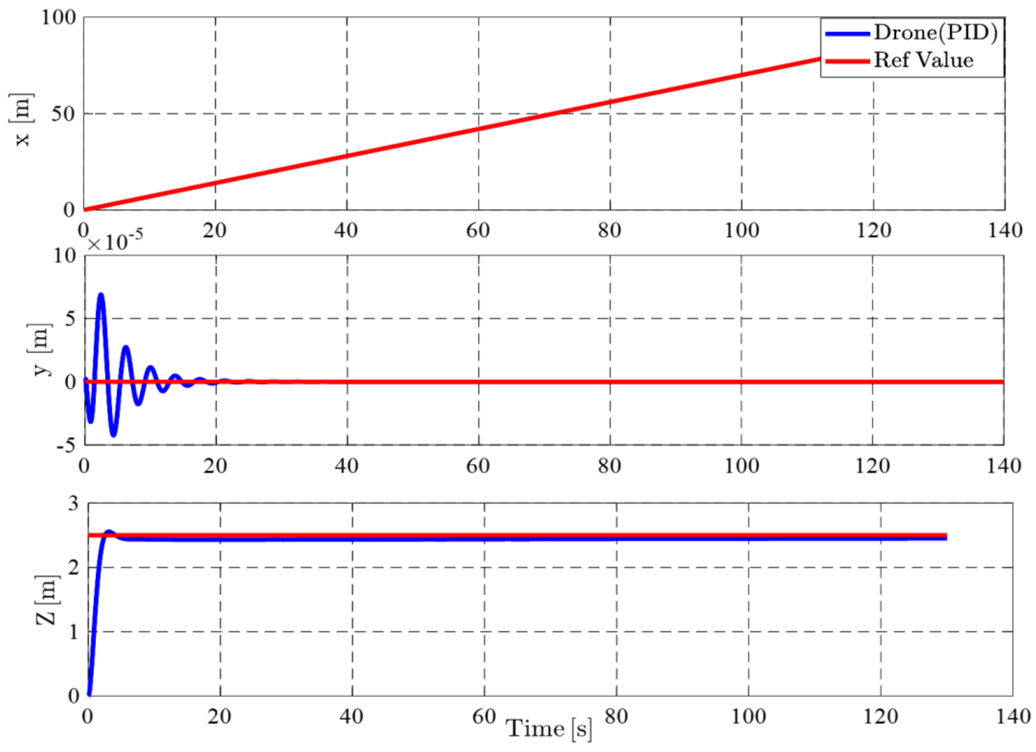


Figure 5.26. Quadrotor position of straight trajectory in (x, y, z) with PID controller .

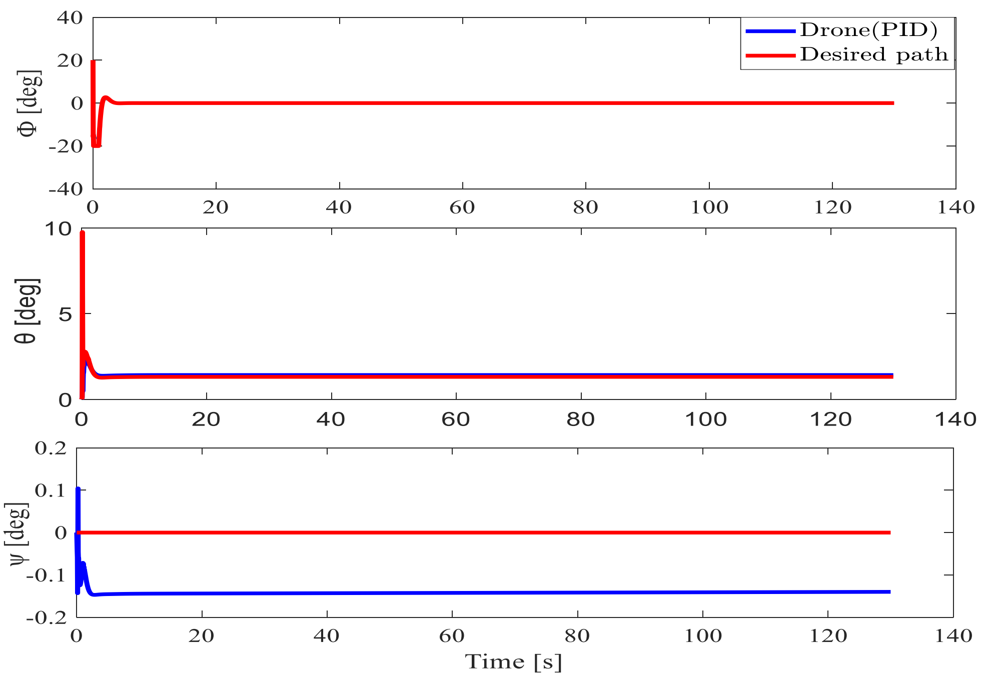


Figure 5.27.Attitude angles of straight trajectory (ϕ, θ, Ψ) with PID controller.

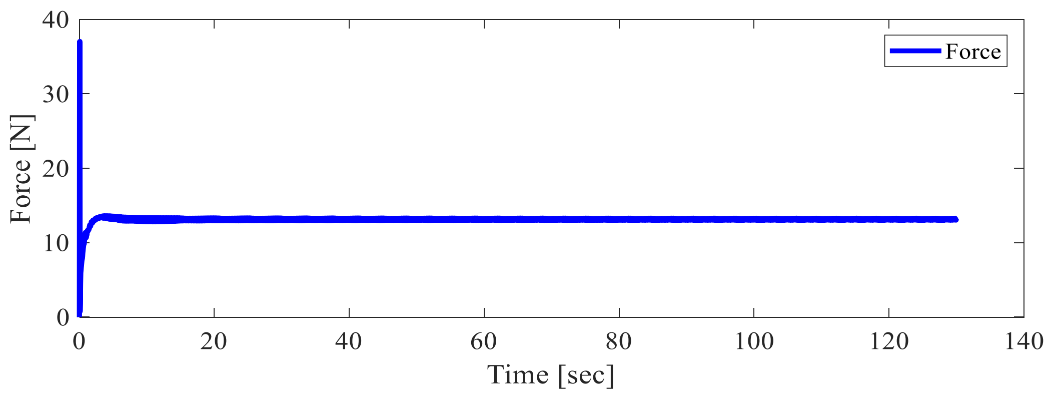


Figure 5.28. Force of straight trajectory (ϕ, θ, Ψ) with PID controller.

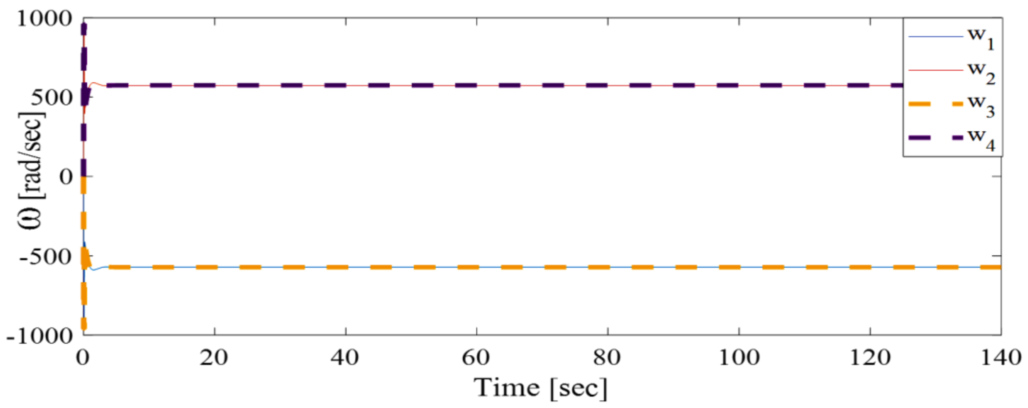


Figure 5.29. The angular velocity of straight trajectory with PID controller.

For comparison, the error can be seen in figure 5.30-5.32, when the adaptive controller and PID are used. The PID controller outperforms the adaptive controller.

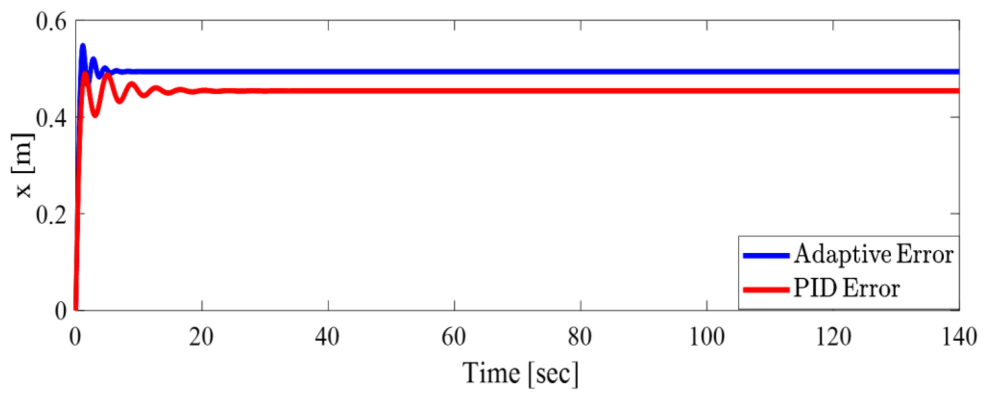


Figure 5.30. Error position x of straight trajectory between Adaptive and PID.

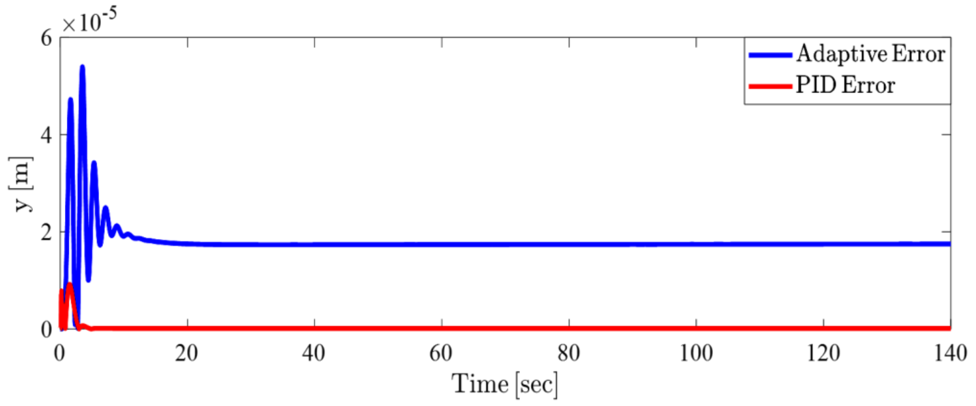


Figure 5.31. Error position y of straight trajectory between Adaptive and PID.

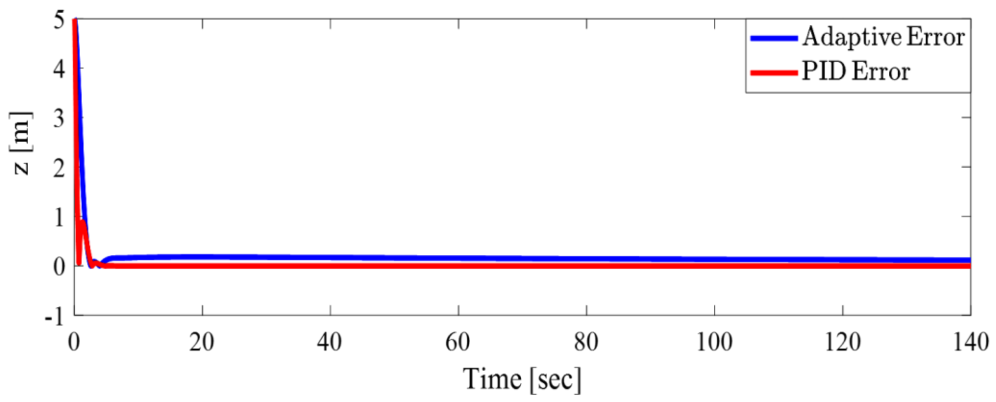


Figure 5.32. Error position z of straight trajectory between Adaptiv and PID.

5.2.3 Simulation result to hover path of 3D CAD model of quadrotor

In this section, the hover path is shown using sliding mode control with adaptive control. A comparison was made between the hover path model original and the hover path of after its development. The simulation results are display in **figure (5.33 – 5.35). Figure 5.33 a** shows the position x of the hover path of the model original with a percentage of the overshoot 86% and settling time 51m.sec while **figure 5.33 b** shows the hover path of after its development so that the a percentage of the overshoot is 29%.

Figure 5.34 a shows a position y of the hover path of the model and the percentage of the overshoot is 72% while **figure 5.34 b** shows the hover path of after its development with a percentage of the overshoot 31%.

Figure 5.35 a shows position z of the hover path of the model and the percentage of the overshoot is 4 % While **figure 5.35 b** shows the hover path of after its development that give a percentage of the overshoot 4% and you can be see that overshoot of position z remain constant.

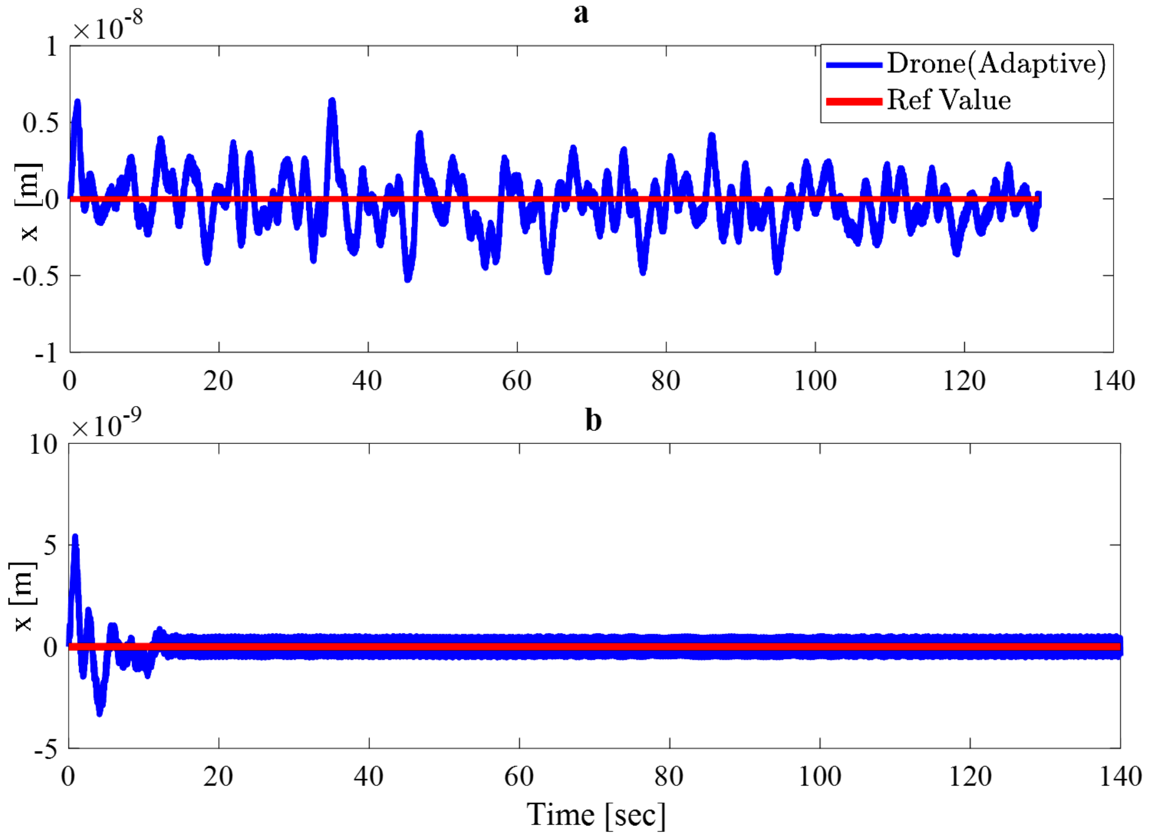


Figure 5.33. comparison between hover path of the position x.

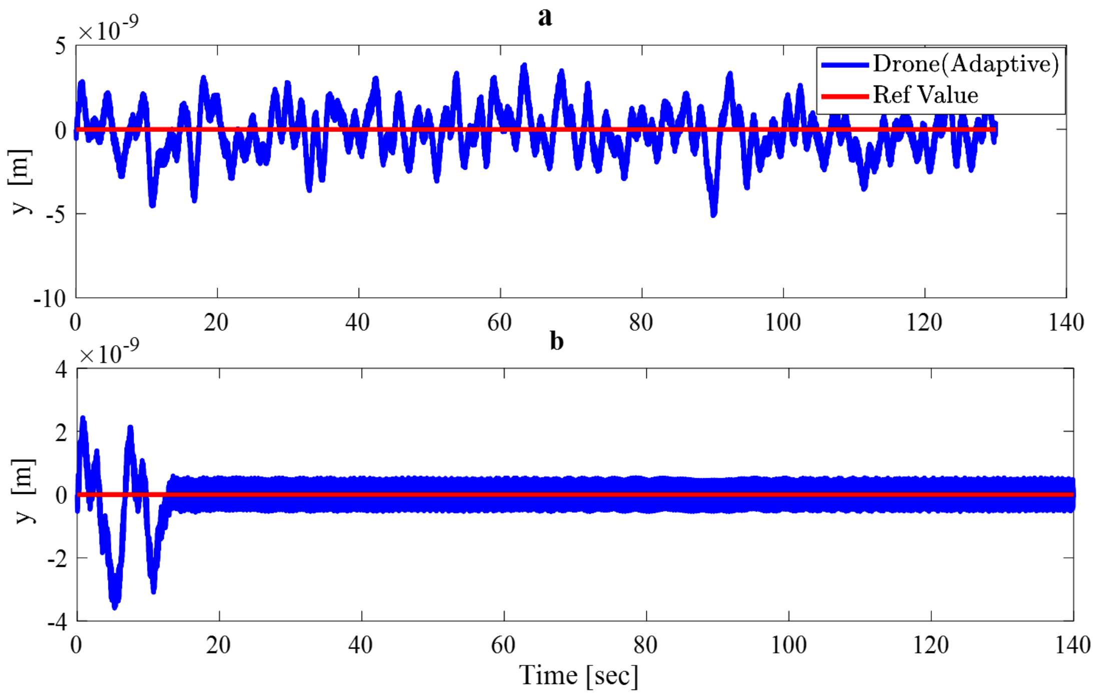


Figure 5.34 . comparison between hover path of the position y.

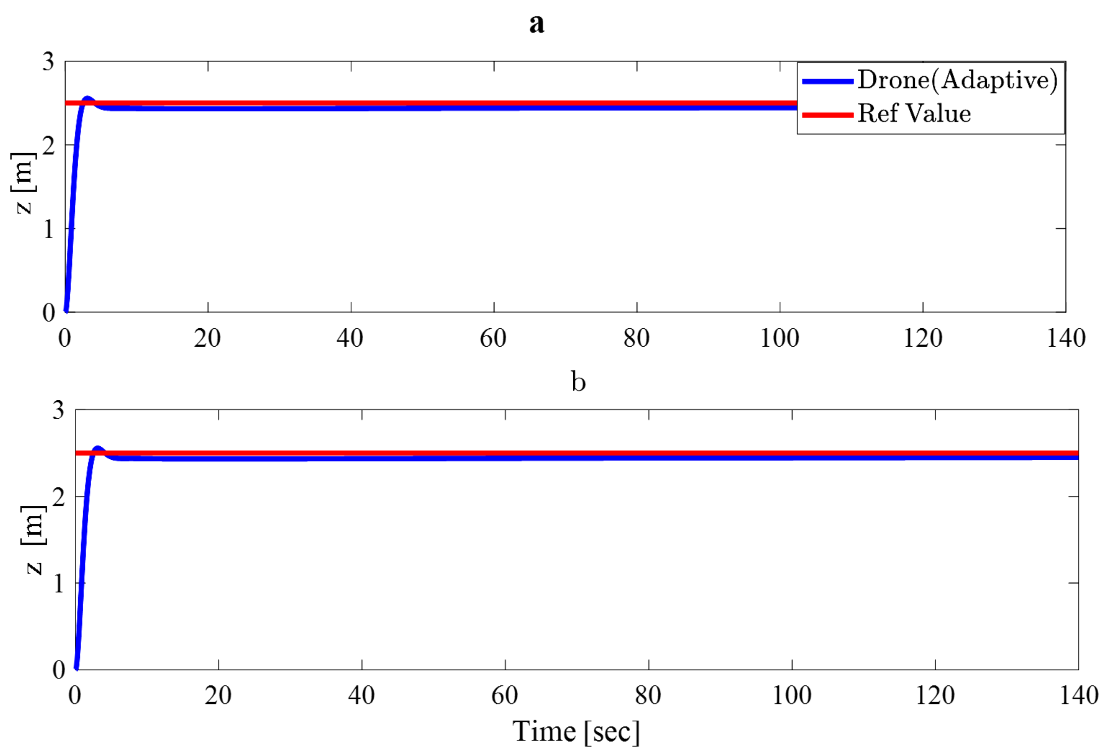


Figure 5.35. comparison between hover path of the position z.

5.3. Experiment Results

The aim of the laboratory experiments was to prove that AR. Drone can be controlled by the LABVIEW and made it follow a certain path correctly. Two methods were used to control the quadrotor by the joystick and the keyboard. So, three separate experiments are carried out by using AR. Drone for task positioning and tracking. The experiments are presented in the same way: with a brief description and graphics of the important variables.

5.3.1. Experiment 1: Hovering by joystick

Experiment 1, where to use a joystick and is depicted in **figure 5.36** creates turbulence throughout the flight, forcing the AR. Drone to drift away from its reference. The objective is to show that the system can maintain its position trajectory even when subjected to significant disturbances. The experiment was conducted at a time of 120 second and the desired results were achieved.

Figure 5.36 a shows 3D trajectory for the drone. **Figure 5.36 b** shows the positions of the drone in (x, y, z). Where as shown in the figure the values of x and y are zero and there is only a value for z. **Figure 5.36 c** shows the angles at which the drone is inclined, namely pitch, roll, and yaw, where it is noted that their values very few.

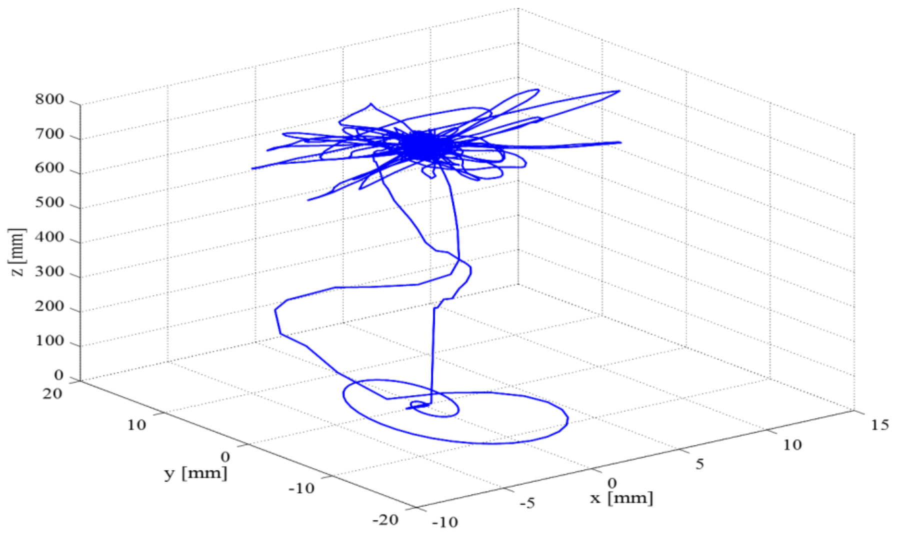


Figure 5.36a. Trajectory of the quadrotor.

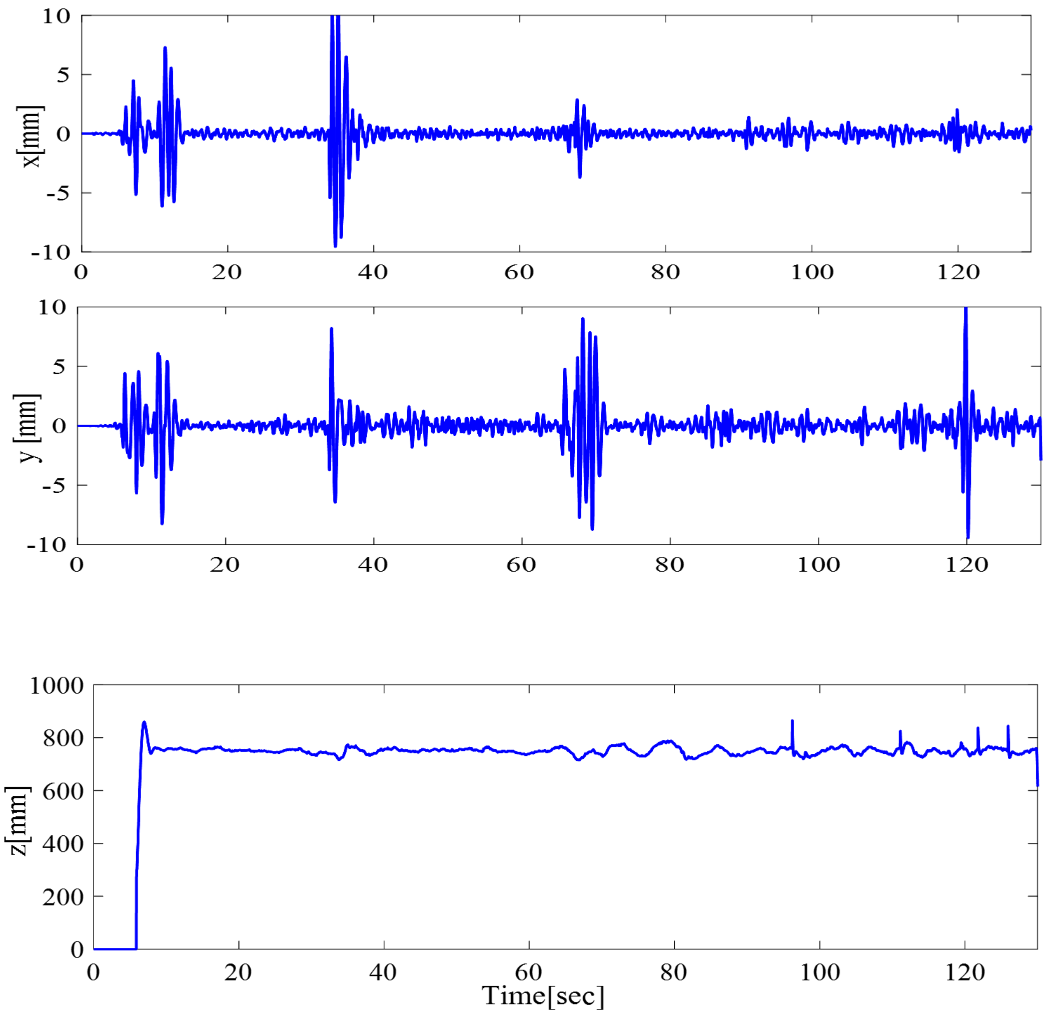


Figure 5.36 b. Positions of the quadrotor.

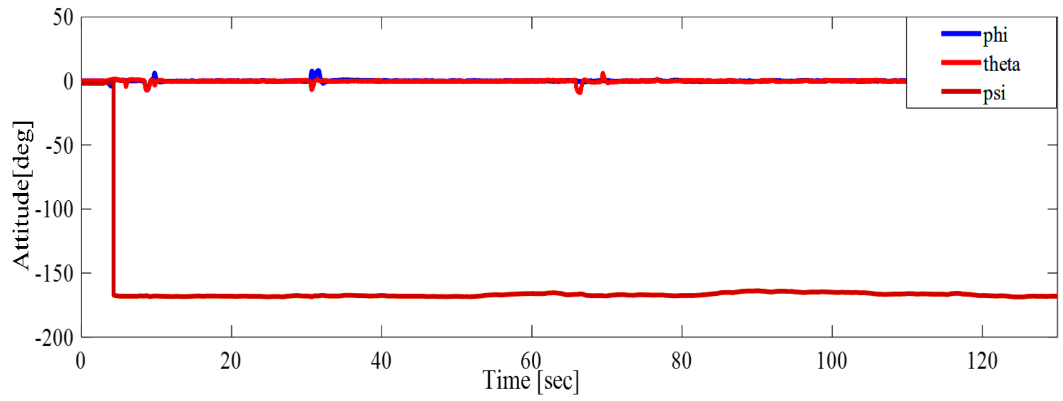


Figure 5.36 c.Atitude of the quadrotor.
Figure 5.36. Experiment 1- Hovering by joystick.

5.3.2. Experiment 2: Hovering by keyboard

Experiment 2, where to use a keyboard and is depicted in explaining **figure 5.37** **.**Throughout the flight, there is turbulence, which causes the AR. Drone going away from its starting position. The aim is to demonstrate that the system can retain its position trajectory despite substantial disruptions. The experiment lasted 120 seconds and produced the intended results.

Figure 5.37 a shows 3D trajectory for the drone. **Figure 5.37 b** shows the positions of the drone in (x, y, z). **Figure 5.37 c** shows the angles at which the drone is inclined, namely pitch, roll, and yaw, where it is noted that their values very few.

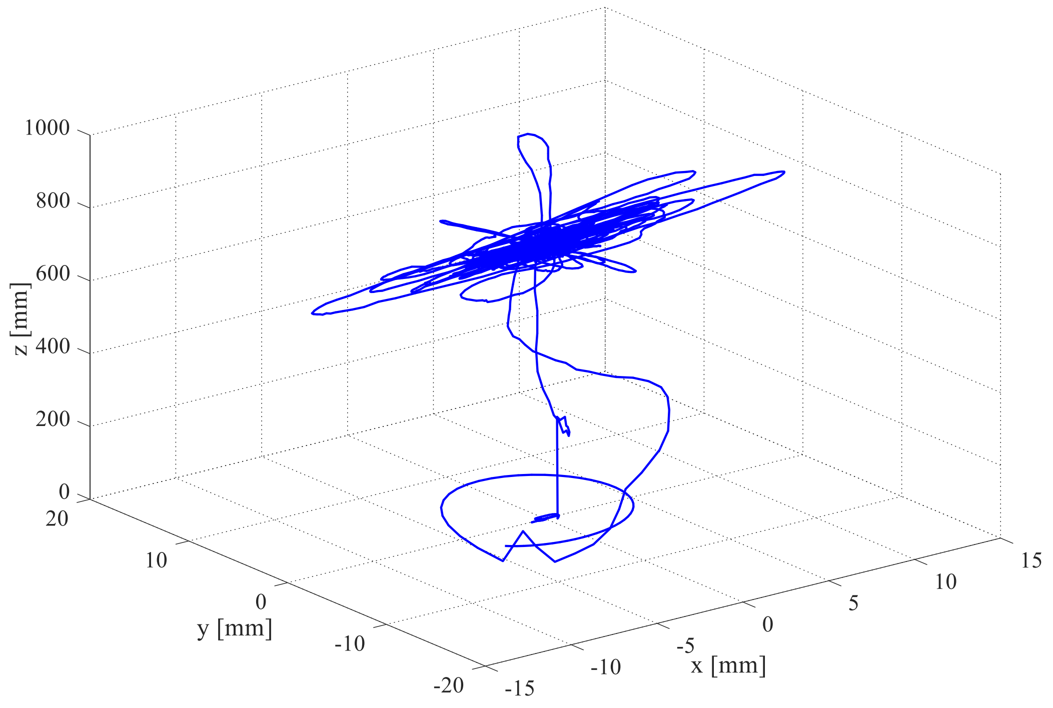


Figure 5.37a. Trajectory of the quadrotor.

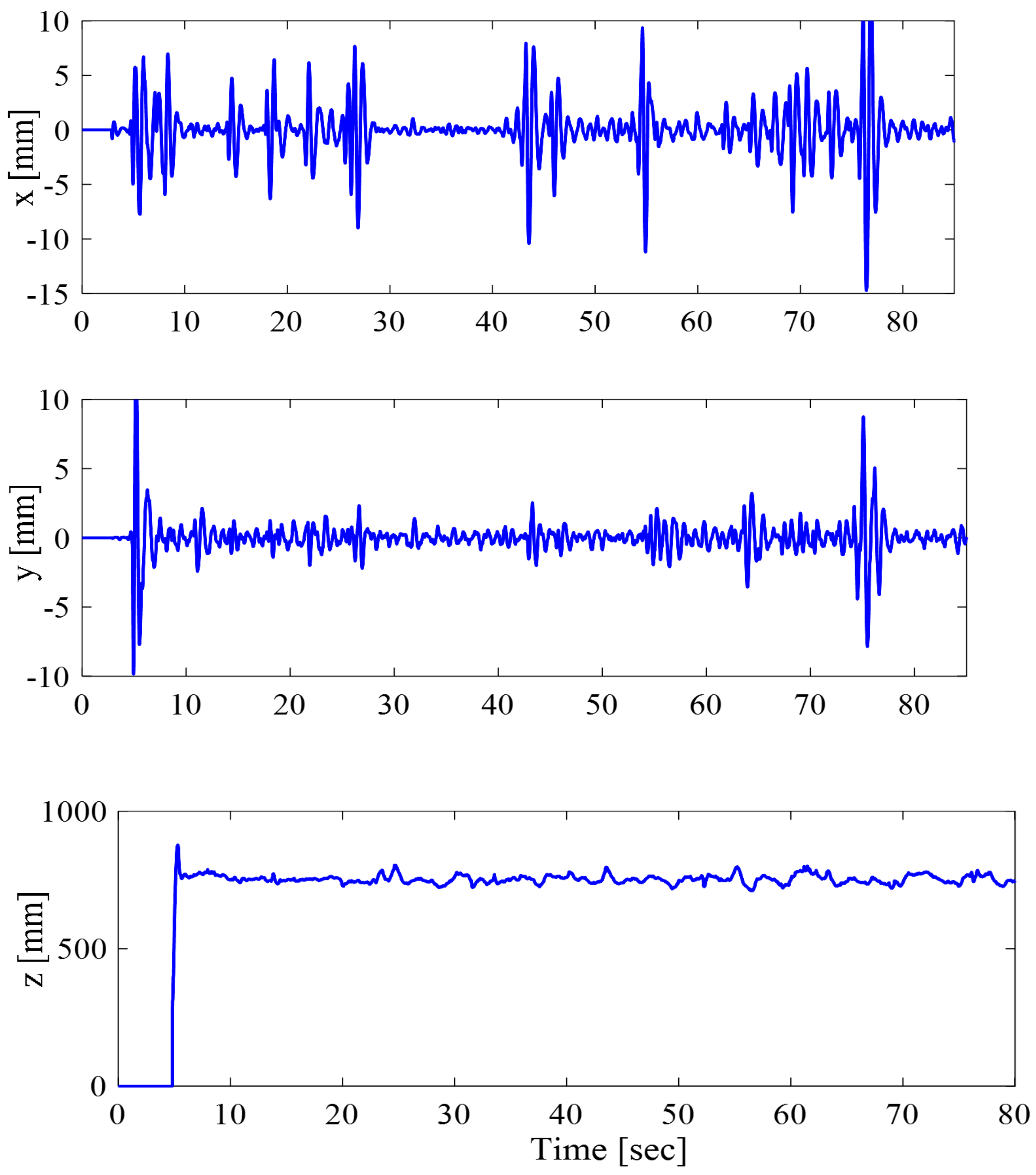


Figure 5.37 b. Positions of the quadrotor.

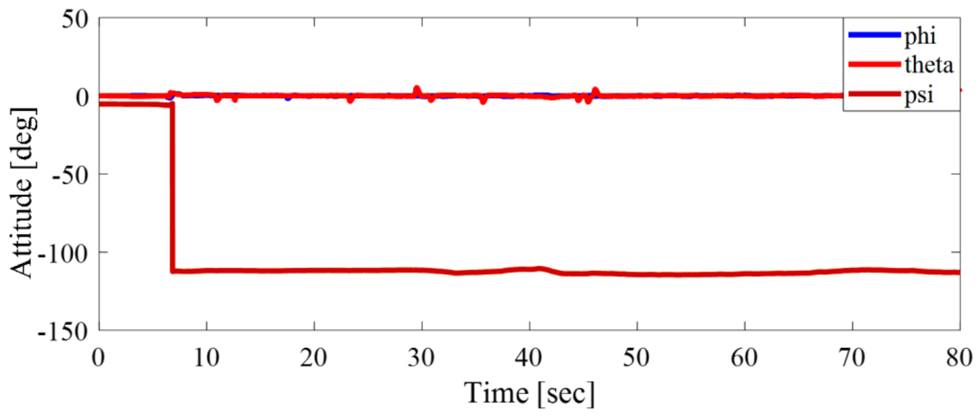


Figure 5.37 c. Attitude of the quadrotor.

Figure 5.37. Experiment 2- Hovering by keyboard .

When comparing the results of keyboard control with joystick control, it was noted that the error in keyboard control for position z is 0.645, while error in joystick control is 0.606. Therefore, it can be considered that controlling the joystick and keyboard gives similar results.

When comparing experimental results with simulation results. In start, it must be position in hover path $x = 0, y = 0, z = height$, in experiment result was position $x = 10 \times 10^{-3}\ m, y = 10 \times 10^{-3} m$ while in Simulink result $x = 5 \times 10^{-9} m, y = 3 \times 10^{-9} m$. Also, magnitude of root square mean for position z in experiment is 0.6 while in simulation 0.03(**as show in table 5.1**). Therefore, it can be said that the trajectory quadrotor application is better in simulation.

5.3.3 Experiment 3: path straight by keyboard

Experiment 3 is creating a straight path using a keyboard as shown in **figure 5.38**. **Figure 5.38 a** shows the 3D trajectory of the drone in a straight path. **Figure 5.38 b** shows the positions of the drone in (x, y, z).

It was also noticed when comparing the simulation path that there is some dispersion in the results of the operation resulting from the vibration of the aircraft during the flight. But the results are similar, which indicates the success of the experiment.

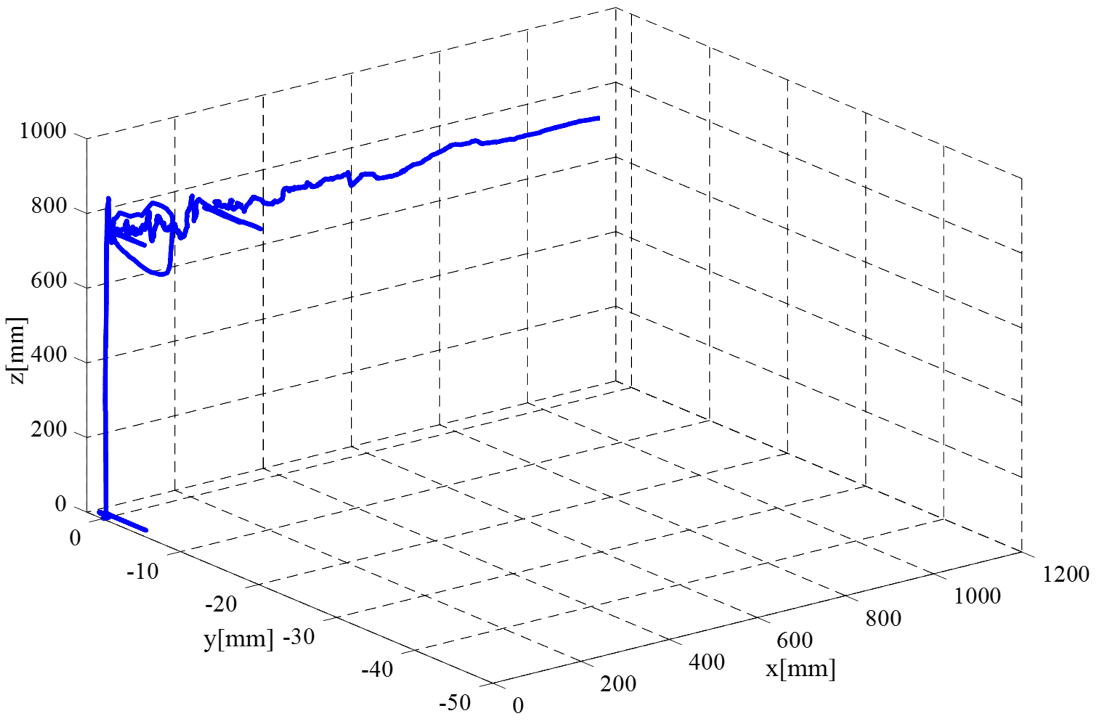
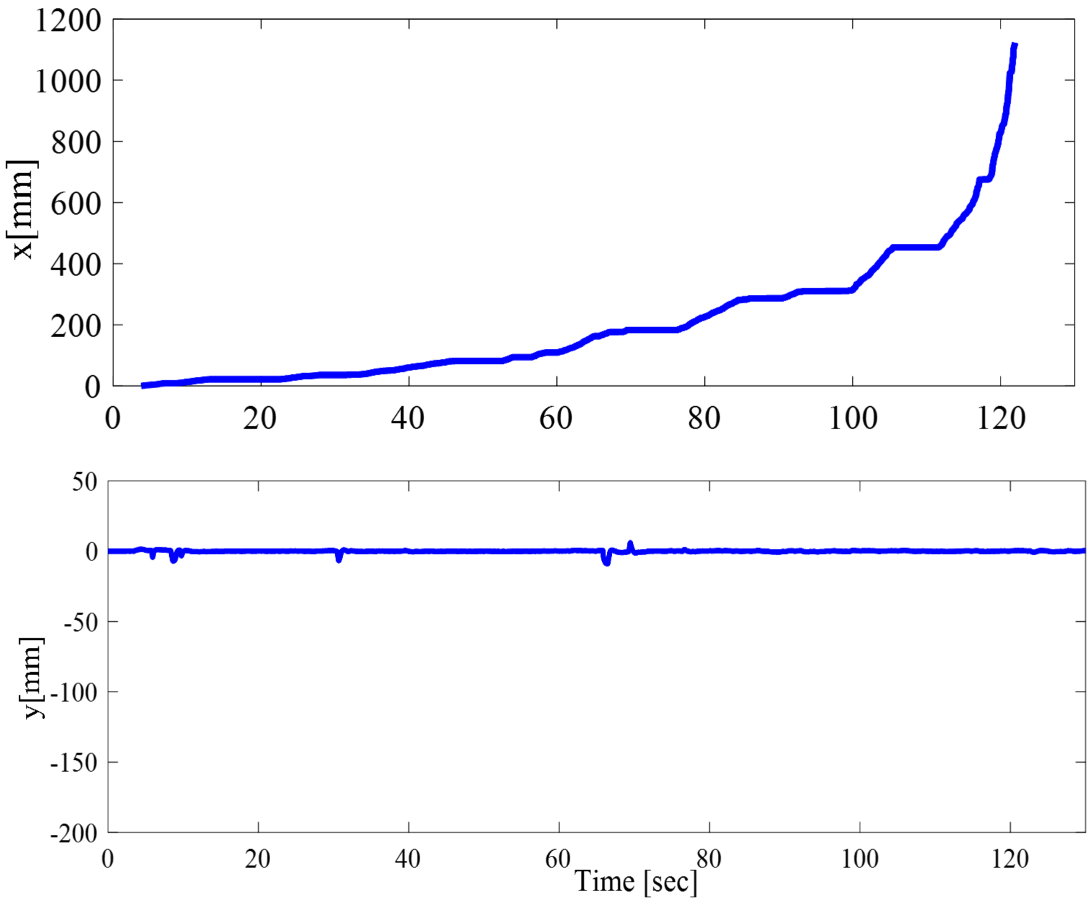


Figure 5.38 a. 3D straight trajectory of quadrotor with the PID controller.



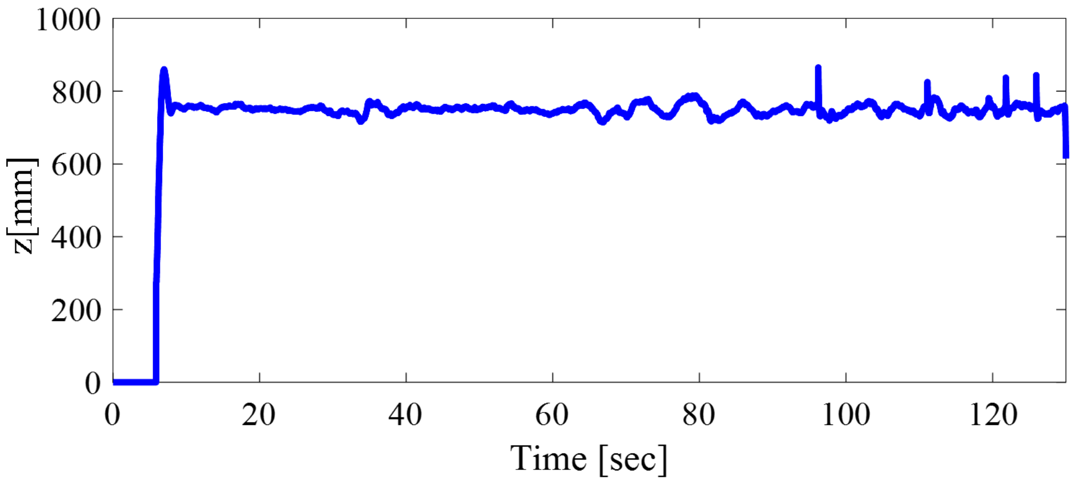


Figure 5.38 b. Positions of the quadrotor.
Figure 5.38: Experiment 3-Straight by keyboard.

5.4. Comparative Analysis

In this section, the performance of the experiment by using (a keyboard and joystick) and the theoretical work (dynamic model and CAD model) are compared by mean square root value. The results are summarized in Table 5.1 and 5.2.

Table 5.1: Error for theoretical work

	X[m]	Y[m]	Z[m]
Mathematical Model (helix path)	0.1258	0.1142	0.4172
CAD Model (helix path)	0.2660	0.262	0.1512
CAD Model (straight path)	0.4931	1.8134e-05	2.3438
CAD Model (hover path)	4.6886e-10	4.2006e-10	0.0314

Table 5.2: Error for experiment work

	X[m]	Y[m]	Z[m]
Hover by joystick	-	-	0.606
Hover by keyboard	-	-	0.645

Chapter Six: Conclusions and Recommendations

CHAPTER SIX

CONCLUSIONS AND RECOMMENDATIONS

A comprehensive numerical study of the quadrotor was conducted. The quadrotor is controlled by implementing a mathematical model and CAD development. Several paths are performed to reach the aircraft's stability. A type of quadrotor (AR. Drone) has been studied and tested. The results obtained led to the following concluding observations:

1. Sliding mode control with PID control outperforms sliding mode control with adaptive controller in CAD model
2. The PID control system for a quadrotor was designed in Simulink / MATLAB using a dynamic model to track trajectory. The results showed the correct path tracking and the success of the proposed controller.
3. The error rate when executing the helix path of the dynamic model is 0.1258 as compared to the CAD model is 0.2670. This proves that executing the path using the dynamic model is better.
4. Reducing overshoot for hover path of 3D CAD model, where it was 86% and become the percentage of overshoot is 29%.
5. It has been proven that controlling the drone using the keyboard and the joystick gives similar results. Where the mean square root value for the keyboard in hovering is 0.645 while it in joystick control is 0.606.

6. It can be said that the quadrotor trajectory application is better in simulation. Although there is a slight difference in the error rate, this indicates the success of the experiments

6.1. Recommendations:

The following points can be studied as future works:

1. Using a low-cost drone, a specific path can be applied to reach areas that are not accessible, such as forest fires, rescue, or espionage .
2. LabView applications can be used for image processing.
3. A camera can be used not only to determine the position but also for a lot of other purposes. For example, it can be required for the tracking of mobile targets or for environment mapping.
4. A specific path can be written and implemented using Lab View to track an object and use it for bushfire rescue

References

[1] Czerwiński, E., Szewc, M., Wojtunik, I., Awrejcewicz, J., & Olejnik, P. (2014). Mathematical model, computer aided design and programming of a multifunctional flying object. *Aviation*, *18*(1), 28-39.

[2] Chamola, V., Kotesh, P., Agarwal, A., Gupta, N., & Guizani, M. (2021). A comprehensive review of unmanned aerial vehicle attacks and neutralization techniques. *Ad hoc networks*, *111*, 102324.

[3] Piskorski, S., Brulez, N., & Eline, P. Ha eyer FD, AR. *Drone Developer Guide SDK*, *2*.

[4] Elruby, A. Y., El-Khatib, M. M., El-Amary, N. H., & Hashad, A. I. (2012, May). Dynamic modeling and control of quadrotor vehicle. In *The International Conference on Applied Mechanics and Mechanical Engineering* (Vol. 15, No. 15th International Conference on Applied Mechanics and Mechanical Engineering., pp. 1-10). Military Technical College.

[5] Michael, N., Mellinger, D., Lindsey, Q., & Kumar, V. (2010). The grasp multiple micro-uav testbed. *IEEE Robotics & Automation Magazine*, *17*(3), 56-65.

[6] Müller, M., Lupashin, S., & D'Andrea, R. (2011, September). Quadrocopter ball juggling. In *2011 IEEE/RSJ international conference on Intelligent Robots and Systems* (pp. 5113-5120). IEEE.

[7] Hernandez, A., Copot, C., De Keyser, R., Vlas, T., & Nascu, I. (2013, October). Identification and path following control of an AR. Drone quadrotor. In *2013 17th international conference on system theory, control and computing (ICSTCC)* (pp. 583-588). IEEE.

[8] Koszewnik, A. (2014). The parrot UAV controlled by PID controllers. *acta mechanica et automatica*, *8*(2).

[9] Prayitno, A., Indrawati, V., & Utomo, G. (2014). Trajectory tracking of AR. Drone quadrotor using fuzzy logic controller. *TELKOMNIKA Telecommunication Computing Electronics and Control*, *12*(4), 819-828.

[10] Cansalar, C. A., Maviş, E., & Kasnakoğlu, C. (2015, March). Simulation time analysis of MATLAB/Simulink and LabVIEW for control applications. In *2015 IEEE International Conference on Industrial Technology (ICIT)* (pp. 470-473). IEEE.

[11] Guo, Y., Zhang, R., & Li, H. (2016, July). Robust trajectory control for quadrotors with disturbance observer. In *2016 35th Chinese Control Conference (CCC)* (pp. 10788-10794). IEEE.

[12] Khusheef, A. S. (2016). An Efficient Approach for Modeling and Control of a Quadrotor. *Wasit Journal of Engineering Sciences*, *4*(2), 1-16.

[13] Holonec, R., Copindean, R., Dragan, F., & Rápolti, L. (2016). Self-guided AR Drone using LabVIEW. *Acta Electrotehnica*, *57*(5).

[14] Mahmoud, G., Mohammed, O., Brahim, F., & M'hamed, M. (2012). Kinematic modelling and simulation of a 2-R robot using solidworks and verification by MATLAB. *International Journal of Advanced Robotic Systems*, *9*(245), 2012.

[15] Gordon, R., Ruff, R. A., & Kumar, P. (2013). Simulating Quadcopter Dynamics using Imported CAD Data. In *AIAA Modeling and Simulation Technologies (MST) Conference* (p. 4735).

[16] Elsamanty, M., Khalifa, A., Fanni, M., Ramadan, A., & Abo-Ismail, A. (2013, July). Methodology for identifying quadrotor parameters, attitude estimation and control. In *2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronics* (pp. 1343-1348). IEEE.

[17] Cekus, D., Posiadała, B., & Warys, P. (2014). Integration of modeling in SolidWorks and Matlab/Simulink environments. *Archive of Mechanical Engineering*, *61*(1).

[18] Shaqura, M., & Shamma, J. S. (2017, January). An Automated Quadcopter CAD based Design and Modeling Platform using Solidworks API and Smart Dynamic Assembly. In *ICINCO (2)* (pp. 122-131).

[19] De Simone, M. C., Russo, S., Rivera, Z. B., & Guida, D. (2017, May). Multibody model of a UAV in presence of wind fields. In *2017 International Conference on Control, Artificial Intelligence, Robotics & Optimization (ICCAIRO)* (pp. 83-88). IEEE.

[20] Zatopek, J., MACHADO, J., & SOUSA, J. (2018). Dynamic simulation of the CAD model in SimMechanics with multiple uses. *Turkish Journal of Electrical Engineering & Computer Sciences*, *26*(3), 1278-1290.

[21] Grau, A., Bolea, Y., & Sanfeliu, A. (2019). CAD-based Approach for Identification of UAVs. In *MATEC Web of Conferences* (Vol. 291, p. 01004). EDP Sciences.

[22] Zabidin, Y. A. A., Pairan, M. F., & Shamsudin, S. S. (2020). Dynamic Modelling and Control for Quadcopter UAV with LabVIEW and X-Plane Flight Simulator. *Journal of Complex Flow*, *2*(2), 19-26.

[23] Santana, L. V., Brandão, A. S., & Sarcinelli-Filho, M. (2020). An open-source testbed for outdoor navigation with the AR. Drone quadrotor. *IEEE Systems Journal*, *15*(3), 3597-3608.

[24] Velasco, O., Valente, J., Alhama Blanco, P. J., & Abderrahim, M. (2020). An open simulation strategy for rapid control design in aerial and maritime drone teams: A comprehensive tutorial. *Drones*, *4*(3), 37.

[25] Idrissi, M., Salami, M., & Annaz, F. (2021). Modelling, simulation and control of a novel structure varying quadrotor. *Aerospace Science and Technology*, *119*, 107093.

[26] Jatsun, S., Lushnikov, B., Emelyanova, O., & Leon, A. S. M. (2021). Synthesis of simmechanics model of quadcopter using solidworks CAD translator function. In *Proceedings of 15th International Conference on Electromechanics and Robotics" Zavalishin's Readings"* (pp. 125-137). Springer, Singapore.

[27] Madani, T., & Benallegue, A. (2006, October). Backstepping control for a quadrotor helicopter. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 3255-3260). IEEE.

[28] Lee, D., Jin Kim, H., & Sastry, S. (2009). Feedback linearization vs. adaptive sliding mode control for a quadrotor helicopter. *International Journal of control, Automation and systems*, *7*(3), 419-428.

[29] Das, A., Subbarao, K., & Lewis, F. (2009). Dynamic inversion with zero-dynamics stabilisation for quadrotor control. *IET control theory & applications*, *3*(3), 303-314.

[30] Li, T., Zhang, Y., & Gordon, B. (2012, August). Investigation, flight testing, and comparison of three nonlinear control techniques with application to a quadrotor unmanned aerial vehicle. In *AIAA*

Guidance, Navigation, and Control Conference (p. 4916).

[31] Dydek, Z. T., Annaswamy, A. M., & Lavretsky, E. (2012). Adaptive control of quadrotor UAVs: A design trade study with flight evaluations. *IEEE Transactions on control systems technology*, *21*(4), 1400-1406.

[32] Khalifa, A., Fanni, M., Ramadan, A., & Abo-Ismail, A. (2013, October). Adaptive intelligent controller design for a new quadrotor manipulation system. In *2013 IEEE International Conference on Systems, Man, and Cybernetics* (pp. 1666-1671). IEEE.

[33] Cömert, C., & Kasnakoğlu, C. (2017). Comparing and developing PID and sliding mode controllers for quadrotor. *International Journal of Mechanical Engineering and Robotics Research*, *6*(3), 194-199.

[34] Fethalla, N., Saad, M., Michalska, H., & Ghommam, J. (2018). Robust observer-based dynamic sliding mode controller for a quadrotor UAV. *IEEE access*, *6*, 45846-45859.

[35] Miranda-Colorado, R., & Aguilar, L. T. (2020). Robust PID control of quadrotors with power reduction analysis. *ISA transactions*, *98*, 47-62.

[36] Martins, L., Cardeira, C., & Oliveira, P. (2021). Feedback linearization with zero dynamics stabilization for quadrotor control. *Journal of Intelligent & Robotic Systems*, *101*(1), 1-17.

[37] Leal, I. S., Abeykoon, C., & Perera, Y. S. (2021). Design, Simulation, Analysis and Optimization of PID and Fuzzy Based Control Systems for a Quadcopter. *Electronics*, *10*(18), 2218.

[38] Fethalla, N., Saad, M., Michalska, H., & Ghommam, J. (2018). Robust observer-based dynamic sliding mode controller for a quadrotor UAV. *IEEE access*, *6*, 45846-45859.

[39] VÉVODA, A. (2009). Porovnání moderních 3D CAD programů.

[40] The MathWorks Inc., Users Guide: SimMechanics, For Use with Simulink®, Reprint for Version 2.2, December 2005.

[41] Schlotter, M. (2003). Multibody system simulation with simmechanics. *University of Canterbury*, 1-23.

[42] The MathWorks Inc. SimMechanics, Users Guide March,2012.

[43] Ahmed, F., Kumar, P., & Patil, P. P. (2016). Modeling and simulation of a quadcopter UAV. *Nonlinear Studies*, *23*(4).

[44] Emran, B. J. (2014) .Nonlinear Adaptive control of Quadrotor. Master of Science in Mechatronics Engineering

[45] Hussein, M. T., & Nemah, M. N. (2015, October). Modeling and control of quadrotor systems. In *2015 3rd RSI International Conference on Robotics and Mechatronics (ICROM)* (pp. 725-730). IEEE.

[46] Zuo, Z. (2010). Trajectory tracking control design with command-filtered compensation for a quadrotor. *IET control theory & applications*, *4*(11), 2343-2355.

[47] Wang, L., He, C., & Zhu, P. (2014). Adaptive sliding mode control for quadrotor aerial robot with I type configuration. *International Journal of Automation and Control Engineering (IJACE)*, *3*, 20-26.

[48]“SimulinkDocumentation”.https://www.mathworks.com/help/simulink/ (toegang verkry Apr 06, 2022).

[49] Mogenson, M. (2012). *The AR Drone LabVIEW Toolkit: A Software Framework for the Control of Low-Cost Quadrotor Aerial Robots* (Doctoral dissertation, Tufts University).

[50] “AR.Drone 2.0 Support from Embedded Coder - File Exchange - MATLABCentral”.https://www.mathworks.com/matlabcentral/fileexchange/48558-ar-drone-2-0-support-from-embedded-coder (toegang verkry Aug 24, 2021).

[51] “AR Drone Simulink Development-Kit V1.1 - File Exchange - MATLABCentral”.https://www.mathworks.com/matlabcentral/fileexchange/43719-ar-drone-simulink-development-kit-v1-1 (toegang verkry Aug 23, 2021).

[52] Portal, J. V. (2011). A Java autopilot for parrot AR drone designed with DiaSpec.

[53] Emter, T., & Stein, A. (2012, May). Simultaneous Localization and Mapping with the Kinect sensor. In *ROBOTIK 2012; 7th German Conference on Robotics* (pp. 1-6). VDE.

[54] Hannah, K. J., Ball, M. J., & Edwards, M. J. (2006).

Telecommunications and Informatics. In *Introduction to Nursing Informatics* (pp. 41-54). Springer, New York, NY.

APPENDIXES

Appendix A: AR. Drone

Drone includes an external casing that is built of a sturdy and flexible expanded polypropylene material to avoid injuries. For shockproofing, carbon fiber is covered with a specific foam substance placed on top of the structure. AR. Drone features a front-facing camera and a downward-facing camera for live video broadcasting. A significant advantage is that it requires no additional power from the drone's battery. There are no additional boards connected to the drone that consume energy, and the drone is not carrying any more weight [49].

A.1. Hardware

AR. Drone can perform automated actions such as departing, landing, and remaining in the same position during flight (a movement known as "hovering"). It can be used outdoors, indoors, with or without a protective frame (here, the "inner body"). Uses the rechargeable lithium battery included with the parrot. The drone provides data on its current condition, location, and direction in real-time and sends a video stream to the selected camera. It features an embedded Linux operating system [54]. The PC and the drone communicate over Wi-Fi. Due to the fact that the signal quality degrades significantly with distance, AR Drone is unable to travel more than 100 meters without signal amplification. However, by adding a compatible GPS module to the quadcopter's USB connection, the quadcopter can be equipped with GPS communication. This module enables AR. Drone to travel greater distances without being tied to the user by programming it to travel to specific coordinates in order to perform a specific activity (eg collecting images, weather data, etc.) [13].

A.2. Sensors

Drones are equipped with a variety of sensors. Automatic stabilization is accomplished by the use of these sensors.

1- Inertial measurement unit

The AR. Drone is equipped with a 6-DOF inertial measurement unit. It gives pitch, roll, and yaw measurements to the onboard software. These measurements are utilized to automatically stabilize pitch, roll, and yaw, as well as assist with tilting control. The IMU is a comprehensive inertial system consisting of a three-axis gyroscope, three-axis acceleration sensors, and a three-axis magnetometer.

2- Ultrasound altimeter

Altitude measurements are taken using an ultrasonic sensor to aid in vertical speed control and automatic altitude stabilization. The ultrasonic sensor is mounted at the bottom of the AR. Drone and pointed downward to determine height above ground. The sensor transmits a packet of ultrasonic sound waves toward the floor, which reflects the sound back to the sensor and then the system monitors how long it takes for the echo to return to the sensor and uses the speed of sound to calculate the distance to the target. AR. Drone's ultrasonic sensor has an effective range of roughly 20 cm to 6 meters, and the pressure sensor may also be used to monitor height. but unlike the ultrasound sensor, which has a limited range, the pressure sensor can measure altitude at any height. The ground speed is measured by the camera pointed at the ground, which allows for autonomous hovering and trimming [55].

III. Cameras

There are two cameras onboard AR. Drone a horizontal front camera and a vertical down-facing camera. AR. Drone's bottom (vertical) camera is used to measure ground speed, which is then used for autonomous hovering and trimming during flight, and AR. Drone can transmit an encoded stream of images captured by these cameras [54].The frontal camera has a CMOS sensor and a lens with a 90-degree field of view. Using a proprietary codec and streaming protocol, the AR. Drone automatically encodes and broadcasts incoming photos to the host device. Both cameras on AR. Drone 2.0 have a resolution of 360p (640x360) or 720p (1280x720), depending on the model (with upscaling from the camera facing the bottom). The frame rate of the video stream can be changed between 15 and 30 frames per second [3].

A.3. Network connection

AR. Drone 2.0 can be controlled from any Wi-Fi-enabled device. When the drone launches, it creates a Wi-Fi network with an Extended Service Set Identifier (ESSID) called and self-assign an odd Internet Protocol (IP) address called ardrone2_ 192.168.1.1 When the device connects to the drone's Wi-Fi network and requests an IP address from the drone Dynamic Host Configuration Protocol (DHCP) server, The DHCP server assigns the device IP address which is the IP address of AR. Drone [56].

الخلاصة

ركزت هذه الرسالة على دراسة طائرة هليكوبتر رباعية المحركات. تم تنفيذ نمذجة النظام الديناميكي وتقييم خوارزمية التحكم. تمت صياغة النظام الديناميكي باستخدام نظرية نيوتن أويلر. تم إجراء الاختبارات على نموذج محاكاة حيث يمكن تقييم الأداء بسهولة باستخدام نهج رياضي. أيضًا، سيركز العمل على مزايا العديد من البرامج الهندسية لتقديم دراسة حالة للأنظمة الرباعية من التصميم الأساسي إلى التحكم في الحركة. يعد برنامج CAD أداة قوية جدًا لتصميم نموذج للأنظمة الديناميكية، مما يلغي اشتقاق المعادلات الرياضية. يمكن لبرنامج MATLAB استقبال نموذج CAD وتحويله إلى نموذج فيزيائي محاكى. تم استخدام نموذج CAD للنظام الرباعي في هذه الدراسة لإظهار قدرة تنفيذ النموذج ثلاثي الأبعاد في النمذجة والتحكم. كما تمت مقارنة خوارزميات التحكم في وضع الانزلاق مع التحكم التكيفي والتحكم في الوضع الانزلاقي مع التحكم في PID. توضح هذه الدراسة تقنية التعرف والتحكم بطائرة Parrot AR Drone. أيضًا تم إنشاء حركة التحكم في موضع المحرك الرباعي (AR. Drone) باستخدام برنامج LabVIEW. هنا، سيتم التحكم في المحرك الرباعي بطريقتين باستخدام عصا التحكم واستخدام لوحة المفاتيح. تم تشكيل AR Drone في مسار التحويم والمسار المستقيم. أظهرت نتائج المحاكاة في النموذج الديناميكي تتبع المسار الصحيح ونجاح وحدة التحكم المقترحة باستخدام MATLAB / Simulink أيضًا ، تُظهر أن نموذج CAD ثلاثي الأبعاد يعمل بشكل جيد مع وحدات التحكم ويحافظ على الموضع المطلوب والموقف على طول المسار المحدد مسبقًا. بالإضافة إلى ذلك ، أثبت أن جهاز التحكم PID أكثر كفاءة من جهاز التحكم التكيفي. ثبت أن معدل الخطأ عند تنفيذ مسار الحلزون للنموذج الديناميكي أفضل من نموذج CAD بفارق 12٪ للموضع. بينما أظهرت النتائج المعملية كفاءة استخدام برنامج LabView في التحكم في هذا النوع من المحركات الرباعية. لقد ثبت أن التحكم في الطائرة بدون طيار باستخدام لوحة المفاتيح وذراع التحكم يعطي نتائج مماثلة عند مقارنة نسبة الخطأ. واستنتج عند مقارنة نتائج المحاكاة والتجربة أن تطبيق المسار أفضل في المحاكاة. على الرغم من وجود اختلاف طفيف في معدل الخطأ ، إلا أن هذا يشير إلى نجاح التجارب.

جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعة بابل / كلية الهندسة
قسم الهندسة الميكانيكية

التحكم التكيفي لطائرة رباعية المراوح باستخدام برنامجي اللابفيو والماتلاب

رسالة
مقدمة إلى جامعة بابل / كلية الهندسة وهي جزء من متطلبات نيل درجة الماجستير في الهندسة/ الهندسة الميكانيكية/ميكانيك تطبيقي

أعدت من قبل
فاطمة عادل رحيمة علي

بأشراف

ا.م.د.مصطفى تركي حسين

1442 هـ 2022 م