Spring 2018

Design and Implementation of an Artificial Neural Network Controller for Quadrotor Flight in Confined Environment

Ahmed Mekky
Old Dominion University

DESIGN AND IMPLEMENTATION OF AN ARTIFICIAL NEURAL NETWORK CONTROLLER FOR QUADROTOR FLIGHT IN CONFINED ENVIRONMENT

by

Ahmed Elhussein Eltayeb Mekky
B.Sc. August 2007, University of Khartoum, Sudan
M.Sc. August 2012, Old Dominion University

A Dissertation Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

AEROSPACE ENGINEERING

OLD DOMINION UNIVERSITY
May 2018

ABSTRACT

DESIGN AND IMPLEMENTATION OF AN ARTIFICIAL NEURAL NETWORK CONTROLLER FOR QUADROTOR FLIGHT IN CONFINED ENVIRONMENT

Ahmed Elhussein Eltayeb Mekky
Old Dominion University, 2018
Director: Dr. Thomas E. Alberts

Quadrotors offer practical solutions for many applications, such as emergency rescue, surveillance, military operations, videography and many more. For this reason, they have recently attracted the attention of research and industry. Even though they have been intensively studied, quadrotors still suffer from some challenges that limit their use, such as trajectory measurement, attitude estimation, obstacle avoidance, safety precautions, and land cybersecurity. One major problem is flying in a confined environment, such as closed buildings and tunnels, where the aerodynamics around the quadrotor are affected by close proximity objects, which result in tracking performance deterioration, and sometimes instability. To address this problem, researchers followed three different approaches; the Modeling approach, which focuses on the development of a precise dynamical model that accounts for the different aerodynamic effects, the Sensor Integration approach, which focuses on the addition of multiple sensors to the quadrotor and applying algorithms to stabilize the quadrotor based on their measurements, and the Controller Design approach, which focuses on the development of an adaptive and robust controller. In this research, a learning controller is proposed as a solution for the issue of quadrotor trajectory control in confined environments. This controller utilizes Artificial Neural Networks to adjust for the unknown aerodynamics on-line. A systematic approach for controller design is developed, so that, the approach could be followed for the development of controllers for other nonlinear systems of similar form. One goal for this research is to develop a global controller that could be applied to any quadrotor with minimal adjustment. A novel Artificial Neural Network structure is presented that increases learning efficiency and speed. In addition, a new learning algorithm is developed for the Artificial Neural Network, when utilized with the developed controller.

Simulation results for the designed controller when applied to the Qball-X4 quadrotor are presented that show the effectiveness of the proposed Artificial Neural Network structure and the developed learning algorithm in the presence of variety of different unknown aerodynamics. These results are confirmed with real time experimentation, as the developed controller was successfully applied to Quanser's Qball-X4 quadrotor for the flight control in confined environment. The practical challenges associated with the application of such a controller for quadrotor flight in confined environment are analyzed and adequately resolved to achieve an acceptable tracking performance.

DEDICATION

This dissertation is dedicated with love and appreciation, to my parents, my brothers, my sister, my wife, my daughters, and my son.

ACKNOWLEDGEMENTS

I would like to thank all those who have directly or indirectly helped me during the course of my research work. First of all, I would like to thank the almighty God (Allah), glory be to him, for giving me the strength throughout my PhD study, without his will, I would not have been able to achieve this degree. I wish to express my deep sense of gratitude and sincere thanks to my advisor Dr. Thomas Alberts for his guidance and continuous support during the course of my research. The knowledge and experience I have gained through his expertise is invaluable. I wish to express my heartfelt thanks to Dr. Oscar González for his advice, help and support, and for providing the Qball quadrotor, and the Parrot SLAM dunk used for the experimental implementation in this research. My thanks also go to Dr. Brett Newman, Dr. Miltos Kotinis, and Dr. Drew Landman for being part of my committee and reviewing my research harvest. I would like to thank all my professors who have prepared me for my doctorate study journey, without their teachings it would be impossible to reach the end. I would like to specifically thank Dr. Steven Gray, Dr. Gene Hou, Dr. Colin Britcher and Dr. Robert Ash. I would like to express my sincere thanks to the staff of the Department of Mechanical and Aerospace Engineering for providing a rich environment for knowledge exchange, I would like to especially thank Diane Mitchell, and Jayne Massey. I would like to acknowledge the financial support provided by the Department of Mechanical and Aerospace, the Old Dominion University Research Foundation, the Dr. Robert A. and Ronnie Slocum Magoon Scholarship for Aerospace Engineering, Prabhav Mniyar Endowed Scholarship, the Oktay Baysal Endowed

Graduate Scholarship in Computational Engineering for Aerospace. I sincerely appreciate their support.

I would like to sincerely thank Dr. Charles Broadfield (Mukhtar) for his endless support, for being a friend and family, and for proof reading this manuscript. I honestly appreciate his help.

I would like to thank Dr. Nabeel, Dr. Hiba, Amin Mohammed, Ibrahim Ahmed and their families for being my family in this country and for their continual support and help. I would like to also thank my friends who had supported me and for their excessive care. I would like to especially thank Samantha Wright for her help during the course of my research. I would like to also thank the Muslim Community at ODU for making me feel at home.

Finally, I would like to acknowledge Elhussein E. Mekky, my father, and Duria Hassan, my mother, for their endless love, encouragement, support and motivation, my brothers, Mohammed, Eltayeb, and Abdulhafiz, and my sister, Maysaa, and their families for their unconditional love, emotional support, and endless encouragement. I would like to sincerely thank my wife, Ruaa, my daughters, Maryam (may her soul rest in peace) and Nour, and my son, Omer, for their love, support, and more importantly, for their sacrifices and patience throughout tough times we have been through together. Without them I would have not been able to pass through hard times and achieve my graduate study. This dissertation is dedicated for them.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Figure Page

Figure Page

Figure Page

Figure Page

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Quadrotors have recently gained major attention in commercial and research communities. The emerging technology is found very useful in many fields; such as science, defense, humanity and lifestyle improvement [1]. Quadrotors also represent an affordable nonlinear testbed for the experimentation of new control synthesis and autonomy approaches. Therefore, many educational and research establishments have built quadrotor dedicated labs. However, many challenges are still surrounding quadrotors that limit their use in many applications. This research targets two objectives: 1) expanded maneuverability, and 2) flight in confined environment.

Researchers have approached the two objectives from three different perspectives; 1) development of a precise quadrotor model that captures all possible aerodynamic effects when flying in different environments [2] [3] [4] [5] [6], 2) additional sensors to provide measurements that could help implement complex control algorithms and maintain performance [7] [8] [9] [10] [11], and 3) design of an adaptive and robust controller that would efficiently work for all possible climates [12] [13] [14] [15] [16]. A great deal of research has been performed in all areas. Nevertheless, researchers are still striving to achieve better performance. When flying in confined environments, the behavior of quadrotors is greatly influenced by the aerodynamic uncertainties associated with the interaction of the airflow from propellers and the features of the environment in close proximity with the quadrotor. There is no available model for quadrotors, thus far, that encompasses the influence of confined environments, as the aerodynamics associated with confined environments are not yet fully understood, and they differ from one setup to another. Therefore, it is not feasible to design fixed controllers based on the available models without accounting for uncertainties that could be of a considerable order of magnitude, and of a high level of complexity.

In this research, a learning controller was developed to address the problem of quadrotor control in confined environments. There are numerous mechanisms for learning and intelligence that could produce an intelligent (learning) machine. Most of those methods are related to computer science applications and are taught in the field of Artificial Intelligence (AI) e.g. hand writing recognition, bioinformatics, computer vison, and others. Nevertheless, Artificial Neural Networks (ANN) were utilized to create a learning controller that learns to better control the system while operating in flight. The proposed controller was developed using sliding mode control methodology and following a backstepping control approach. Two ANNs were used to compensate for aerodynamic force and moment uncertainties. A novel ANN structure was proposed that is believed to increase the speed and the accuracy of learning. The learning algorithm was developed using dynamic optimization for an objective function developed using the sliding mode technique, putting in mind both learning performance, and ANN weights boundedness. The stability of the closed loop system was analyzed using Lyapunov stability theory, and accordingly the limits for design parameters were defined for the guaranteed stability. The effectiveness of the proposed ANN structure, the developed learning laws, and the developed nonlinear controller is presented using nonlinear numerical simulations. Furthermore, experimental results are provided for the developed controller when used for the trajectory control of the Qball quadrotor in confined environments. The rest of this dissertation is organized as follows

In Chapter 1, a brief introduction about the problem of quadrotor flight control in confined environments, and the proposed solution, is provided. The literature pertaining to this research is briefly discussed, focusing on learning control systems, the use of ANN for aircraft control, and the current state of research in the field of quadrotor trajectory control. The research statement, and the approach of this research are provided, and the objectives of this study are addressed.

In Chapter 2, the theoretical background required for the development of the proposed controller is provided, focusing on the theory of ANNs, sliding mode control methodology, backstepping control approach, Lyapunov stability theorem, and the mathematical model for quadrotor dynamics.

In Chapter 3, a step-by-step derivation of the ANN controller when applied for quadrotor trajectory control is provided. The sliding mode methodology is used to design the nonlinear controller following backstepping control approach. Two ANNs are used to compensate for unknown aerodynamic forces and moments associated with environment interaction. A novel ANN structure is introduced, and the corresponding learning law is developed. The stability of the closed loop system is discussed using Lyapunov stability theorem.

Chapter 4 focuses on the practical implementation of the developed controller. First, numerical simulation results are presented to validate the effectiveness of the proposed ANN structure and the developed learning algorithm in approximating a wide range of nonlinear uncertainties. And finally, real time experimental results are provided for the Qball quadrotor trajectory control using the proposed controller, and with a different variety of environmental setups, to test for controller robustness and effectiveness.

In Chapter 5, conclusions that can be drawn from this research work are presented. Furthermore, recommendations for future work are provided.

1.2 LITERATURE REVIEW

1.2.1 Learning Control of Nonlinear Systems

Intelligent control describes the discipline in which control methods are developed in an attempt to emulate important characteristics of human intelligence [17] [18] [19] [20] [21] [22] [23] [24]. These characteristics include adaptation, learning, planning under large uncertainty, and coping with the availability of large amounts of data [17]. The number of ideas and concepts of learning is very large, but not one of them purports to encompass the whole problem. Areas of adaptive control systems, pattern recognition, production of systems, learning controllers, and neural networks state different goals of learning systems [25].

1.2.1.1 Intelligent vs Adaptive Control

What is the relationship/difference between adaptive control and learning (intelligent) control [26]? Learning is achieved when an adaptive control algorithm is employed to adapt the controller parameters so that, for example, stability is maintained. In this case, the system learns, and the knowledge acquired is the new value for the parameters. However, if later the same changes occur again and the system is described by exactly the same parameters identified earlier, the adaptive control algorithm still needs to recalculate the controller and perhaps the plant parameters since nothing was kept in memory [26]. So, in that sense, the system has not learned.

There are many areas in control where learning can be useful [26]: (1) learning about the plant; (2) learning about the environment; (3) learning about the controller; and (4) learning new design goals and constraints.

1.2.1.2 Intelligence in Control of Nonlinear Systems

Many learning control systems were developed that utilize, in general, two basic AI techniques: (1) Artificial Neural Networks; and (2) Fuzzy Logic, and an offspring of both in the so-called Hybrid Neuro-Fuzzy control.

1.2.1.2.1 Artificial Neural Networks

Artificial Neural Networks have found a significant interest in the field of the control of nonlinear systems since the early 1990's [17] [20] [23] [24] [26] [27] [28] [29] [30] [31] [32] [33] [34]. In [35], it is clarified that *Neural Networks for control systems* refers to the study of ANNs that go beyond monitoring or classifying their input signals to actually influencing them. Unlike most ANNs, these are explicitly designed to learn from a closed loop interaction with the system. Closed loop control involves a very different set of requirements for learning methods than those usually considered in conventional ANN research. Some of those requirements are, for example, in control, it is important to learn online, incrementally, and without an explicit supervisor specifying desired behavior [35]. This reference, [35], is a good starting point for researching in the field of ANNs for control systems, as it was published as a result of a workshop entitled "The Application of Neural Networks to Robotics and Control", held at the University of New Hampshire, Durham, USA, in October, 1988. The philosophy of using ANN in control systems was tackled from different points of view, and some practical examples for the implementation of ANNs in control systems are presented. One of the early books to collect the use of ANN in control systems is reference [36], which was a report resulted from the IEE Workshop on Neural Networks for Control Systems: Principles and Applications, at the University of Reading, Reading, UK, in April 1992. This book provides a broad material about the topic of ANN and their implementation in control systems, focusing on the practical aspect of the matter. *Neural Networks for Modeling and Control of Dynamic Systems* [37], 2000, is another reference that presented the topic of ANNs in control systems, with a focus on two-layer perceptron neural networks with hyperbolic tangent hidden units and linear output units; which is regarded as the most commonly used type of ANN in control systems [37]. However, it is pointed out that despite the fact that concepts and methods were developed for this specific type of ANNs, they are still applicable for different types of ANNs with minimal adjustments. This book is aimed at providing a practical insight and the required theory for the implementation of such ANN structure in the context of the control of dynamical systems. In *Intelligent Observer and Control Design for Nonlinear System* [33] the problem of using ANNs for the control of nonlinear systems with a focus on the indirect control structure is presented. In which, the main focus is to identify system's dynamics, and the observed output states are used to control the actual plant. ANNs and Fuzzy logic techniques are blended with control methodologies to achieve good performance with guaranteed stability. Extensive experimental results were presented for the developed control strategies when used in motion control applications.

It has been shown and proven in the literature that a Neural Network with one hidden layer can map effectively any nonlinear function [21] [23] [38] [39] [29] [28] [40], therefore, it can be used to identify complex uncertainties in the dynamical system, and to model the impact of time on mechanical parts within the system, e.g. the change of backlash, friction, resistance and weight with time. This criterion is very effective for the identification of nonlinear systems offline, and in some applications, online. However, two aspects of neural network are challenging: the network size and structure which affect learning time, memory, and the required performance (precision) [21] [23] [29] [41] [33]. There is a tradeoff between performance and computational effort; an ANN with a bigger size would produce a better approximation for the unknown function while requiring a greater computational effort and bigger memory. Therefore, the choice of a suitable ANN size and structure is not obvious. Hsu et al [24] has attempted to solve that problem by developing an ANN controller that uses the Kohonen [42] self-organizing neural network, that updates its size according to performance to reduce learning time when possible. The drawback of this method is that ANN size update requires extra processing i.e. more computational effort.

Static and Dynamic Neural Networks and their theory are intensively studied in the literature [43] [22] [28] [44]. And a thorough body of material is available about their application in the fields of computer science, design optimization and control systems etc. [18] [20].

ANN controllers are generally developed following two methodologies from control systems theory: 1) Optimal control, critique/actor method [45] [46] [47] [48] [49] [50] [51] [52], and, 2) Nonlinear control methods, such as, Dynamics Inversion, Sliding mode, Feedback Linearization, Backstepping …etc. [53] [54] [55] [21] [31] [33].

1.2.1.2.2 Fuzzy Logic

Fuzzy logic is a powerful tool for modeling human thinking and cognition [28]. The concept was first introduced by Lotfi A. Zadeh in 1965 [56]. It is aimed at a formalization of the remarkable human capability to perform a wide variety of physical and mental tasks without any measurements and any computations [57]. It is referred to as the first computational intelligence technique to be used in control systems [28]. Fuzzy logic is a form of multivalued logic, it deals with reasoning that is approximate rather than fixed and exact. The main advantages of fuzzy logic in control systems are: 1) No accurate mathematical model of the system is required, and 2) linguistic fuzzy (loose) control rules or linguistic fuzzy description of the system can be incorporated to control the system [58]. Fuzzy control has been successfully employed in many commercial and industrial systems [29] [40] [57] [58].

Despite the fact that existing fuzzy controllers are capable of incorporating linguistic information, they are heuristic in nature in the sense that there are no general design methods that guarantee the basic requirement of control e.g. stability, robustness, etc. [58].

1.2.1.3 Direct and Indirect Controller

Indirect controller refers to a controller that works in parallel with the system to continually identify the nonlinear plant and the output is used as observed states to control the actual plant [33]. So, in this sense, the controller learns indirectly to control the plant by learning about the plant. The learning process here could be performed off-line and applied to the plant for further adjustments online. On the other hand, the direct controller refers to the controller that learns to optimally control the plant online i.e. it finds the optimum input function u^* that produces the "best" output using some criterion to judge [24]. Therefore, in this structure the controller learns to control the plant directly by evaluating the chosen performance index. Figure 1.1 shows the block diagram of the indirect controller (a), and the direct controller (b) and (c). Notice that the word “optimal” here is loosely used to refer to the final goal of learning, or adaptation, which requires a performance measure that is needed to be improved, maximized or minimized, along the learning process.

1.2.2 ANN Control of Aircraft

The application of ANN in the closed loop control of aircrafts has been conceptualized since the early 1990s. Robert F. Stengel has published in his paper “Towards Intelligent Flight Control” in 1993 [59] a thorough study about intelligent guidance, navigation and control of aircrafts. He provided the definition for most of the concepts and terms that are used in the decades afterwards in the field of intelligent control. According to [59], intelligent flight control could help in making the aircraft less dependent on the proper human operator, enhance aircraft capability, improve performance, increase reliability and safety, and lower cost [59]. In his paper, he emphasized the use of ANN as an imperative component of the intelligent control algorithm. The role of ANN in the intelligent control, according to [59], is to provide a rapid, nonlinear, input-output function.

Byoung S. Kim et al [30] has designed a feedback linearization controller for F-18 that uses an off-line trained ANN to invert the flight dynamics and another online trained ANN to compensate for modeling error in the first ANN. The proof of stability for the closed loop system is provided in [30] for the case of using only the offline trained ANN and in the case of using the two. However, simulation results show a poor response when using the two ANN, and an unstable response when using only one ANN. Besides, basis functions are chosen to be a combination of sigma-pi polynomials and radial basis functions, under the poor assumption that aircraft attitude dynamics could be decoupled.

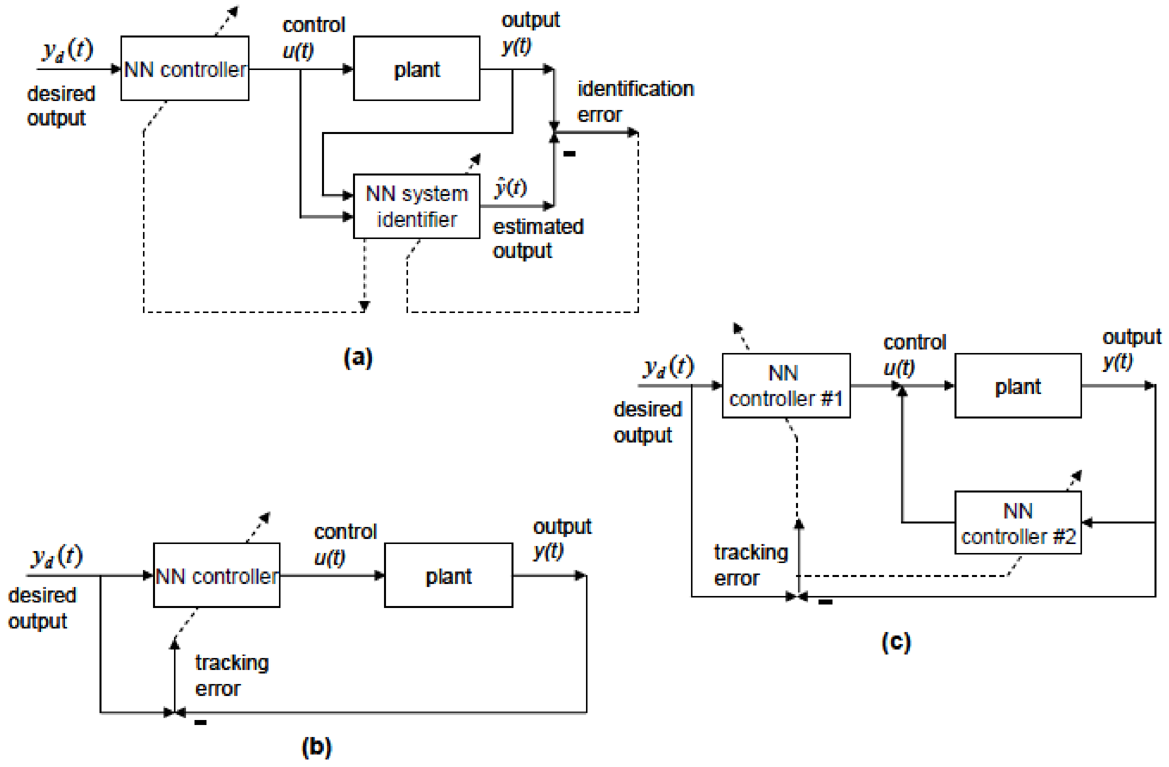


Figure 1.1 Direct and Indirect ANN Controllers

1.2.3 Quadrotor Control

A survey of developed control algorithms for quadrotors is provided in [60], where a summary of the available quadrotor control algorithms is provided with a discussion about advantages and disadvantages of each algorithm. Quadrotors are generally controlled using Proportional-Integral-Derivative (PID) controllers [61]. While PIDs are simple and easy to design and implement, it is agreed upon, in the research community, that its performance is not satisfactory when applied to highly nonlinear systems [60]. Ideas such as variable gain [62] and gain scheduling PID controllers are shown to provide a better performance. However, PID, as a linear controller, limits the options for the system input to a linear combination of system's output errors. Hence, even with the use of a variable gain, it would not be sufficient to achieve the optimal control input as it might be a highly nonlinear function. Nonlinear controllers on the other hand, are generally derived based on the mathematical expression of the system [63] [64]. Thus, their performance depends heavily on the knowledge of the system to be controlled. High performance could be achieved using nonlinear controllers, as nowadays it is possible to produce very precise models for nonlinear systems with the current revolution of technology and advancement of sensing and computation. Nevertheless, the ideal condition in laboratory might not cover all possible environments that the system would experience in real-life. Accordingly, nonlinear controllers usually perform poorly in some operational conditions. Fused with nonlinear control methods, adaptive controllers provide a good solution for uncertainties in the dynamical model. However, most of the available adaptive controllers require extensive computations that would challenge performance, as adaptive controllers would adjust every time a change happens. Learning controllers, on the other hand, exploit the merits of adaptive controllers with the added benefit of memory. With memory, adaptive controllers do not need to re-learn every time a similar situation happens. Thus, resulting in a reduced computational effort and an improved performance.

With respect to quadrotors, PID controllers usually do well in normal flying conditions. However, flying quadrotors in confined environment is still an unsolved puzzle. Some researchers tried to solve the problem by improving the dynamical model, so that, it would capture the aerodynamic effect of being in a confined environment [4] [3] [2] [65] and then design the controller accordingly. Others worked on designing an adaptive nonlinear controller that is capable of controlling quadrotor when subjected to different flying conditions [51] [53] [54] [65] [66] [67] [68].

The dynamical model for quadrotors has been deeply studied in the literature. Adjustments to account for blade aerodynamic effect, wind field, rotor drag and close proximity aerodynamic effects have been made to better control quadrotors in different situations. However, each model is adjusted for a specific situation and a controller is accordingly designed. The general dynamical model for the quadrotor, used in this research, is derived in [69], which is derived following the Lagrange method. This form of quadrotor dynamics is found interesting because it allows for the separation of the navigation loop (outer) and the attitude loop (inner), and has been used in many research papers afterwards.

Many researches have tackled the design of PID controllers for quadrotors. While it has been experimentally shown to be sufficient to control quadrotors in normal hovering and trajectory tracking [61], PID controllers need a considerable effort to be fine-tuned for the specific quadrotor and they perform poorly in critical situations. Therefore, the designed controller will not perform properly if applied to a different quadrotor with different properties. Furthermore, quadrotor flight in nonideal aerodynamics, such as in the presence of wind fields or flight in a confined environment, would result in poor performance as shown in [4] [3].

Pure Backstepping controller design for Quadrotors is presented in [69] [70] [7]. Experimental implementation is shown in [69], and [7]. Backstepping approach and sliding mode method are followed to design the controller for an indoors quadrotor in [71]. This method is very close to that used in this research without the use of the ANNs. Experimental results were shown only for the attitude dynamics to validate the controller performance against a pure backstepping controller.

Many research papers covered the use of adaptive and ANN controllers for quadrotor control. The experimental results of an ANN augmented PID controller is presented in [67], where a PID controller properly designed for a specific quadrotor could be applied directly to another quadrotor with different characteristics without adjustment. A Neuro-adaptive dynamic inversion controller is developed in [68]. An ANN is added to the dynamic inversion controller derived in [54] to compensate for model uncertainty. An optimal ANN based controller for an UAV helicopter is designed in [51], simulation results were shown for the cases of take-off, hovering and landing to validate the control algorithm. Most recently, Joshi et. al. presented in [72] a Neuro-Adaptive Controller for Quadrotors based on dynamic inversion. A Radial Basis ANN is used to compensate for attitude dynamics uncertainties, the ANN update law assumes the availability of a good estimate of these uncertainties, therefore, an Extended Kalman Filter (EKF) is used to estimate unknown disturbances and the estimate is used to, again, train the ANN to estimate the same uncertainties i.e. the ANN weight errors are explicitly calculated at each iteration as the ideal values are available beforehand. Only simulation results in the case of blade flapping and measurement noise are shown, which is a considerably weak form of uncertainty. The most relevant research work is [53], which combines the use of two nonlinear control methodologies; namely Sliding mode [63] and Backstepping [64]. Two ANNs are used to compensate for the aerodynamic effects and un-modeled dynamics, which makes the designed controller general enough to be applied to any quadrotor with minimal adjustment. However, there are some missing pieces in the controller design approach that will be addressed in this research. Furthermore, an experimental validation for the designed controller will be investigated.

1.3 RESEARCH OBJECTIVES AND APPROACH

This research is aimed to design and implement a learning controller that is capable of controlling a quadrotor in the presence of model and aerodynamic uncertainties, with a focus on the aerodynamic uncertainties associated with flying in confined environments. This controller is required to be general enough, so that it would be easily implemented in any quadrotor aircraft. The hypothesis here is that the learning controller would compensate for uncertainties, indirectly, by correcting for the considerable deterioration in quadrotors's performance, and, as a result, would improve quadrotor's performance without the intervention of a human in the loop. Many adaptive controllers were developed for quadrotors. However, none have yet fully-addressed the issue of flight in a confined environment. This work extends on the controller approach presented in [53] where an ANN controller is developed for a quadrotor using a Sliding mode technique and following a Backstepping approach. This approach was found attractive because of two reasons: 1) the designed controller requires minimal computational effort, and 2) the controller is developed primarily based on control systems theory; rather than Artificial Intelligence approaches.

The contributions of this research are: 1) to address some issues with controller stability limits, 2) to use a universal basis function and ANN structure, and 3) to perform an experimental implementation of the controller on a quadrotor flight in confined environment.

CHAPTER 2

THEORETICAL BACKGROUND

This chapter provides a brief elaboration about the concepts and theories used for the controller design. Five topics are covered, Artificial Neural Networks, Sliding Mode Control Methodology, Backstepping Design approach, Lyapunov Stability theorem, and Dynamical model for quadrotors.

2.1 ARTIFICIAL NEURAL NETWORKS

Artificial Neural Networks are developed in an attempt to mimic the functionality of the biological neural system [36]. The brain is a complex, nonlinear and parallel computer. It has the ability to perform tasks such as pattern recognition, perception and muscle control much faster than any computer [73]. In addition, brains have the capability to learn, memorize and generalize. All of these characteristics lead to the vast interest in the development of ANN algorithms.

The biological nervous system processes information through a huge number of interconnected nerve cells, called neurons, Figure (2.1). Signals are propagated in the form of potential difference between the inside and outside of these units. Dendrites bring signals from other neurons into the cell body or the soma, possibly multiplying each incoming signal by a weight coefficient. In the soma, cell capacitance integrates the signals which collect in the axon hillock. Once the composite signal exceeds a cell threshold, a signal, the action potential, is transmitted through the axon. Cell nonlinearities make the composite action potential a nonlinear function of the combination of arriving signals. The axon connects through synapses with the dendrites of subsequent neurons. The synapses operate through the discharge of neurotransmitter chemicals across intercellular gaps, and can be either excitatory (tending to fire the next neuron) or inhibitory (tending to prevent firing of the next neuron).

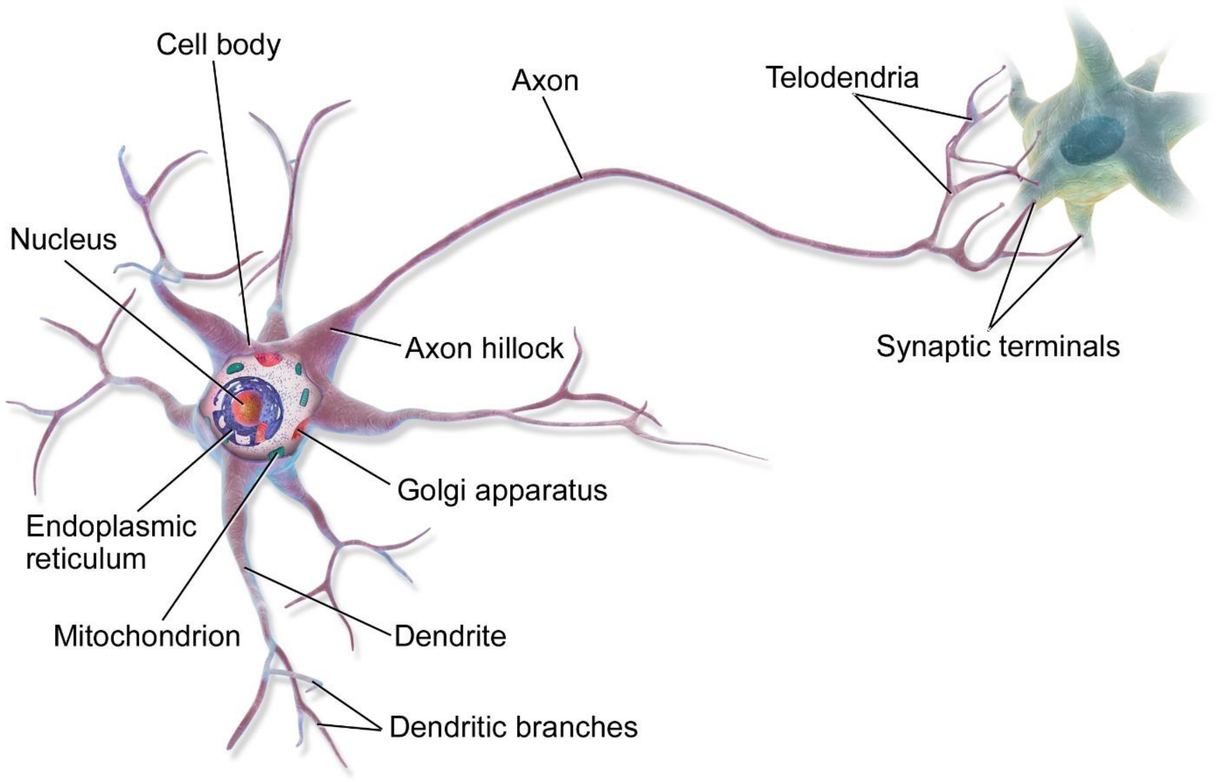


Figure 2.1 Biological Neuron

An artificial neuron (AN), on the other hand, is a model of a biological neuron (BN). Each AN receives signals from the environment, or other ANs, gathers these signals, and when fired, transmits a signal to all connected ANs. Figure 2.2 depicts an artificial neuron. Input signals are inhibited or excited through negative and positive numerical weights associated with each connection to the AN. The firing of an AN and the strength of the exiting signal are controlled by a function, called activation function. The AN collects all incoming signals, and computes a net input signal as a function of respective weights. The net input signal serves as input to the activation function which calculates the output of the AN.

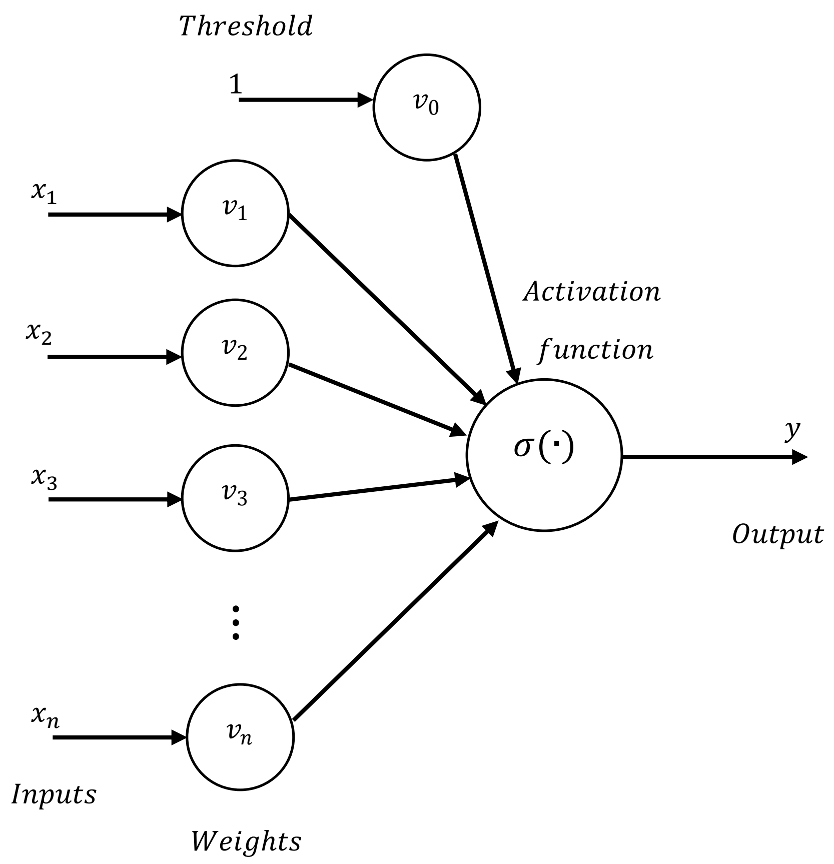


Figure 2.2 Artificial Neuron

An Artificial Neural Network (ANN) is a layered network of ANs. An ANN may consist of an input layer, hidden layers and an output layer. ANs in one layer are connected, fully or partially, to the ANs in the next layer. Feedback connections to previous layers are also possible. A typical ANN structure is shown in Figure 2.4.

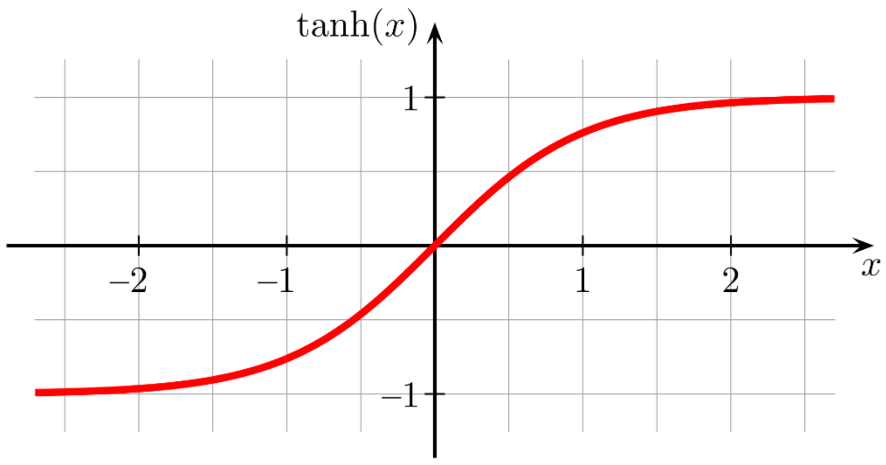


Figure 2.3 Hyperbolic Tangent Activation Function

Several different ANN types have been developed [73], including for example:

- Single-layer ANNs, such as the Hopfield network;
- Multilayer feedforward ANNs, including standard backpropagation, functional link and product units networks;
- Temporal NNs, such as the Elman and Jordan simple recurrent networks as well as time-delayed neural networks;

- Self-organizing NNs, such as the Kohonen self-organizing feature maps and learning vector quantizer;
- Combined supervised and unsupervised ANNs, e.g. some radial basis function networks.

These ANN types have been used for a wide range of applications, including diagnosis of diseases, speech recognition, data mining, music composition, image processing, forecasting, systems control, design optimization, credit approval, classification, pattern recognition, planning game strategies and many others.

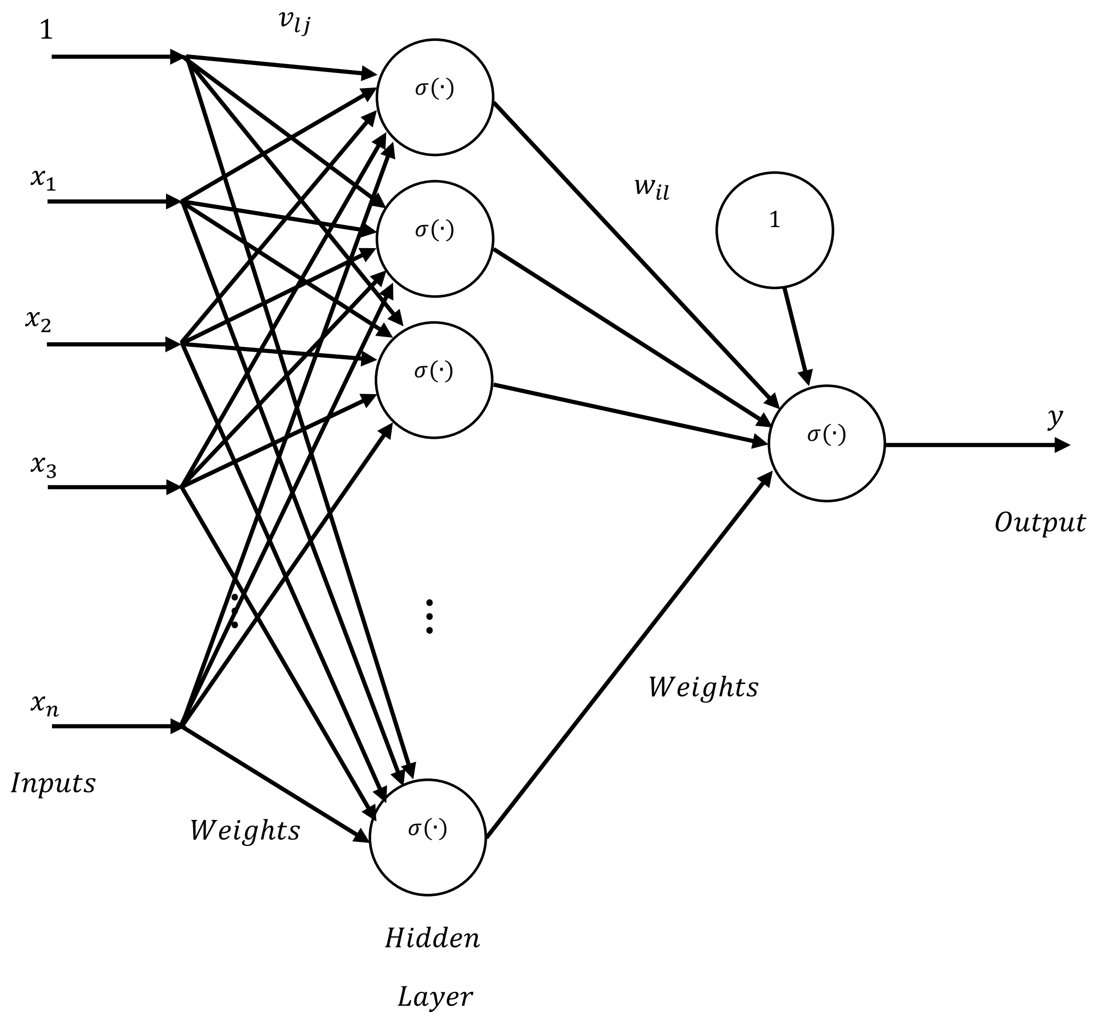


Figure 2.4 Typical Fully Connected ANN

2.1.1 The Mathematical Model for ANNs

In this section, the focus is drawn towards the type of ANNs used in this research, which is Multilayers feedforward ANN, the reader is advised to look at [28] for the other types of ANNs. A novel structure is introduced in this research that modifies the existing structure of feedforward ANNs to achieve fast learning and more accuracy.

A three-layer ANN is depicted in Figure 2.4, where there are two layers of neurons and an input layer. The ANN has n inputs feeding into the second layer having L neurons feeding into the third layer of M neurons. The first layer is known as the Input layer, with n inputs; the second layer is known as the hidden layer, with L the number of hidden-layer neurons; the third layer is known as the output layer, with M the number of outputs. ANN with multiple layers are called multilayer perceptrons.

The output of the three-layer ANN is given by

$$y_i = \sigma\left(\sum_{l=1}^{L} w_{il}\sigma\left(\sum_{j=1}^{n} v_{lj}x_j + v_{l0}\right) + w_{i0}\right); i = 1,2, \dots, m \qquad (2\text{-}1)$$

where v_j represent dendrite weights for the hidden layer, v_{l0} is the firing threshold or bias, ω_{il} represent the dendrite weights for the output layer, ω_{i0} is the firing threshold or bias for the output layer, and $\sigma(\cdot)$ is a differentiable activation function. The derivative of the activation function is needed for the ANN learning algorithm.

It is computationally convenient to represent the ANN output in a matrix form; with second layer weight matrix $\bar{V}$ and third layer weight matrix $\bar{W}$ given by

$$\bar{V}^T = \begin{bmatrix} v_{10} & v_{11} & v_{12} & \cdots & v_{1n} \\ v_{20} & v_{21} & v_{22} & \cdots & v_{2n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ v_{L0} & v_{L1} & v_{L2} & \dots & v_{Ln} \end{bmatrix}$$

$$\bar{W}^T = \begin{bmatrix} w_{10} & w_{11} & w_{12} & \cdots & w_{1L} \\ w_{20} & w_{21} & w_{22} & \cdots & w_{2L} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ w_{m0} & w_{m1} & w_{m2} & \dots & w_{mL} \end{bmatrix} \qquad (2\text{-}2)$$

Then the output from the ANN becomes

$$y = \bar{\sigma}\big(\bar{W}^T \sigma(\bar{V}^T \bar{x})\big) \qquad (2\text{-}3)$$

with activation function vector given by

$$\sigma(\beta) \equiv [1 \;\; \bar{\sigma}(\beta)^T]^T \equiv [1 \; \sigma(\beta_1) \; \sigma(\beta_2) \; \sigma(\beta_3) \; \dots \sigma(\beta_L)]^T \qquad (2\text{-}4)$$

where the first entry is 1 to account for the threshold or bias inputs, and $\beta \in \mathbb{R}^L$ is the input vector to the activation function $\bar{\sigma} \in \mathbb{R}^{L+1}$, $\bar{x} = [1 \; x^T]^T \in \mathbb{R}^{n+1}$ is the augmented ANN input. These matrices are sometimes referred to as the augmented matrices as they account for threshold weights and inputs.

In this research, the output activation function is chosen to be linear. In that sense, the output from the ANN becomes

$$y = \bar{W}^T \sigma(\bar{V}^T x) \qquad (2\text{-}5)$$

The bar over the augmented matrices and vectors will be dropped from this point and on, as they will always be used in the augmented form.

2.1.1.1 Linear-In-the-Parameter (LIP) ANN

If the weights and the activation function for the hidden-layer, $\sigma(\bar{V}^T x)$, are known a priori, then the ANN is only defined by output-layer weights and threshold, $\bar{W}^T$. Then the ANN is called functional-link ANN, and $\phi(x) = \sigma(\bar{V}^T x)$ is called basis-function. In this sense, the ANN is linear-in-the-parameter W, and is easier to be taught compared with three-layer-ANN. Functional-link-ANNs allow for a more general activation function options, especially if the elementary components of the function to be estimated are accurately known, e.g. sinusoidal, quadratic …etc..

In addition, a non-diagonal activation function could be used in this context as well. In general, LIP ANNs are found very applicable for control systems applications. There are many types of LIP ANNs such as, Radial-Basis-Function ANN, and Cerebellar Model Articulation Controller ANN.

2.1.1.2 Stochastic-Basis-Functional-Link ANN

The Random Vector version of the functional-link ANN consists of a one hidden layer of the form Eq. (2-5), where the parameter V of the hidden layer is selected randomly and independently in advance with a suitable activation function σ, and the output layer weights W are learnt using an appropriate learning rule.

It is proven in the literature that SBFL are general function approximators, and a comprehensive study about their validation is provided in [39].

2.1.2 ANN Learning Algorithms

The ability to learn is a fundamental feature of intelligent systems. The learning process in the ANN context could be defined as the process of updating ANN architecture and connection weight values so that an ANN can efficiently perform the targeted task [74]. Given an ANN structure, the task then is to calculate ANN weights such that the ANN closely models the targeted function. Although there are direct methods to calculate ANNs, such as Least Squares method, they usually assume certain relationship between input and outputs. Therefore, for complex ANN structures and complex targeted functions, iterative learning is inevitable. Iterative learning refers to the process of iteratively modifying ANN weights such that better approximation is achieved at each learning step. Generally, there are three main types of learning:

Supervise Learning: where the ANN is provided with a data set consisting of input vectors and expected or desired outputs associated with each input vector. This data is referred to as the training set. The aim of supervised learning is to adjust ANN weights such that the error between the output from the ANN and the target output is minimized.

Unsupervised Learning: where the aim is to discover functions, patterns, or features in the input data with no assistance from preexisting data.

Reinforcement Learning: is a variation of supervised learning in which the ANN is provided with only a critique on the correctness of network outputs, not the expected true outputs themselves. This is often referred to as critique actor method.

There are four basic types of training Rules:

Gradient Descent Learning Rule:

It is regarded as the most used training rule for ANNs. Gradient Descent requires the definition of an error objective function to neuron's error in approximating the targeted output. One choice for the objective function is the sum of squared errors. The goal of the gradient descent is to find the values for ANN weights such that the error objective function is minimized. This is achieved by calculating the gradient of the objective function in the weight space, and to move the weight vector along the negative gradient (downhill), with a less than one step size, called learning rate.

Widrow-Hoff Learning Rule:

The Widrow-Hoff learning rule, also known as the Least-Means-Squares algorithm, was one of the first algorithms to be used for training layered Neural Networks with multiple adaptive liner neurons [73]. It assumes that the gradient of the activation function with respect to ANN weights is unity, thus linear activation functions are used, to avoid singularity in the case of discontinuous activation functions.

Generalized Delta learning Rule:

The generalized delta learning rule is a generalization of the Widrow-Hoff learning rule that assumes differentiable activation functions.

Error-Correction Learning Rule:

The basic principle of the error-correction learning rule is to use the error signal to modify connection weights to gradually reduce this error. It assumes binary-valued activation functions, and weights are only adjusted when the perceptron makes an error.

The use of ANNs in closed loop control systems imposes extra requirements for the learning algorithm, in contrast to their use in Machine Learning, such as, stability of learning, boundedness of ANN weights, and more importantly, speed of learning. These characteristics are particularly important in the application of closed loop control, because the output of the ANN influences the behavior of the entire system, as the ANN, in that context, represents a sub-dynamical system of the entire closed loop system. Therefore, guarantees for stability, boundedness, and learning dynamics should be provided prior application of ANN in control systems. In this research, an ANN learning algorithm is developed that guarantees stability of the closed loop system and boundedness of ANN weights. Furthermore, the design parameters for the developed algorithm could be intuitively adjusted to achieve certain learning dynamics.

2.2 SLIDING MODE CONTROL

The use of sliding mode control in Variable Structure Control (VSC) systems was introduced in 1964 by Emel'yanov [75]. Since then sliding mode has been a very important method for the deterministic control of uncertain and nonlinear systems [75].

Sliding motion occurs when the system state repeatedly crosses and immediately re-crosses a switching manifold, as a result of all the motion in the neighborhood of the manifold being directed inwards. This motion occurs on individual switching surfaces in the state space, or in all of them at the same time, until the last of them is reached, then the system is said to be in the *sliding mode,* Figure 2.5.

Sliding mode refers to the sliding subspace where the dynamic motion of the system is constrained to lie within a certain subspace of the full state space, and is efficiently described by an unforced system of lower order, called the *equivalent system*. The dynamic behavior of the equivalent system is different from that of each of the constituent subsystems.

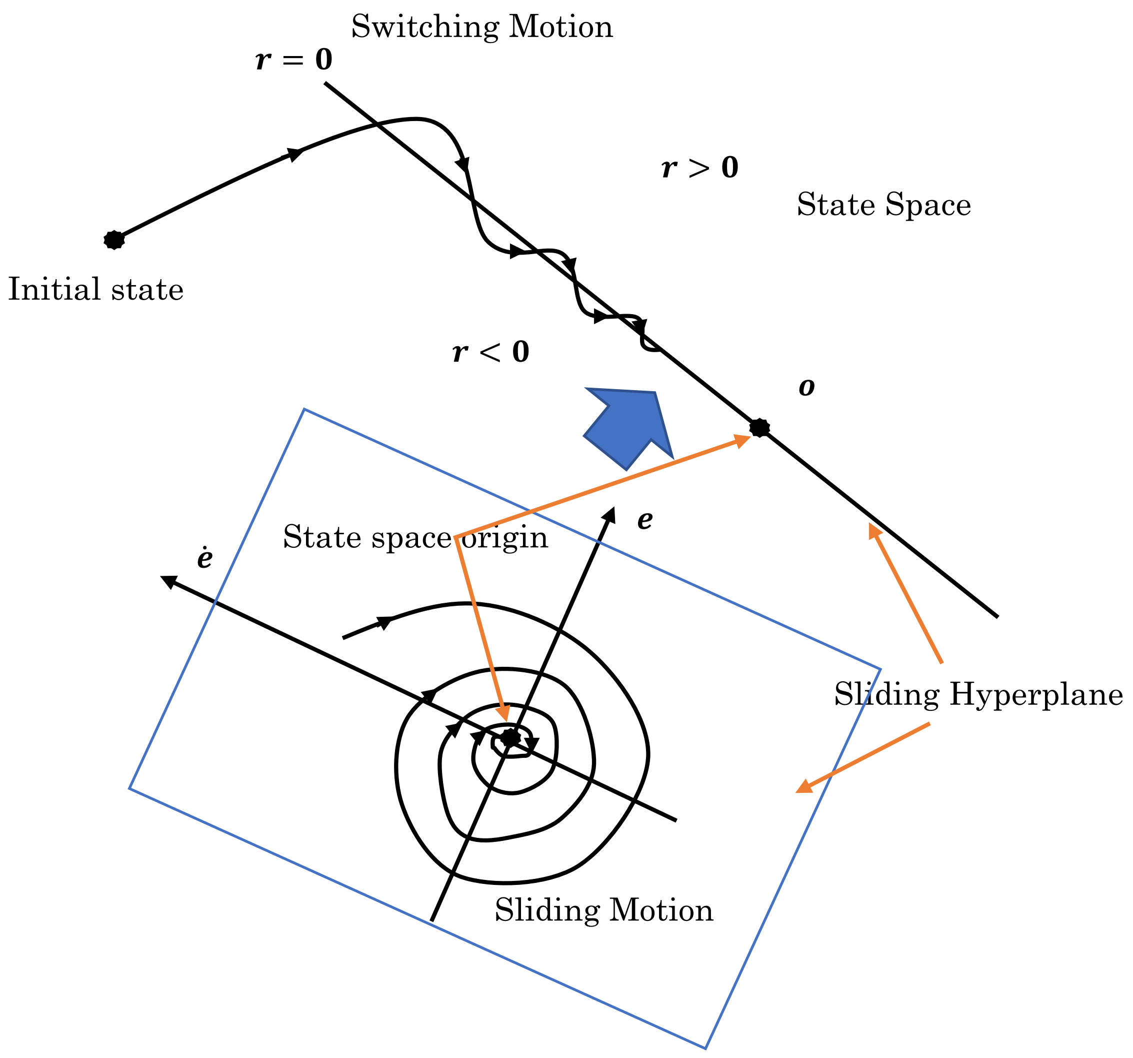


Figure 2.5 Illustration of Sliding Motion

The switching surfaces are usually fixed hyperplanes in the state space passing through the state space origin, the intersection of which forms the sliding subspace. The objective of the design is to drive the state of the system from an arbitrary initial condition in the state space to the sliding subspace. Once reached, the action of the control is only required to maintain the state on the intersection manifold. The equivalent system is required to be asymptotically stable in order to guarantee that the state of the system will reach the state space origin, while on the sliding mode.

The transient behavior of the sliding motion consists of two stages: a fast motion bringing the state of the system to the sliding subspace, in which sliding occurs; and a slower sliding motion during which system's state slides towards the state space origin, while restricted on the sliding subspace.

The switching logic is originally proposed as a discontinuous control function, which causes control chattering, at a high frequency, when shifting between switching surfaces. This phenomenon is undesired in practical applications and could be fixed by implementing a smooth continuous nonlinear control function for the switching process instead.

2.2.1 Siding Mode Control Design Procedure

The sliding mode design procedure starts with selection of the slow dynamics that drives the system's state on the sliding manifold from an initial state to equilibrium. This dynamics transforms the system from $\boldsymbol{x}$ to a reduced order system on $\boldsymbol{r}$. When $\boldsymbol{r} \to \mathbf{0}$, the system's motion is described by the sliding mode. $\boldsymbol{r}$ is selected such that $\dot{\boldsymbol{r}}$ contains the system's global input $\boldsymbol{u}$. A common choice for $\boldsymbol{r}$ is

$$\boldsymbol{r} = \dot{\boldsymbol{x}} + \boldsymbol{\Lambda x} \tag{2-6}$$

The following step is to design the control input $\boldsymbol{u}$ on $\boldsymbol{r}$ that drives the state of the system from an initial value in the state space to the sliding hypersurface, and force it to stay there. These dynamics are commonly selected as a discontinuous function which causes controller chattering. Continuous switching functions are also developed to avoid chattering with some adjustments to the original design method. In this research, a continuous nonlinear control algorithm is developed following backstepping design approach. The control input is designed such that:

- $\boldsymbol{r}\dot{\boldsymbol{r}} < \mathbf{0}$ outside the sliding hypersurface.
- $\boldsymbol{r} = \mathbf{0}$ on the hypersurface.

Upon selecting an appropriate sliding hyperplane, the focus of the control problem shifts towards designing stable dynamics for $\boldsymbol{r}$ that drives it rapidly from its initial value to zero. Then the system is guaranteed stability as $\boldsymbol{\Lambda}$ is required to be Hurwitz.

2.3 BACKSTEPPING CONTROLLER DESIGN APPROACH

Backstepping design approach is well regarded in the design of controllers for nonlinear systems, because of its simplicity and intuitive nature. It was first introduced, in its current form, by Kanellakopoulos *et. al.* [76] in 1991, since then the method has been heavily used in research and industry for the design of nonlinear control systems.

Backstepping requires the dynamical system to be expressed in a specific mathematical form, namely, *strict feedback* form; where each state variable of the system is a strict function of itself and the higher order state variable, and system's input only appears in the time derivative of the highest order variable. While it is very strict, methods have been developed to transform systems of other forms into strict feedback, e.g. [45].

The design process starts with defining the desired state and the required dynamics for the first variable, which also represents system's output. Then, using the equation of the first time-derivative, a virtual input is required by the higher order variable, to achieve both stability and required performance. Similarly, the desired state for the second order variable is the virtual input to the time derivative of the first order variable, and with the desired dynamics for the second order variable being defined, the equation for the derivative of the second order variable is used to calculate the virtual input required by the third order variable. The process continues backwards until the highest order equation is reached, then the global input to the system is designed such that stability is guaranteed and desired dynamics is achieved.

The nature of the design process that starts with the lowest subsystem, and backwards towards the highest order lead to the name *Backstepping*. Control Lyapunov Function (CLF) [64] is usually used to calculate virtual inputs required at each step and the final control input to the system, other methods such as feedback linearization, optimal control are used as well.

The nonlinear continuous-time system in strict-feedback form is as follows

$$\dot{x}_i = f_i(x_1, \dots, x_i) + g_i(x_1, \dots, x_i)x_{i+1}, \;\; for\; 1 \le i \le N-1 \; and \; N \ge 2 \tag{2-7}$$

$$\dot{x}_N = f_N(x_1, \dots, x_i) + g_N(x_1, \dots, x_N)u \tag{2-8}$$

$$y = x_1 \tag{2-9}$$

where, $x_i \in \Re$ is a system state variable, $u \in \Re$ represents system's input, $f_i \in \Re$ is a smooth internal dynamic function, $g_i \in \Re$ is a smooth output function and $y \in \Re$ is system's output.

The first step is to define the required dynamics for output error $e_1 = x_{1_d} - x_1$. Then, equation (2-7), at $i = 1$, is solved for the virtual input, x_{2_d}, required by, x_2, to satisfy response requirements and to assure stability of x_1. A nonlinear or a linear control method could be used to calculate x_{2_d}, however, CLF is mostly used. Step two is to solve equation (2-7), at $i = 2$, for x_{3_d} following the same method used previously. Same steps are followed until the N^{th} equation is reached, then the global control input is solved for, such that, the stability of the entire system is guaranteed.

In this research, the backstepping approach is followed to design a controller for Quadrotors that stabilizes the system and guarantees well tracking of desired trajectories. More details about the Backstepping design approach could be found in [76] [64]. The details of Backstepping approach implementation in this research are provided in Chapter 3.

2.4 STABILITY OF NONLINEAR SYSTEMS

2.4.1 Lyapunov Stability

In 1892 the Russian mathematician Alexander Mikhailovich Lyapunov presented his doctoral thesis "On the General Problem of the Stability of Motion" at the University of Moscow [77], where he had introduced basic definitions and fundamental theorems for evaluating the stability of solutions of a broad spectrum of differential equations. Lyapunov stability theorem is an inevitable tool for analyzing the behavior of system trajectories near equilibrium, without the need to calculate the explicit solution for system equations.

Consider the non-autonomous unforced dynamical system given by

$$\dot{x} = f(t, x) \tag{2-10}$$

with an equilibrium point at the origin of the state space of the system, hence

$$f(t, 0) = 0, \qquad \forall\, t \geq 0 \tag{2-11}$$

Definition (2-1): Stability of the Equilibrium in the Lyapunov Sense

The equilibrium point, $x^{\star} = 0$, of the nonautonomous unforced system Eq. (2-10) is stable if for any $\varepsilon > 0$ and $t_0 \geq 0$ there exists $\delta(\varepsilon, t_0) > 0$ such that for all initial conditions $\|x(t_0)\| < \delta$ and for all $t \geq t_0 \geq 0$, the corresponding system trajectories are bounded by $\|x(t)\| < \varepsilon$. The equilibrium is uniformly stable if it is stable and δ does not depend on t_0. Finally, the equilibrium is unstable if it is not stable.

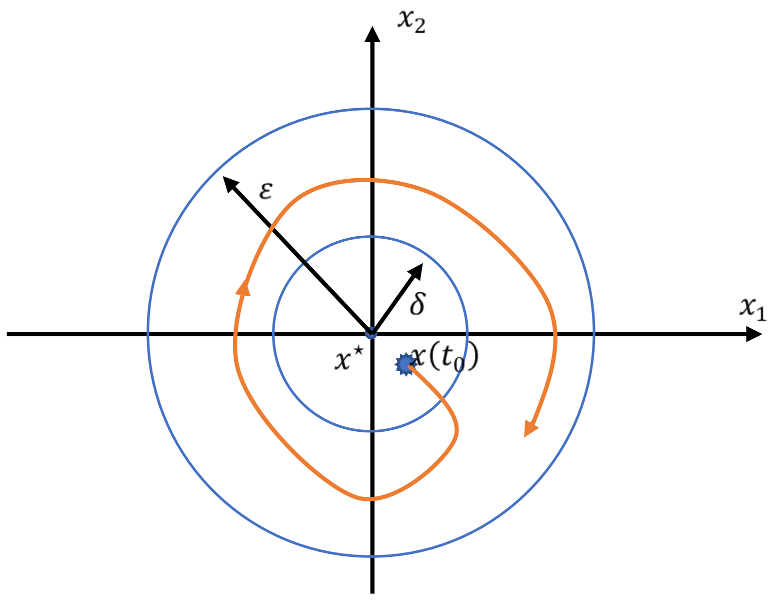


Figure 2.6 Lyapunov Stability

Definition (2-2): Global Stability

The origin is globally stable if it is stable in the Lyapunov sense and $\lim_{\varepsilon \to \infty} \delta(\varepsilon, t_0) = \infty$.

Definition (2-3): Asymptotic Stability

The equilibrium point $x^\star = 0$ of Eq. (2-10) is asymptotically stable if it is stable and there exists a positive constant $c = c(t_0)$ such that $x(t) \to 0$ as $t \to \infty$, for all $\|x(t_0)\| \le c$.

Definition (2-4): Uniform Asymptotic Stability

The equilibrium point $x^\star = 0$ of Eq. (2-10) is uniformly asymptotically stable if it is asymptotically stable and the constant c is independent of t_0.

Definition (2-5): Global Uniform Asymptotic Stability

The origin is globally uniformly asymptotically stable if it is uniformly asymptotically stable and $\lim_{\varepsilon \to \infty} \delta(\varepsilon) = \infty$.

Definition (2-6): Positive-Definite and Semidefinite Functions

A scalar function $V(x): \mathbb{R}^n \to \mathbb{R}$ of a vector argument $x \in \mathbb{R}^n$ is called locally positive definite if $V(0) = 0$, and there exists a constant $r > 0$ such that $V(x) > 0$, for all nonzero $x \in \mathbb{R}^n$ from the r-neighborhood of the origin $B_r = \{x \in \mathbb{R}^n: \|x\| \le r\}$, the function is said to be globally positive definite if $B_r = \mathbb{R}^n$. If $V(x) \ge 0$, then the function is said to be positive semidefinite.

Definition (2-7): Negative-Definite and Semidefinite Functions

A scalar function $V(x): \mathbb{R}^n \to \mathbb{R}$ of a vector argument $x \in \mathbb{R}^n$ is called locally negative definite if $V(0) = 0$, and there exists a constant $r > 0$ such that $V(x) < 0$, for all nonzero $x \in \mathbb{R}^n$ from the r-neighborhood of the origin $B_r = \{x \in \mathbb{R}^n: \|x\| \le r\}$, the function is said to be globally positive definite if $B_r = \mathbb{R}^n$. If $V(x) \le 0$, then the function is said to be negative semidefinite.

Theorem (2-1): Lyapunov Direct Method for Stability

Let $x^\star = 0 \in \mathbb{R}^n$ be an equilibrium point for the nonautonomous dynamics Eq. (2-10), whose initial conditions are drawn from a domain $D \subset \mathbb{R}^n$, with $x^\star \in D$ and $t_0 = 0$. Suppose that on the domain D there exists a continuously differentiable locally positive definite function $V(x): D \to \mathbb{R}$, whose time derivative along the system trajectories is locally negative semidefinite:

$$\dot{V}(x) = \nabla V(x) f(t,x) \leq 0 \tag{2-12}$$

for all $t \geq 0$ and for all $x \in D$. Then the equilibrium of the system $x^{\star} = 0$ is locally uniformly stable in the Lyapunov sense. Furthermore, if $\dot{V}(x) < 0$ for all nonzero x and for all $t \geq 0$, then the origin is locally uniformly asymptotically stable in the Lyapunov sense.

Definition (2-8): Uniform Ultimate Boundedness

The solutions of Eq (2-10) are uniformly ultimately bounded with ultimate bound b if there exist positive constants b and c, independent of $t_0 \geq 0$, and for every $a \in (0, c)$, there is a constant $T = T(a, b)$, independent of t_0, such that

$$\|x(t_0)\| \leq a \Rightarrow \|x(t)\| \leq b, \forall t \geq t_0 + T \tag{2-13}$$

These solutions are said to be globally uniformly ultimately bounded if Eq (2-13) hold for an arbitrarily large a.

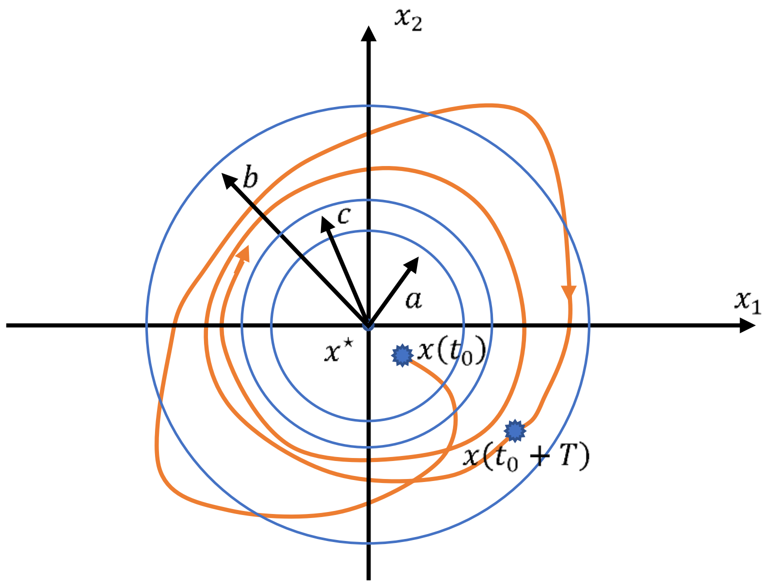


Figure 2.7 Uniform Ultimate Boundedness

2.5 QUADROTOR DYNAMICS MODEL

The quadrotor dynamical model used in this research is derived in [69] following Euler-Lagrange modeling method. Consider the quadrotor system shown in Figure 2.8.

The generalized coordinates of the quadrotor are

$$q = (x, y, z, \theta, \phi, \psi) \in \mathbb{R}^6 \tag{2-14}$$

where $(x, y, z) = \xi \in \mathbb{R}^3$ denote the position of the center of mass of the quadrotor with respect to the inertial frame, and $(\theta, \phi, \psi) = \eta \in \mathbb{R}^3$ are the Euler angles, (pitch, roll, yaw), and they denote the attitude of the quadrotor about its center of mass.

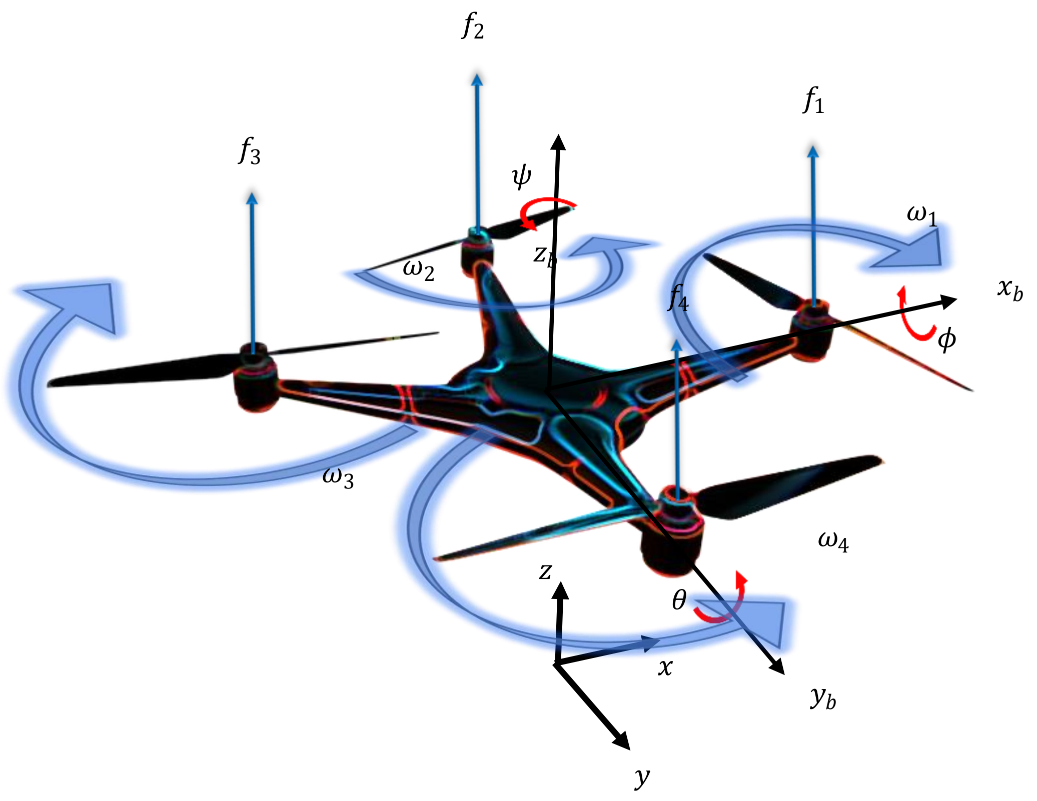


Figure 2.8 Typical Quadrotor Model

The Lagrangian of the quadrotor is given by

$$\mathcal{L}(q,\dot{q}) = T_{trans} + T_{rot} - U \qquad (2\text{-}15)$$

where $T_{trans} = \frac{m}{2}\dot{\xi}^T\dot{\xi}$ is the translational kinetic energy , $T_{rot} = \frac{1}{2}\dot{\eta}^T J\dot{\eta}$ is the rotational kinetic energy, and $U = mgz$ is the potential energy of the quadrotor, with m represents the mass of the quadrotor, z the altitude, g the gravitational acceleration, and J denotes the inertia matrix.

The dynamical model for the quadrotor is then obtained using the Euler-Lagrange method with external generalized forces

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}} - \frac{\partial \mathcal{L}}{\partial q} = (F_\xi, \tau_\eta) \tag{2-16}$$

where $F_\xi = R\hat{F} \in \mathbb{R}^3$ is the translational forces applied to the quadrotor due to throttle control input, $\tau_\eta \in \mathbb{R}^3$ denotes applied pitch, roll and yaw torques, and $R(\theta, \phi, \psi) \in SO(3)$ represents the rotational matrix of the quadrotor with respect to the inertial frame. The force vector $\hat{F}$ is given by

$$\hat{F} = \begin{pmatrix} 0 \\ 0 \\ u \end{pmatrix} \tag{2-17}$$

where u is the total rotors input given by

$$u = \sum_{i=1}^{4} f_i \tag{2-18}$$

and f_i is the force produced by the i^{th} rotor. Generally, $f_i = k_i\omega_i^2$, where k_i is the force constant for the i^{th} rotor, and ω_i represents the rotational velocity of the i^{th} motor.

The generalized torques are given by

$$\tau_\eta = \begin{pmatrix} \tau_\theta \\ \tau_\phi \\ \tau_\psi \end{pmatrix} \triangleq \begin{pmatrix} (f_1 - f_3)l \\ (f_2 - f_4)l \\ \sum_{i=1}^{4} \tau_{\psi_i} \end{pmatrix} \tag{2-19}$$

where l is the distance between the motors to the center of mass, and τ_{ψ_i} is the yaw torque produced by the i^{th} motor about the center of mass of the quadrotor.

Since the Langrangian contains no cross terms in the kinetic energy involving $\dot{\xi}$ and $\dot{\eta}$, the Euler-Lagrange equation could be partitioned into two parts. One represents the dynamics along the ξ coordinates, and the second for the dynamics at the η coordinate. Then, the equations of motion of the quadrotor is given by

$$m\ddot{\xi} = -\begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} + F_{\xi} \tag{2-20}$$

$$J(\eta)\ddot{\eta} = -C(\eta,\dot{\eta})\dot{\eta} + \tau_{\eta} \tag{2-21}$$

with

$$J(\eta) = T_{\eta}^{T}\Sigma T_{\eta}, \qquad T_{\eta} = \begin{bmatrix} -\sin\theta & 0 & 1 \\ \cos\theta\sin\psi & \cos\psi & 0 \\ \cos\theta\cos\psi & -\sin\psi & 0 \end{bmatrix}, \qquad \tau_{\eta} = \begin{bmatrix} l & 0 & -l & 0 \\ 0 & -l & 0 & l \\ c & -c & c & -c \end{bmatrix}\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix}$$

The vector $C(\eta,\dot{\eta})\dot{\eta}$ is the Coriolis/Centripetal term [53]. Assuming a symmetric inertia matrix Σ, for a typical quadrotor $C(\eta,\dot{\eta})$ is given by:

$$C(\eta,\dot{\eta}) = 2T_{\eta}^{T}\Sigma\dot{T}_{\eta} - \frac{1}{2}\frac{\partial}{\partial\eta}\left(\dot{\eta}^{T}J(\eta)\right) \tag{2-22}$$

Matrix $C(\eta,\dot{\eta})$ is not unique [78]. However, it is shown in [55] that using the above expression $\left(\frac{d}{dt}J(\eta) - 2C(\eta,\dot{\eta})\right)$ is skew-symmetric. This property is needed later in stability analysis.

CHAPTER 3

CONTROLLER DESIGN

In this chapter, the controller design following sliding mode methodology and backstepping approach is detailed. In addition, the proposed ANN structure used in the controller algorithm is presented with the accompanied mathematical expression. Furthermore, the derivation of the ANN learning algorithm is provided. And finally, the stability of the developed controller with the proposed ANN structure, and the developed learning algorithm is analyzed using Lyapunov theorem.

3.1 CONTROLLER DESIGN

3.1.1 Modified Quadrotor Dynamics Model

In this section, the dynamical model of quadrotors, which is detailed in Chapter 2, is modified for the purpose of controller design. In summary, the Lagrange dynamics model of a typical quadrotor, shown in Fig. 3.1, is given by

Position dynamics

$$m\ddot{\xi} = M_g + F_d \tag{3-1}$$

Attitude dynamics

$$J(\eta)\ddot{\eta} = -C(\eta,\dot{\eta})\dot{\eta} + \tau \tag{3-2}$$

where $\xi = [x,y,z]^T \in \mathbb{R}^3$ is the inertial position of the center of gravity, $M_g = [0,0,-mg]^T$ is the gravitational force, and $\eta = [\phi,\theta,\psi]^T \in \mathbb{R}^3$ represents quadrotor attitude, roll, pitch, and yaw.

$$F_d = u \begin{bmatrix} -\sin\theta \\ \cos\theta\sin\phi \\ \cos\theta\cos\phi \end{bmatrix}, u = \sum_{i=1}^{4} f_i,$$

$$f_i = k_{\omega_i}\omega_i^2 \qquad (3\text{-}3)$$

Here f_i is the force provided by the i^{th} rotor, k_{ω_i} is the force constant, and ω_i is rotational speed for the i^{th} rotor. It is shown in the literature that the force provided by each rotor is far more complicated than the expression shown above [4] [61]. However, by far Eq. 3-3 is the most commonly used model. In most of the cases, quadrotor thrust force is experimentally modeled; to closely fit the specific combination of Speed Controller, motor, and propeller.

$$J(\eta) = T_\eta^T \Sigma T_\eta, \qquad T_\eta = \begin{bmatrix} -\sin\theta & 0 & 1 \\ \cos\theta\sin\psi & \cos\psi & 0 \\ \cos\theta\cos\psi & -\sin\psi & 0 \end{bmatrix}, \qquad \tau = \begin{bmatrix} l & 0 & -l & 0 \\ 0 & -l & 0 & l \\ c & -c & c & -c \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix}$$

Here, l is the distance from the center of the rotor to the center of gravity of the quadrotor, Σ is the constant quadrotor inertia matrix, and c is a constant known as force-to-moment scaling factor caused by the reactive force on the rotor stator.

The control input to the quad rotor is collected in one vector as:

$$\begin{bmatrix} u \\ \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ l & 0 & -l & 0 \\ 0 & -l & 0 & l \\ c & -c & c & -c \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} = \Gamma f \qquad (3\text{-}4)$$

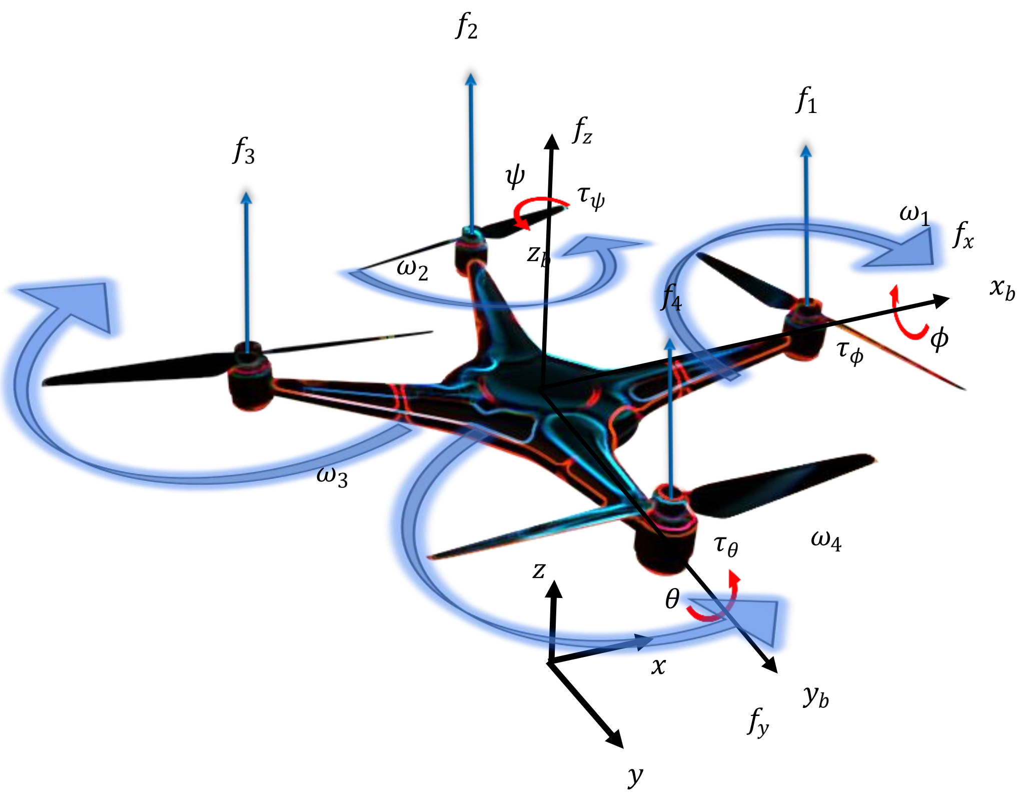


Figure 3.1 Quad-Rotor Model with Aerodynamics Uncertainties

The vector $C(\eta,\dot{\eta})\dot{\eta}$ is the Coriolis/Centripetal term [53]. Assuming a symmetric inertia matrix Σ, for a typical quadrotor $C(\eta,\dot{\eta})$ is given by:

$$C(\eta,\dot{\eta}) = 2T_\eta^T\Sigma\dot{T}_\eta - \frac{1}{2}\frac{\partial}{\partial\eta}\left(\dot{\eta}^T J(\eta)\right) \tag{3-5}$$

Matrix $C(\eta,\dot{\eta})$ is not unique [79]. However, it is shown in [55] that using the above expression $\left(\frac{d}{dt}J(\eta) - 2C(\eta,\dot{\eta})\right)$ is skew-symmetric. This property is needed later in stability analysis.

This system is a coupled-Lagrangian form, under-actuated with six outputs and four inputs.

To account for unknown aerodynamic forces and moments associated with aggressive maneuverability and environment impact [53] [3] [65] [4], quadrotor dynamics are modified to Eq. 3-6 and Eq. 3-7. These forces and moments are difficult to calculate and expected to be highly nonlinear [61]. The adjusted Lagrange dynamic equations for the quadrotor become:

Position dynamics

$$m\ddot{\xi} = M_g + F_d + A_\xi(\chi_1) \tag{3-6}$$

Attitude dynamics

$$J(\eta)\ddot{\eta} = -C(\eta, \dot{\eta})\dot{\eta} + \tau + A_\eta(\chi_2) \tag{3-7}$$

where χ_1 and χ_2 are the explanatory variables for the unknown aerodynamics, chosen based on the influence on the corresponding dimension, and will be detailed later. For brevity, the arguments of the aerodynamic forces and moments will be dropped for the rest of this dissertation.

3.1.2 Backstepping Controller Design

Backstepping controller design approach is widely used for the control of nonlinear systems [64], especially when ANN are used [31] [20]. In this research, the method is directly applied to a sliding mode derived from the quadrotor dynamics in Lagrange form. This is not a straightforward process, because of the fact that Eq. 3-5 is bilinear with respect to control inputs, due to the first term in the right-hand side of the equation [53]. Considering this bilinear term as a virtual control input leads to complications [80]. The problem that arises is one of control input allocation due to more than one solution existing. Figure 3.3 shows the block diagram for the designed controller.

The objective of the controller design is to make the quadrotor follow a desired smooth position (at least twice differentiable with respect to time). Controller design procedure followed here is based on that described in [55]. Which is detailed as follows:

Step One: starting with the trajectory dynamics, define the tracking error and the desired "sliding mode error" respectively, as follows

$$e_1 = \xi_d - \xi \tag{3-8}$$

$$r_1 = \dot{e}_1 + \Lambda_1 e_1 \tag{3-9}$$

where, Λ_1 is a diagonal positive definite design parameter matrix (Hurwitz). It is desired to make sliding mode error Eq. 3-9 be zero or very small, thus system's response would follow the sliding motion defined by Λ_1. This parameter matrix is selected for the desired response on the sliding hypersurface

$$e_1(t) = e^{-\Lambda_1 t} e_1(0) \tag{3-10}$$

Substituting quadrotor dynamics Eq. 3-6 and Eq. 3-7 into Eq. 3-9, the augmented error dynamics becomes

$$m\dot{r}_1 = m\ddot{\xi}_d + m\Lambda_1 r_1 - m\Lambda_1^2 e_1 - F_d - M_g - A_\xi \tag{3-11}$$

Hence, the ideal force (virtual input) that stabilizes the tracking error dynamics is chosen as:

$$\hat{F}_d = m\ddot{\xi}_d - m\Lambda_1^2 e_1 - M_g - \hat{A}_\xi + K_{r_1} r_1 + K_{i_1} \int_0^t r_1 dt' \tag{3-12}$$

Here, $\hat{A}_\xi \in \mathbb{R}^3$ is an approximation of A_ξ, and $K_{r_1} > 0, K_{i_1} > 0$ are diagonal control gains (PID controller gains). Accordingly, the closed loop augmented error dynamics becomes

$$m\dot{r}_1 = -\left(K_{r_1} - m\Lambda_1\right) r_1 + \tilde{A}_\xi - K_{i_1} \int_0^t r_1 dt' \tag{3-13}$$

where, $\tilde{A}_{xyz} = A_{xyz} - \hat{A}_{xyz}$ is approximation error. Eq. 3-13 defines the dynamics of the closed loop system when starting from an arbitrary state until it approaches the sliding mode hypersurface Eq. 3-10. The PID gains K_{r_1} and $K_{i_1}$are selected for the desired augmented error performance.

A_ξ is unknown, therefore, an Artificial Neural Network (ANN) is used to approximate aerodynamic forces. It is proven in [39] [81] that feedforward ANNs are universal function approximators; when properly sized and structured. The ANN representation of unknown aerodynamics is given by

$$v_{NN_1} = -A_\xi \equiv W_1^T \mu_1(\chi_1) + \varepsilon_1 \tag{3-14}$$

where W_1 is the unknown ideal ANN weight matrix, $\chi_1 = \left[\xi, r_1, \dot{\xi}, \eta\right]^T$ is the input to the ANN and $\mu_1(\cdot)$ is a smooth activation function to be selected so that the aerodynamic forces can be captured by the ANN. The reconstruction error ε_1 is assumed to be upper bounded by a positive constant ε_{N_1}, such that $\|\varepsilon_1\| < \varepsilon_{N_1}$.

In contrast to [53], a general stochastic linear-in-the-parameters ANN is proposed in this research [39]. This type of ANN is more accurate than the regularly used two layer ANN, as it uses three layers with only the linear weights being adjusted online, while maintaining the mathematical simplicity. The mathematical model for the proposed ANN is detailed later. Now, adding this ANN estimation to the desired virtual input equation we get

$$\hat{F}_d = m\ddot{\xi}_d - m\Lambda_1^2 e_1 - M_g - \widehat{W}_1^T \mu_1(\chi_1) + K_{r_1} r_1 + K_{i_1} \int_0^t r_1 dt' \qquad (3\text{-}15)$$

and the augmented error dynamics become

$$m\dot{r}_1 = -(K_{r_1} - m\Lambda_1) r_1 - \widetilde{W}_1^T \mu_1(\chi_1) - K_{i_1} \int_0^t r_1 dt' + \varepsilon_1 \qquad (3\text{-}16)$$

where $\widetilde{W}_1^T = W_1^T - \widehat{W}_1^T$ define ANN weights error. ANN weights update algorithms are discussed later.

Step Two is to find total rotor input u and required torque τ to produce $\hat{F}_d$. As shown previously, control variables in F_d are the produced rotor force u and the quadrotor attitude η. Therefore, given the value of $\hat{F}_d$, we should be able to calculate corresponding u_d and η_d. In this step, we shall find τ_d that produces the desired attitude.

Similar to Step One, define the attitude tracking error and sliding mode error as

$$e_2 = \eta_d - \eta \qquad (3\text{-}17)$$

$$r_2 = \dot{e}_2 + \Lambda_2 e_2 \tag{3-18}$$

where Λ_2 is Hurwitz and has similar characteristics as Λ_1. Then designing a controller to keep $\|r_2\|$ small to guarantee that $\|e_2\|$ *and* $\|\dot{e}_2\|$ are small. Similar to the first step in backstepping control design, the control input τ needed to guarantee the stability of attitude tracking dynamics is given by

$$\tau = \hat{v}_{NN_2} + K_{r_2} r_2 + K_{i_2} \int_0^t r_2 dt' \tag{3-19}$$

with K_{r_2} and K_{i_2} are positive diagonal PID gain matrices. And $\hat{v}_{NN_2} = \widehat{W}_2^T \mu_2(\chi_2)$, $\chi_2 = [\eta, \dot{\eta}, r_2, \dot{\xi}]^T$ are the ANN estimation of unknown nonlinearities in augmented error equation and the ANN input respectively. The ANN is used to approximate the following nonlinear function

$$v_{NN_2} \triangleq \{J\ddot{\eta}_d - A_\eta + C(\eta, \dot{\eta})(\dot{\eta}_d + \Lambda_2 e_2) - J\Lambda_2^2 e_2\} \equiv W_2^T \mu_2(\chi_2) + \varepsilon_2 \tag{3-20}$$

Notice here that the ANN is used to estimate difficult to calculate terms in addition to unknown aerodynamics; such globality is a property of ANNs.

Accordingly, the augmented error dynamics becomes

$$J\dot{r}_2 = \widetilde{W}_2^T \mu_2(\chi_2) - \{K_{r_2} + C(\eta, \dot{\eta}) - J\Lambda_2\} r_2 - K_{i_2} \int_0^t r_2 dt' + \varepsilon_2 \tag{3-21}$$

and $\widetilde{W}_2^T$ is the ANN weight error matrix.

Notice that Eq. 3-16 and Eq. 3-21 are stable as long as the uncertainty is bounded and the PID gains are strictly positive with proportional gains chosen such that

$$\begin{gathered}(K_{r_1} - m\Lambda_1) > 0 \\ \{K_{r_2} + C(\eta, \dot{\eta}) - J\Lambda_2\} > 0\end{gathered} \tag{3-22}$$

3.1.3 Solving for $\mathbf{u_d}$ and $\boldsymbol{\eta_d}$

The inverse kinematics method is employed to calculate u_d and η_d for a given $\hat{F}_d$; this method is widely used in the control of robotic manipulators [79] [55]. In contrast to the virtual inputs calculated in [53], the required attitude η_d is calculated taking into account the heading angle, ψ, which has been dropped in [53]. Consider inertial and body coordinate systems, shown in figure 3.2. The required virtual force is calculated using Eq. 3-15, with inertial components $f_{d_{x_I}}$, $f_{y_{d_I}}$, and $f_{z_{d_I}}$ in the inertial X, Y, and Z directions respectively. Then

$$\hat{F}_d = \begin{bmatrix} -u_d \sin\theta_{d_I} \\ u_d \sin\phi_{d_I} \\ u_d \cos\theta_{d_I} \cos\phi_{d_I} \end{bmatrix} = \begin{bmatrix} f_{x_{d_I}} \\ f_{y_{d_I}} \\ f_{z_{d_I}} \end{bmatrix} \tag{3-23}$$

where, the subscript I in desired pitch and roll angles indicates that these angles are calculated using inertial force components, and

$$u_d = \sqrt{f_{x_{d_I}}^2 + f_{y_{d_I}}^2 + f_{z_{d_I}}^2} \tag{3-24}$$

$$\theta_{d_I} = \sin^{-1}\left(-\frac{f_{x_d}}{u_d}\right) \tag{3-25}$$

$$\phi_{d_I} = \sin^{-1}\left(\frac{f_{yd}}{u_d}\right) \tag{3-26}$$

Now, the required inertial forces could be projected on the body fame as a result of the yaw angle, ψ, as follows

$$f_{x_{d_b}} = f_{x_{d_I}} \cos\psi - f_{y_{d_I}} \sin\psi \qquad (3\text{-}27)$$

$$f_{y_{d_b}} = f_{y_{d_I}} \cos\psi + f_{x_{d_I}} \sin\psi \qquad (3\text{-}28)$$

$$f_{z_{d_b}} = f_{z_{d_I}} \qquad (3\text{-}29)$$

Notice that body and inertial frames are identical in the case of zero yaw, $\psi = 0$. The desired control force input, u_d, is not affected by the yaw angle and is equal in the body and inertial frames. If the required attitude angles are calculated using forces in body frame, then

$$f_{x_{d_b}} = -u_d \sin\theta_{d_b} \qquad (3\text{-}30)$$

and

$$f_{y_{d_I}} = u_d \sin\phi_{d_b} \qquad (3\text{-}31)$$

Substituting the X amd Y components of Eq. 3-23, Eq. 3-30 and Eq. 3-31 into Eq. 3-27 and Eq. 3-28, we get

$$-u_d \sin\theta_{d_b} = -u_d \sin\theta_{d_I} \cos\psi - u_d \sin\phi_{d_I} \sin\psi \qquad (3\text{-}32)$$

and,

$$u_d \sin\phi_{d_b} = u_d \sin\phi_{d_I} \cos\psi - u_d \sin\theta_{d_I} \sin\psi \qquad (3\text{-}33)$$

Now, dividing by the control input force, the desired attitude become

$$\theta_{d_b} = \sin^{-1}\left(\sin\theta_{d_I} \cos\psi + \sin\phi_{d_I} \sin\psi\right) \qquad (3\text{-}34)$$

and,

$$\phi_{d_b} = \sin^{-1}\left(\sin\phi_{d_I} \cos\psi - \sin\theta_{d_I} \sin\psi\right) \qquad (3\text{-}35)$$

The attitude angles, calculated with respect to the body frame, Eq. 3-34 and Eq. 3-35, are used to control quadrotor's attitude. First, the attitude angles are calculated based on inertial frame, given inertial virtual force components, using Eq. 3-25 and Eq. 3-26. Then Eq. 3-34 and Eq. 3-35 are used to calculate equivalent desired attitude angles for attitude control. This transformation is necessary because in the case of nonzero heading angle, the desired pitch and roll angles calculated based on inertial forces will deceivingly direct the quadrotor towards the wrong direction. It is assumed that $f_{z_d} \neq 0$ i.e. θ *and* $\phi \neq 90°$. Notice here that $\hat{F}_d$ does not depend on ψ_d, therefore, its value could be selected as a reference input [53].

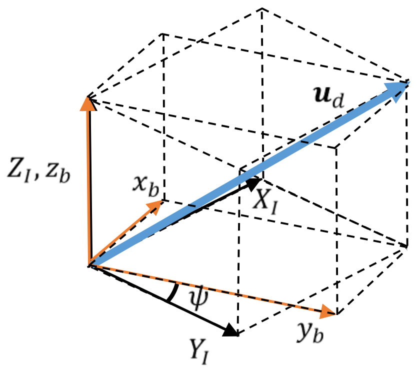


Figure 3.2 Virtual Force Transformation

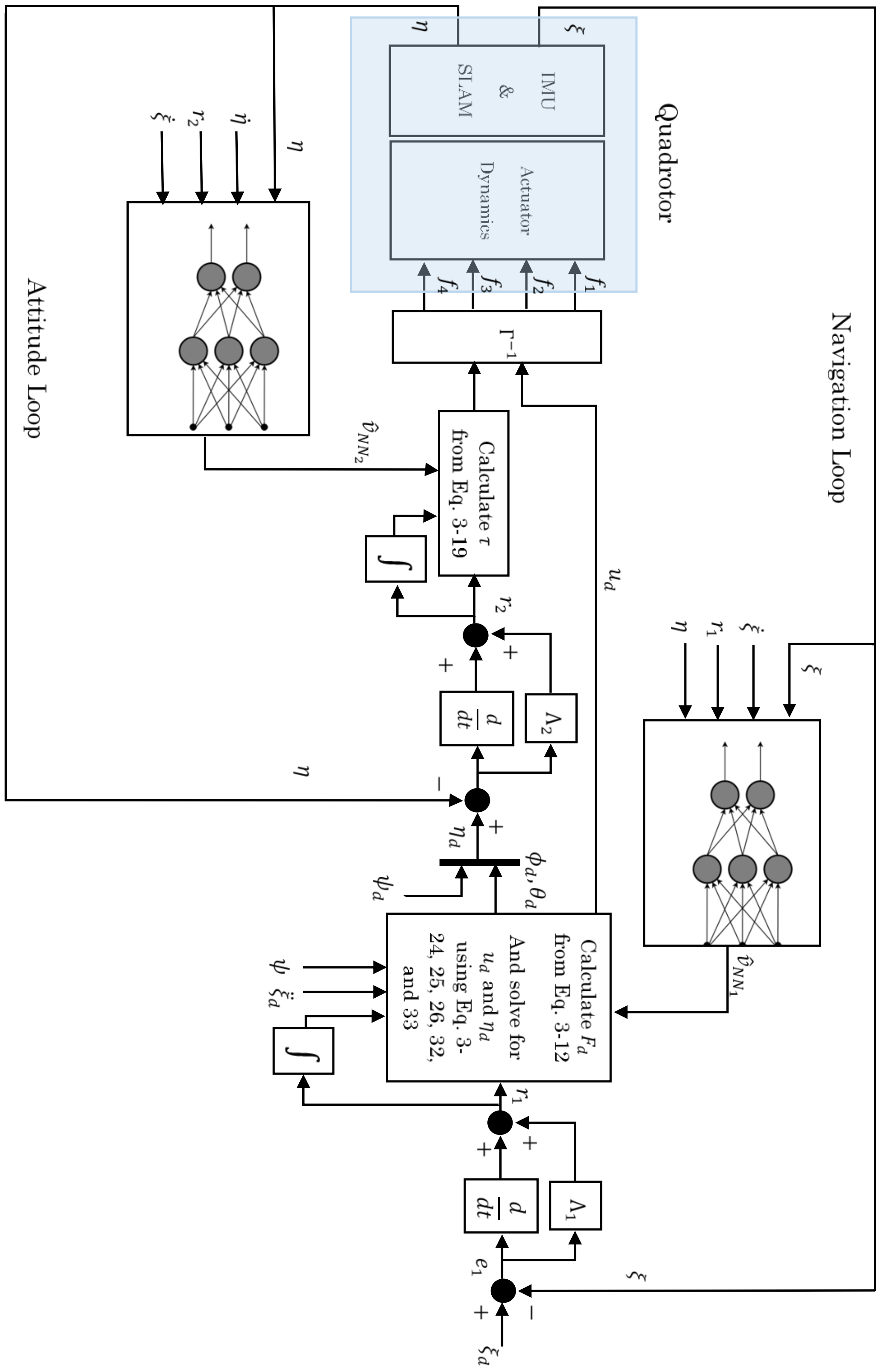


Figure 3.3 ANN Controller Block Diagram

3.2 MODIFIED SBFL ANN

Linear functions could become very complex when estimated using only nonlinear components. Hence, a sum of a large number of these components is required to closely estimate that function. Taylor expansion, for example, aims to estimate nonlinear functions using a linear combination of function's independent variables, when perturbed about an equilibrium state, and for better accuracy, nonlinear terms are added one higher order at a time. This emphasizes on the importance of the linear relationship between system's outputs and system's states, over the nonlinear one. On the other hand, a big number of sinusoidal bases functions are required to approximate the output of a linear function with acceptable accuracy. Accordingly, the feedforward ANN structure is modified to account for the linear relationship between the outputs and inputs of the function; as according to Taylor's expansion method, it could be of more influence on the output than the nonlinear relationship.

The input layer is connected to the hidden layer through weights, and also directly to the output layer through a different set of weights. The output activation function sums the weighted outputs from the input layer and the hidden layer. An illustration is shown in Figure 3.4. This ANN structure is believed to increase accuracy and decrease learning time, compared with conventional ANN structures. Besides, it is a closer model to the biological Neural Network. One Layer ANNs with linear output activation function are known for their fast learning, however, limited approximation accuracy could be achieved. Multilayer ANNs on the other hand, are known for their high approximation accuracy, while requiring more time for learning compared with One Layer ANN [55] [43] [28]. The proposed ANN structure exploits the merits of both types.

The mathematical expression for the output of the Mekky's ANN is as follows

$$y = \bar{W}^T \sigma(\bar{V}^T x) \tag{3-36}$$

where, the weight matrices for the hidden and output layers are given by

$$\bar{V}^T = \begin{bmatrix} v_{10} & v_{11} & v_{12} & \cdots & v_{1n} \\ v_{20} & v_{21} & v_{22} & \cdots & v_{2n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ v_{L0} & v_{L1} & v_{L2} & \ldots & v_{Ln} \end{bmatrix}$$
$$\bar{W}^T = \begin{bmatrix} w_{10} & w_{11} & w_{12} & \cdots & w_{1(L+n)} \\ w_{20} & w_{21} & w_{22} & \cdots & w_{2(L+n)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ w_{m0} & w_{m1} & w_{m2} & \ldots & w_{m(L+n)} \end{bmatrix} \tag{3-37}$$

Notice, that the size of the output weight matrix has changed from its original size, Eq. 2-2, to $m \times (L + n)$. The weights of the hidden layer $\bar{V}$ are randomly selected using the "rand" command in Matlab ®, while the weights of the output layer are adjusted on-line following a learning algorithm. Stability and learning algorithms are discussed in a subsequent section.

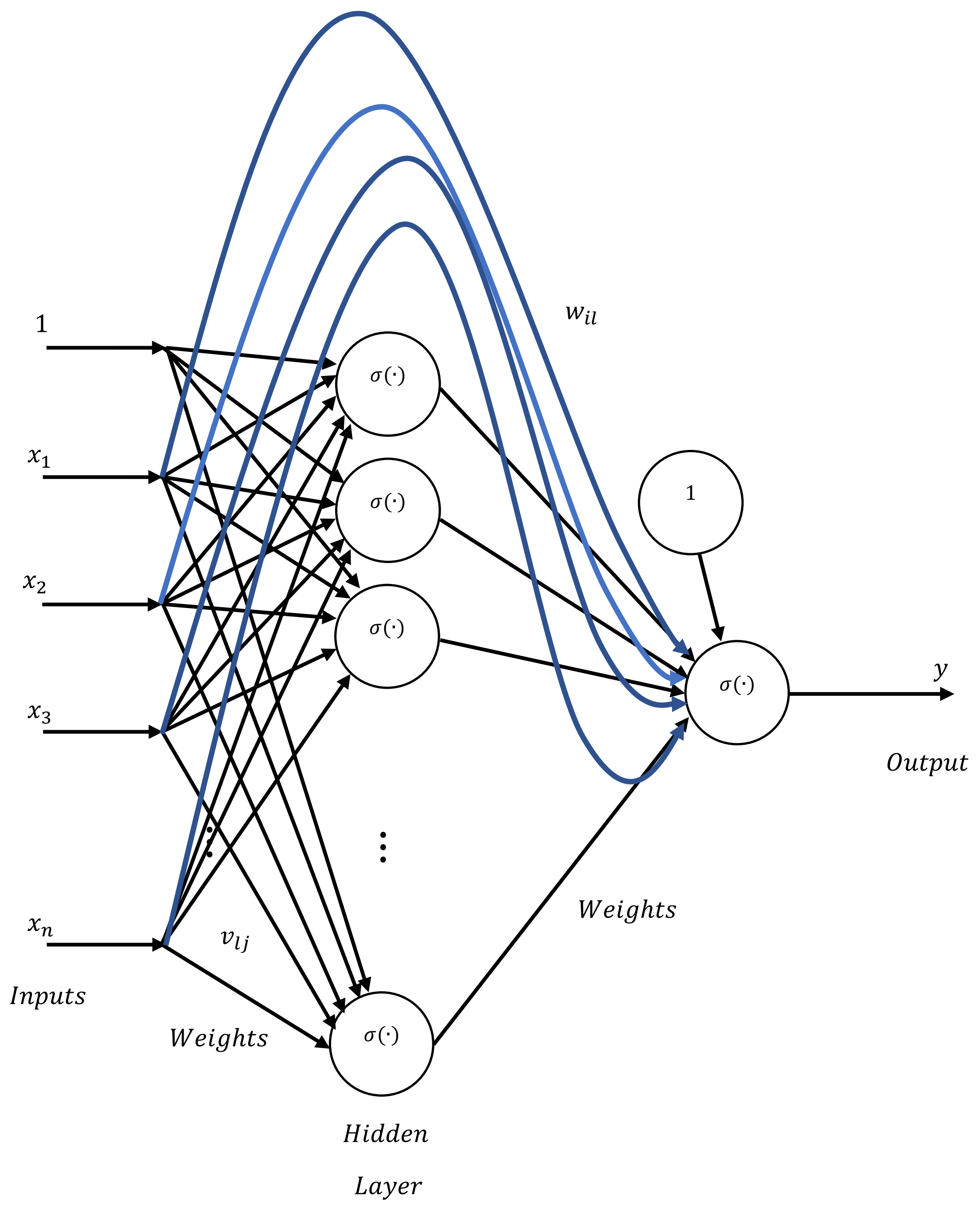


Figure 3.4 Modified SBFL ANN

3.3 STABILITY ANALYSIS AND ANN LEARNING ALGORITHM

The desired "external" command inputs are given by $\xi_d(t) = [x_d(t),\ y_d(t),\ z_d(t)]^T$ and ψ_d. In this research, in contrast to that given in [53] [55] , a novel ANN training algorithm and the Lyapunov-based analysis for the control law are presented. It is assumed that the norm of desired trajectory $q_d = [\xi_d^T\ \psi_d]^T$ is bounded by some bound q_b, the norm of the ideal ANN weights is bounded by W_b and the auxiliary inertia matrix is upper bounded by a positive bound J_M. The validity of those assumptions is straightforward: the first is guaranteed by the choice of desired trajectory, the second is managed by the learning algorithm and is valid in a compact set, and the third is valid because the auxiliary inertia matrix consists of a combination of trigonometric functions.

The challenging stage in the design of ANN controllers is to show that stability is guaranteed and to define its limits. In contrast to the method followed in [55], the learning law is derived as the solution to a quadratic optimization problem. The following section details the derivation of the ANN learning laws.

3.3.1 Learning Algorithm

The learning law for the ANN is developed so as to minimize a quadratic objective function. Starting with trajectory dynamics, the objective function is given by

$$\mathcal{J}_1 = \frac{1}{2}\Upsilon_1^T\Upsilon_1 + \frac{1}{2}\widehat{W}_1^T\kappa_1\widehat{W}_1 \qquad (3\text{-}38)$$

Subject to the augmented error dynamics (3-16),

$$m\dot{r}_1 = \tilde{A}_{xyz} - (K_{r_1} - m\Lambda_1)r_1 - K_{i_1}\int_0^t r_1 dt' \tag{3-16}$$

where Υ_1 is given by the learning sliding mode

$$\Upsilon_1 = m\dot{r}_1 + Q_{r_1}K_{r_1}r_1 + Q_{i_1}K_{i_1}\int_0^t r_1 dt' \tag{3-39}$$

Here, $Q_1 = [Q_{r_1}, Q_{i_1}]$, with $Q_{r_1} \in \mathbb{R}^3$ and $Q_{i_1} \in \mathbb{R}^3$, is chosen so as to make the learning dynamics faster than augmented error dynamics defined by Eq. 3-16. Therefore, Q_1 is Hurwitz with elements $Q_1(i) \geq 1$. Then, the gradient of the objective function with respect to estimated ANN weights $\widehat{W}_1$, is calculated using the chain rule as follows

$$\frac{\partial \mathcal{J}_1}{\partial \widehat{W}_1} = \frac{\partial \mathcal{J}_1}{\partial \Upsilon_1}\frac{\partial \Upsilon_1}{\partial \widehat{W}_1} + \kappa_1\widehat{W}_1 = \Upsilon_1\frac{\partial \Upsilon_1}{\partial \widehat{W}_1} + \kappa_1\widehat{W}_1 \tag{3-40}$$

The gradient of the learning sliding mode Υ_1 with respect to estimated ANN weights $\widehat{W}_1$ is identical to that of the ANN estimation error $\tilde{A}_{xyz}$, accordingly

$$\frac{\partial \mathcal{J}_1}{\partial \widehat{W}_1} = -\mu_1\Upsilon_1 + \kappa_1\widehat{W}_1 \tag{3-41}$$

Substituting the learning sliding mode dynamics, we get

$$\frac{\partial \mathcal{J}_1}{\partial \widehat{W}_1} = -\mu_1\left[m\dot{r}_1 + Q_{r_1}K_{r_1}r_1 + Q_{i_1}K_{i_1}\int_0^t r_1 dt'\right] + \kappa_1\widehat{W}_1 \tag{3-42}$$

It is a common practice to define the ANN weights update law as a scaled version of objective function gradient. Accordingly, the learning dynamics become

$$\dot{\widehat{W}}_1 = -F_1\left(\frac{\partial \mathcal{J}_1}{\partial \widehat{W}_1}\right) = F_1^T\mu_1\left[m\dot{r}_1 + Q_{r_1}K_{r_1}r_1 + Q_{i_1}K_{i_1}\int_0^t r_1 dt'\right] - \kappa_1F_1\widehat{W}_1 \tag{3-43}$$

with the requirement for stability that,

$$\left[K_{r_1} - m\Lambda_1\right] > 0$$

and design learning parameters matrices $F_1 = F_1^T > 0$ and $\kappa_1 > 0$.

Similar steps are followed to derive the learning algorithm for attitude dynamics. Consider the quadratic objective function given by

$$\mathcal{J}_2 = \frac{1}{2}\Upsilon_2^T\Upsilon_2 + \frac{1}{2}\widehat{W}_2^T\kappa_2\widehat{W}_2 \tag{3-44}$$

Subject to the augmented error dynamics (3-21),

$$J\dot{r}_2 = \tilde{A}_{\phi\theta\psi} - \{K_{r_2} + C(\eta,\dot{\eta}) - J\Lambda_2\}r_2 - K_{i_2}\int_0^t r_2 dt' \tag{3-21}$$

where Υ_2 is given by the learning sliding mode

$$\Upsilon_2 = J\dot{r}_2 + Q_{r_2}K_{r_2}r_2 + Q_{i_2}K_{i_2}\int_0^t r_2 dt' \tag{3-45}$$

It follows that

$$\dot{\widehat{W}}_2 = -F_2\left(\frac{\partial\mathcal{J}_2}{\partial\widehat{W}_2}\right) = F_2^T\mu_2\left[J\dot{r}_2 + Q_{r_2}K_{r_2}r_2 + Q_{i_2}K_{i_2}\int_0^t r_2 dt'\right] - \kappa_2F_2\widehat{W}_2 \tag{3-46}$$

with the requirement for stability that,

$$\left[K_{r_2} + C(\eta,\dot{\eta}) - J\Lambda_2\right] > 0$$

and design learning parameters matrices $F_2 = F_2^T > 0$ and $\kappa_2 > 0$.

Notice that this learning law guarantees that Υ_1 and Υ_2 are minimized, which in turn imply that r_1 and r_2 are minimized. The second term in the learning objective function is added to guarantee the boundedness of ANN weights, also referred to as *persistence of excitation* in the literature. The choice of κ governs the tradeoff between performance and boundedness.

3.3.2 Stability Analysis

This part presents the main result in this section.

Theorem 3.1

Given the system described by Eq. 3-16 and Eq. 3-21, and control inputs given by Eq. 3-15 and Eq. 3-19 with the conditions in Eq. 3-22 satisfied. And given the ANN structures defined by Eq. 3-14 and Eq. 3-20, with ANN learning Laws Eq. 3-43 and Eq. 3-46, with design parameters $F_1, F_2, \kappa, Q,$ *and* ϱ *chosen such that*

$$\begin{gathered} F1, F2 > 0 \\ \kappa > 0 \\ \varrho > 0 \\ Q_r \geq \frac{1}{K_v\|\mu\|^2} + \varrho \\ Q_i = 1 \end{gathered} \tag{3-47}$$

Then, $r_1, r_2, \tilde{A}_{xyz},$ *and* $\tilde{A}_{\phi\theta\psi}$ *are stable in the Lyapunov sense around zero. Moreover,* e_1*, and* e_2 *are guaranteed to be arbitrarily small with transient dynamics shaped by the choice of* $\Lambda_1, \Lambda_2, K_{r_1}, K_{r_2}, K_{i_{r_1}}$ *and* $K_{i_{r_2}}$.

Proof.

Consider the Lyapunov function candidate given by

$$\mathbb{L} = \frac{1}{2} r_1^T m r_1 + \frac{1}{2}\left[\int_0^t r_1^T dt'\right] K_{i_1}\left[\int_0^t r_1 dt'\right] + \frac{1}{2}\left[\int_0^t r_2^T dt'\right] K_{i_2}\left[\int_0^t r_2 dt'\right] + \frac{1}{2} r_2^T J r_2 + \frac{1}{2}\{\tilde{A}_{xyz}^T F^{-1} \tilde{A}_{xyz}\} + \frac{1}{2}\{\tilde{A}_{\phi\theta\psi}^T F^{-1} \tilde{A}_{\phi\theta\psi}\} \quad (3\text{-}48)$$

The time derivative along the solution of the dynamic system is given by:

$$\dot{\mathbb{L}} = r_1^T m \dot{r}_1 + r_2^T J \dot{r}_2 + \frac{1}{2} r_2^T \dot{J} r_2 + r_1^T K_{i_1}\left[\int_0^t r_1 dt'\right] + r_2^T K_{i_2}\left[\int_0^t r_2 dt'\right] + \left\{\tilde{A}^T F^{-1} \dot{\tilde{A}}\right\} \quad (3\text{-}49)$$

with $\tilde{A} = \left[\tilde{A}_{xyz}, \tilde{A}_{\phi\theta\psi}\right]^T$, and $F = diag(F_1, F_2)$. Recall that augmented error dynamic of the system is given by

$$m\,\dot{r}_1 = \tilde{A}_{xyz} - \left(K_{r_1} - m\Lambda_1\right) r_1 - K_{i_1} \int_0^t r_1 dt'$$

and

$$J\,\dot{r}_2 = \tilde{A}_{\phi\theta\psi} - \{K_{r_2} + C(\eta, \dot{\eta}) - J\Lambda_2\} r_2 - K_{i_2} \int_0^t r_2 dt'$$

Substitute Eq. 3-16, and Eq. 3-21 into Eq. 3-49, we get

$$\dot{\mathbb{L}} = r_1^T\left\{\tilde{A}_{xyz} - (K_{r_1} - m\Lambda_1)r_1 - K_{i_1}\int_0^t r_1 dt'\right\} + \frac{1}{2}r_2^T\dot{J}r_2$$

$$+ r_2^T\left\{\tilde{A}_{\phi\theta\psi} - \{K_{r_2} + C(\eta,\dot{\eta}) - J\Lambda_2\}\,r_2 - K_{i_2}\int_0^t r_2 dt'\right\} \qquad (3\text{-}50)$$

$$+ r_1^T K_{i_1}\left[\int_0^t r_1 dt'\right] + r_2^T K_{i_2}\left[\int_0^t r_2 dt'\right] + \left\{\tilde{A}^T F^{-1}\dot{\tilde{A}}\right\}$$

The following steps show straightforward mathematical simplifications for Eq. 3-50; first, integral terms cancel out

$$\dot{\mathbb{L}} = r_1^T\{\tilde{A}_{xyz} - (K_{r_1} - m\Lambda_1)r_1\} + r_2^T\{\tilde{A}_{\phi\theta\psi} - \{K_{r_2} + C(\eta,\dot{\eta}) - J\Lambda_2\}\,r_2\} + \frac{1}{2}r_2^T\dot{J}r_2$$

$$+ \left\{\tilde{A}^T F^{-1}\dot{\tilde{A}}\right\}$$

Then, similar terms are collected and organized in a matrix form, as follows

$$\dot{\mathbb{L}} = \{-r_1^T(K_{r_1} - m\Lambda_1)r_1 + r_1^T\tilde{A}_{xyz}\} + \frac{1}{2}r_2^T\dot{J}r_2$$

$$+ \{r_2^T\tilde{A}_{\phi\theta\psi} - r_2^T(K_{r_2} - J\Lambda_2)\,r_2 - r_2^T C(\eta,\dot{\eta})\,r_2\} + \left\{\tilde{A}^T F^{-1}\dot{\tilde{A}}\right\}$$

$$\dot{\mathbb{L}} = \{-r_1^T(K_{r_1} - m\Lambda_1)r_1 - r_2^T(K_{r_2} - J\Lambda_2)\,r_2\} + \frac{1}{2}r_2^T\dot{J}r_2 - r_2^T C(\eta,\dot{\eta})\,r_2 + r_2^T\tilde{A}_{\phi\theta\psi}^T + r_1^T\tilde{A}_{xyz}^T$$

$$+ \left\{\tilde{A}^T F^{-1}\dot{\tilde{A}}\right\}$$

$$\dot{\mathbb{L}} = -r^T K_v r + \frac{1}{2}r^T\dot{J}_{C_v}r + r_2^T\tilde{A}_{\phi\theta\psi}^T + r_1^T\tilde{A}_{xyz}^T + \left\{\tilde{A}^T F^{-1}\dot{\tilde{A}}\right\}$$

The derivative of the Lyapunov function candidate is then given by

$$\dot{\mathbb{L}} = -r^T K_v r + \frac{1}{2} r^T J_{C_v} r + \left\{ \tilde{A}^T F^{-1} \dot{\tilde{A}} + \tilde{A}^T_{\phi\theta\psi}\, r_2^T + \tilde{A}^T_{xyz}\, r_1^T \right\} \tag{3-51}$$

where

$$K_v = \begin{bmatrix} K_{r_1} - m\Lambda_1 & 0 \\ 0 & K_{r_2} - J\Lambda_2 \end{bmatrix}, J_{C_v} = \begin{bmatrix} 0 & 0 \\ 0 & \frac{1}{2}\left(\dot{J} - 2C(\eta,\dot{\eta})\right) \end{bmatrix}$$

$$r = [r_1, r_2]^T$$

It is shown in [55] that with the choice of $C(\eta,\dot{\eta})$ in (5), $\left(\frac{d}{dt}J(\eta) - 2C(\eta,\dot{\eta})\right)$ is skew-symmetric. Accordingly, the second term in Eq. 3-51 is eliminated. The time derivative of the ANN estimation error is calculated as follows

$$\dot{\tilde{A}} = \left(\frac{d\tilde{A}}{d\widehat{W}}\right)^T \frac{d\widehat{W}}{dt}$$

$$\frac{d\widehat{W}}{dt} \triangleq -F\frac{\partial \mathcal{J}}{\partial \widehat{W}} = -F\left\{\Upsilon \frac{d\Upsilon}{d\widehat{W}} + \kappa\widehat{W}\right\}$$

$$\dot{\tilde{A}} = -F\left(\frac{d\tilde{A}}{d\widehat{W}}\right)^T \Upsilon \frac{d\Upsilon}{d\widehat{W}} - F\left(\frac{d\tilde{A}}{d\widehat{W}}\right)^T \kappa\widehat{W}$$

$$\dot{\tilde{A}} = -F(-\mu^T)\Upsilon(-\mu) - F(-\mu^T)\kappa\widehat{W}$$

$$\dot{\tilde{A}} = -F(\mu^T)\Upsilon(\mu) + F(\mu^T)\kappa\widehat{W} \tag{3-52}$$

where $\widehat{W} = \left[\widehat{W}_1, \widehat{W}_2\right]^T$, $\mu = diag(\mu_1, \mu_2)$, and $\Upsilon = [\Upsilon_1, \Upsilon_2]^T$.

The learning sliding mode, Υ, has a direct relationship with aerodynamics uncertainty estimation error, $\tilde{A}$, which is produced by the substitution of *dynamics sliding* mode Eq. 3-16 and Eq. 3-27 into, Υ

$$\Upsilon = \tilde{A} + (Q_r - \varrho)K_r r + (Q_i - 1)K_i \int_0^t r dt' \tag{3-53}$$

Here, $K_r = diag(K_{r_1}, K_{r_2})$, $K_i = diag(K_{i_1}, K_{i_2})$ $Q_r = diag(Q_{r_1}, Q_{r_2})$, and $Q_i = diag(Q_{i_1}, Q_{i_2})$. With $0 < \varrho < 1$, derived from the stability requirement Eq. 3-22. Substituting Eq. 3-53 into Eq. 3-52, we get

$$\dot{\tilde{A}} = -F\mu^T\mu \left[\tilde{A} + (Q_r - \varrho)K_r r + (Q_i - 1)K_i \int_0^t r dt'\right] + F\mu\kappa\widehat{W} \tag{3-54}$$

and Accordingly,

$$\left\{\tilde{A}^T F^{-1}\dot{\tilde{A}}\right\} = \tilde{A}^T \left\{-\mu^T\mu \left[\tilde{A} + (Q_r - \varrho)K_r r + (Q_i - 1)K_i \int_0^t r dt'\right] + \mu\kappa\widehat{W}\right\}$$

$$\left\{\tilde{A}^T F^{-1}\dot{\tilde{A}}\right\} = \left\{-\mu^T\mu \left[\tilde{A}^T\tilde{A} + \tilde{A}^T(Q_r - \varrho)K_r r_1 + \tilde{A}^T(Q_i - 1)K_i \int_0^t r dt'\right] + \tilde{A}^T\mu\kappa\widehat{W}\right\}$$

$$\left\{\tilde{A}^T F^{-1}\dot{\tilde{A}}\right\} = \left\{-\mu^T\mu\tilde{A}^T\tilde{A} - \mu^T\mu\tilde{A}^T(Q_i - 1)K_i \int_0^t r dt' + \tilde{A}^T\mu\kappa\widehat{W}\right\} - \mu^T\mu\tilde{A}^T(Q_r - \varrho)K_r r \tag{3-55}$$

Now, substituting Eq. 3-55 into the time derivative of the Lyapunov function candidate, Eq. 3-51. We get,

$$\dot{\mathbb{L}} = -r^T K_v r + \frac{1}{2} r^T J_{C_v} r + \left\{ \left\{ -\mu^T \mu \tilde{A}^T \tilde{A} - \mu^T \mu \tilde{A}^T (Q_i - 1) K_i \int_0^t r dt' + \tilde{A}^T \mu \kappa \widehat{W} \right\} - \mu^T \mu \tilde{A}^T (Q_r - \varrho) K_r r + \tilde{A}^T_{\phi\theta\psi} r_2^T + \tilde{A}^T_{xyz} r_1^T \right\} \tag{3-56}$$

The last three terms in right hand side of Eq. 3-56 are dealt-with as follows

$$-\mu^T \mu \tilde{A}^T (Q_r - \varrho) K_r r + r^T \tilde{A}$$

with $r^T \tilde{A} = [r_1^T, r_2^T]\left[\tilde{A}_\xi, \tilde{A}_\eta\right]^T$. Recall the upper limit on the norm of r given by

$$\|r\| \le \frac{\|\tilde{A}\|}{K_v}$$

and, the definition of Q_r

$$Q_r > 1$$

Then,

$$-\left\|\mu^T \mu \tilde{A}^T (Q_r - \varrho) K_r r - r^T \tilde{A}\right\| \le -\left\|\tilde{A}\right\|^2 \left(\frac{\mu^T \mu (Q_r - \varrho)}{\varrho} - \frac{1}{\varrho K_r} \right) \le -\rho \left\|\tilde{A}\right\|^2 \tag{3-57}$$

with the choice of $Q_i = 1$, the integral term in Eq. 3-56 vanishes. And the fifth term in Eq. 3-56 has the following inequality apply

$$\left\|\hat{A}\right\| \le \frac{\|\mu^T \mu \Upsilon\|}{\kappa} \tag{3-58}$$

Which, with the limit on the magnitude of r, implies

$$\|\hat{A}\| \leq \frac{\|\mu^T \mu \tilde{\mathrm{A}}\|}{\kappa} \tag{3-59}$$

Accordingly, the time derivative of the Lyapunov candidate function is given by

$$\dot{\mathbb{L}} \leq -K_v \|r\|^2 - \rho \|\tilde{A}\|^2 \tag{3-60}$$

Which is strictly negative semidefinite, with $\rho > 0$. ■

Based on the Lyapunov theorem (2-1), the closed loop system is asymptotically stable in the lyapunov sense around the state space origin, when the design parameters are selected according to Eq. 3-47.

CHAPTER 4

ANN CONTROLLER IMPLEMENTATION

This chapter presents the details of the practical implementation of the developed ANN controller. First, the experiment apparatus is described, and the required setup is presented. Second, the method followed for trajectory measurement is discussed. Afterwards, numerical simulation results for the ANN controller when used for Qball trajectory control are provided. Several unknown aerodynamic and model uncertainties were applied to the numerical simulation to test for the effectiveness of the proposed ANN structure with the developed learning algorithm, while stability conditions are satisfied. Finally, real time experimental results are shown for the Qball trajectory control with several different environment setups, when using the predesigned PID controller, and the developed ANN controller.

4.1 QUANSER'S QBALL-X4

Quanser is Canadian company founded in 1989, which specializes on the creation of real-time platforms for education and research. Their products are supported by MatLab ® and Simulink ® with the required addition of Quanser's QUARC ® software.

The Quanser Qball-X4, shown in Figure 4.1, is an UAV quadrotor designed for indoors laboratory setup. It represents a flexible, open-architecture platform that can be used to develop and apply controllers and control algorithms. A fiber carbon protective cage is built around the Qball, to protect its components and people from collision damage. To measure onboard sensors and drive thrust motors, the Qball-X4 utilizes Quanser's onboard avionics data acquisition card (DAQ), the HiQ, and the embedded Gumstix computer. The HiQ DAQ is a high-resolution Inertial Measurement Unit (IMU) and avionics Input/Output (I/O) card designed to accommodate a wide variety of research applications e.g. attitude estimation and trajectory control. The high precision aero sensory board, HiQ, is equipped with 3-axis accelerometer, 3-axis gyroscope, 3-axis magnetometer, 2 barometers, an altitude SONAR sensor, and additional input ports for extra inputs. Table 4-1 summarizes the technical specifications of the Qball-X4 quadrotor [82].

QUARC ®, Quanser's real-time control software, allows to rapidly develop and test controllers on actual hardware through a MATLAB/Simulink interface. QUARC's open-architecture hardware and the extensive Simulink block set provide a powerful controls development tool. QUARC can target the Gumstix embedded computer, automatically generate the code, and execute controllers onboard the vehicle. During flight, while the controller is executing on the Gumstix, code parameters could be tuned, and sensor measurements could be observed in real-time from a host ground station computer (PC or laptop) [82]. It is worth mentioning that Simulink offers the same capability in the Simulink Real-Time toolbox.

The interface to the Qball-X4 is MATLAB Simulink with QUARC. The controllers are developed in Simulink with QUARC on the host computer, and these models are downloaded and compiled into executables on the target, the Gumstix.

For more details about the Quanser Qball-X4, the reader is advised to check [82].

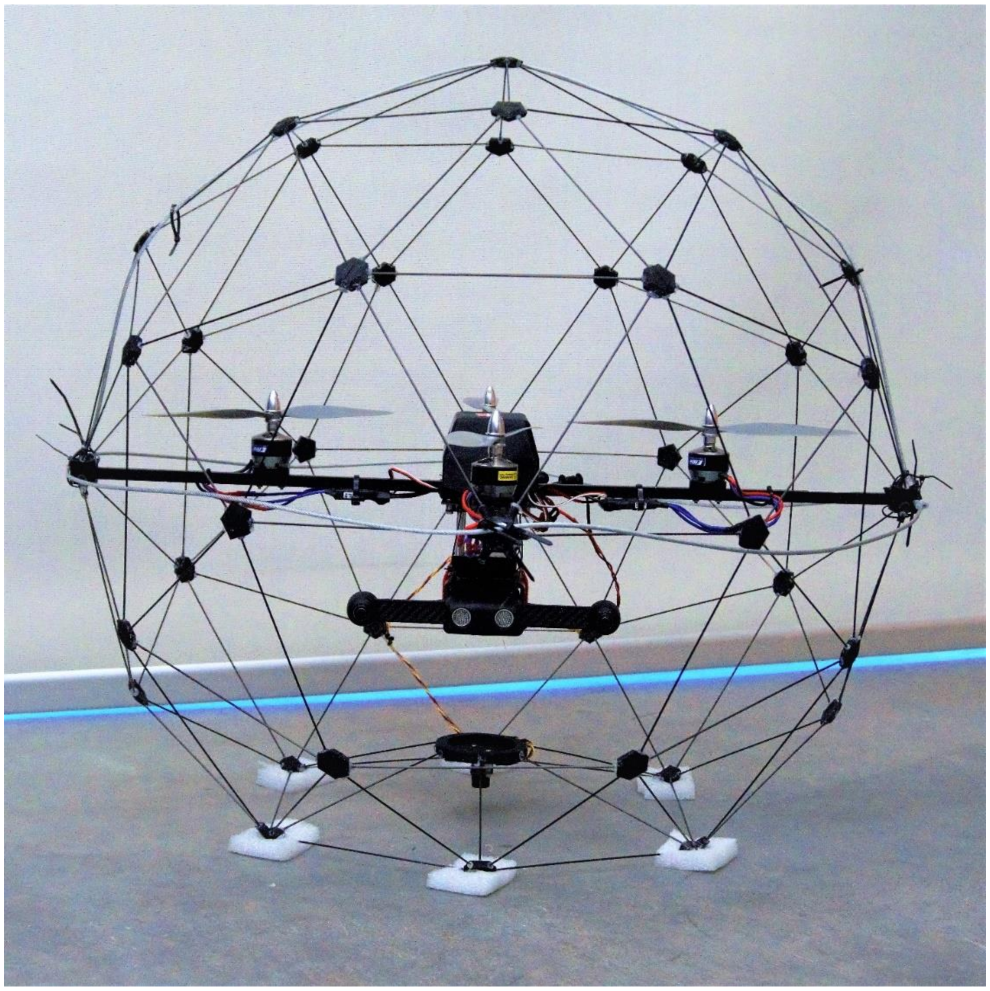


Figure 4.1 Qball-X4

In this research, the hardware and the software of the Qball are significantly modified. To replace Quanser's expensive off-the-shelf OptiTrack localization system, a Simultaneous Localization and Mapping (SLAM) sensor is integrated with the Qball to constantly measure its location in an indoors setup. This modification makes Qball completely inclusive, so, it could be operated anywhere in the presence of the

Host computer with the running modules, without the need to setup the tracking system every time the environment changes. In addition, this modification allows to extend Qball's flying zone, which is usually confined by the OptiTrack's cameras field of view. The software provided by Quanser is modified to support the hardware modification.

Table 4-1 Qball-X4 Features

Parameter	**Description**
Power	2 LiPo rechargeable batteries, 3000 mAh, 3-Cell (upgraded from original)
On-board Computer	Gumstix Verdex, with integrated 802.11 b/g/n WiFi
Outputs	10 PWM (servo motor outputs)
gyroscope	3-axis reconfigurable range for $\pm 75°/s$, $\pm 150°/s$, or $\pm 300°/s$. With a resolution of $0.0125°/s/LSB$ at the used range of $\pm 75°/s$
Accelerometer	3-axis accelerometer, resolution 3.33 mg/LSB
Pressure	2 pressure sensors, absolute and relative pressure
Sonar Input	4 Maxbotix sonar inputs
Input Power	10-20 V
Extra I/O	11 reconfigurable digital I/O 2 TTL serial ports Serial GPS input
Propellers	4 of APC 10x4.7
Motors	4 of E-Flite Park 480 (1020 Kv)

Table 4-2 Qball-X4 Properties

Parameter	Value	Unit
J_{xx}	0.03	$kg.m^2$
J_{yy}	0.03	$kg.m^2$
J_{zz}	0.04	$kg.m^2$
m	1.4	kg
l	0.2	m
F	$K_f \frac{w}{s+w}$	N/PWM
w	15	rad/sec
$Ext.Dia.$	0.74	m
K_F	120	N/PWM
K_y	4	$N.m/PWM$

4.1.2 Qball's Software Module

Two Simulink models are required to operate the Qball, namely: Host_Jostick_TYPE_B.mdl, and qball_x4_control_v4_Ahmed_Sam.mdl. The former runs on the host computer and communicates Joystick measurement to the Qball, while the latter acts as the main control module for the quadrotor and is executed on the Qball's onboard computer.

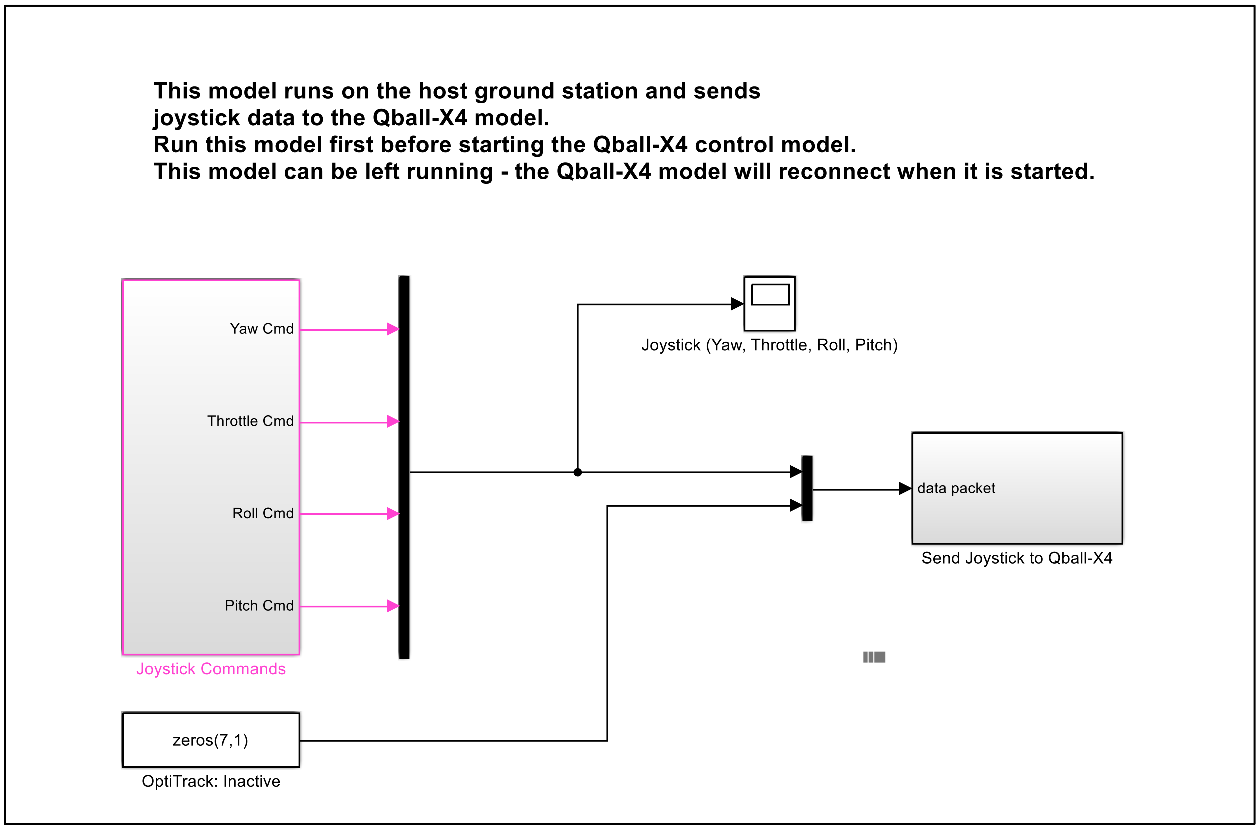
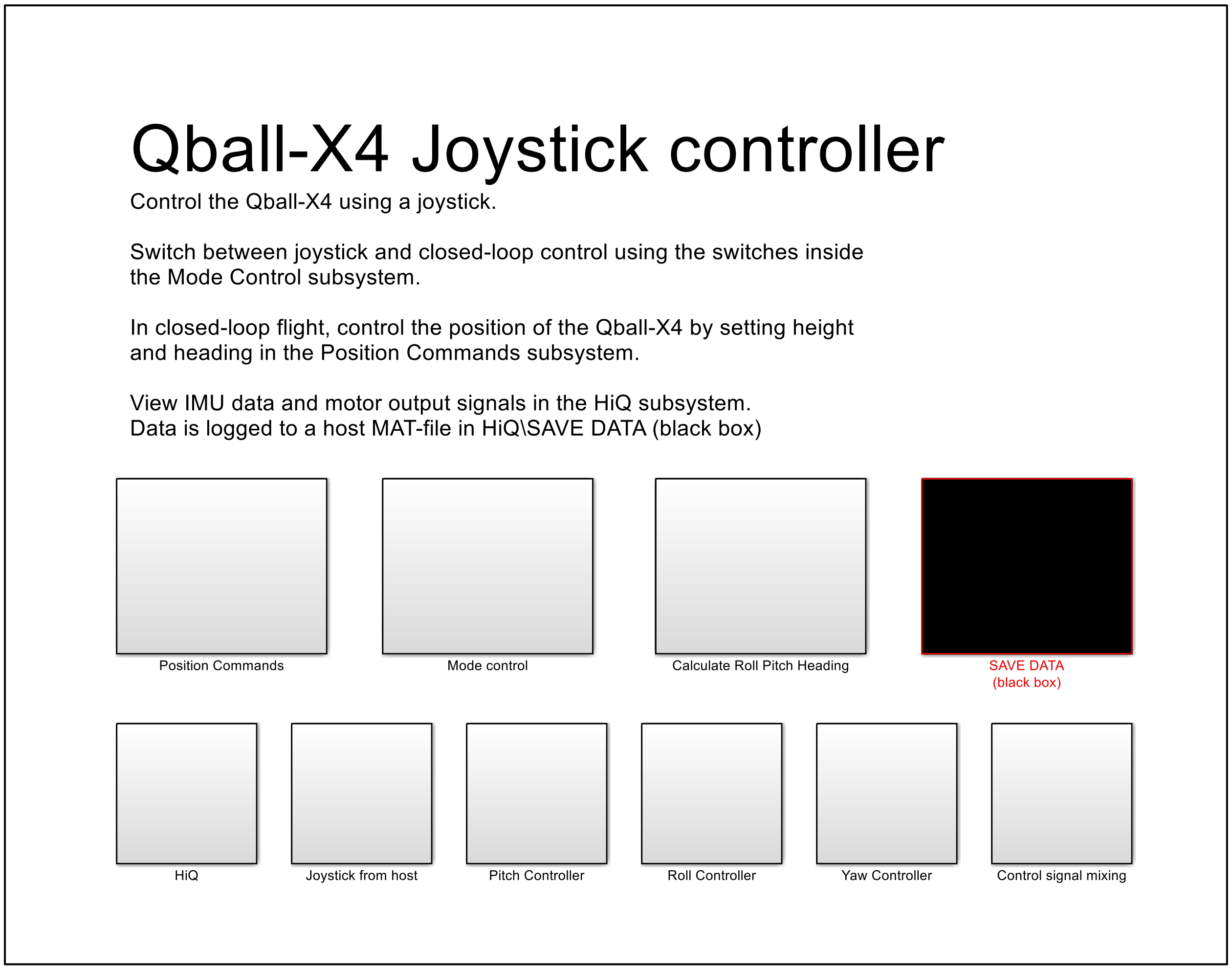

This model runs on the host ground station and sends
joystick data to the Qball-X4 model.
Run this model first before starting the Qball-X4 control model.
This model can be left running - the Qball-X4 model will reconnect when it is started.
Yaw Cmd
Throttle Cmd
Roll Cmd
Pitch Cmd
Joystick Commands
Joystick (Yaw, Throttle, Roll, Pitch)
data packet
Send Joystick to Qball-X4
zeros(7,1)
OptiTrack: Inactive

Figure 4.2 Host Joystick Module

Figure 4.2 shows the Simulink model for the Host Joystick Module. The *Joystick Commands* block acquires commands from the serially connected joystick to the host computer, while the *Send Joystick to Qball-X4* sends joystick commands to the embedded computer on Qball via WiFi. The *OptiTrack* block is not active in the original set up, as this feature was not purchased with the platform.


Qball-X4 Joystick controller

Control the Qball-X4 using a joystick.

Switch between joystick and closed-loop control using the switches inside the Mode Control subsystem.

In closed-loop flight, control the position of the Qball-X4 by setting height and heading in the Position Commands subsystem.

View IMU data and motor output signals in the HiQ subsystem.
Data is logged to a host MAT-file in HiQ\SAVE DATA (black box)

Figure 4.3 Qball-X4 Controller Module

The Qball-X4 controller Module, shown in Figure 4.3, on the other hand, initializes and configures the onboard aero sensory board through the *HiQ* block. The *Joystick from host* block receives joystick inputs and position measurements (when available). *Pitch Controller*, *Roll Controller* and *Yaw Controller* blocks apply an LQ designed PID controller for the respective attitude dynamics. The *Calculate Pitch Roll Heading* block is responsible for calculating attitude angles using IMU measurements with the aid of complimentary filters. The measurements from

Accelerometer and Gyroscope are used for estimating pitch and roll angles, while magnetometer is used for calculating heading angle. The *Control Mode* block is used to switch between open-loop control (via the joystick), and closed-loop control when trajectory measurement is available. A switcher is dedicated to each trajectory degree-of-freedom; namely X, Y, Z, and ψ. Therefore, the Qball could be operated with a closed-loop set up for one degree-of-freedom, e.g. height, while the rest are set to open-loop control, and vice versa.

The *Position Command* block provides trajectory commands when the closed-loop set up is selected. This block also contains the altitude PID controller, which uses the sonar sensor, mounted at the bottom of the cage, for height measurement. The X, Y, and ψ controllers are embedded inside the Pitch controller, Roll controller, and Yaw controller blocks respectively.

The *Control signal Mixing* block combines PWM commands from the different degrees of freedom to send four signals to each motor.

Finally, the *Save Data* block saves all selected measurements to a data file on MatLab's directory.

4.1.3 Qball PID Controller

Figures 4.4, 4.5, 4.6 show the Qball controller designed by Quanser. Attitude Proportional Integral Derivative (PID) controllers are designed using Linear Quadratic (LQ) method with the experimentally realized Qball model [82]. The gains for the X, and Y Proportional Derivative (PD) controllers are experimentally designed. The altitude and the heading are controlled with experimentally designed PID controllers as well. Table 4-3 shows the gain values for Quanser's controller parameters.

Table 4-3 Qball PID Controller Parameters

Parameter	Value
Attitude Controller (roll and pitch)	
K_p	0.062
K_d	0.013
K_I	0
K_v	1.107
Heading Controller	
K_p	0.0316
K_d	0.015
Position Controller	
K_p	0.67
K_d	0.36
Altitude Controller	
K_p	0.0062
K_d	0.0078
K_I	0.0021

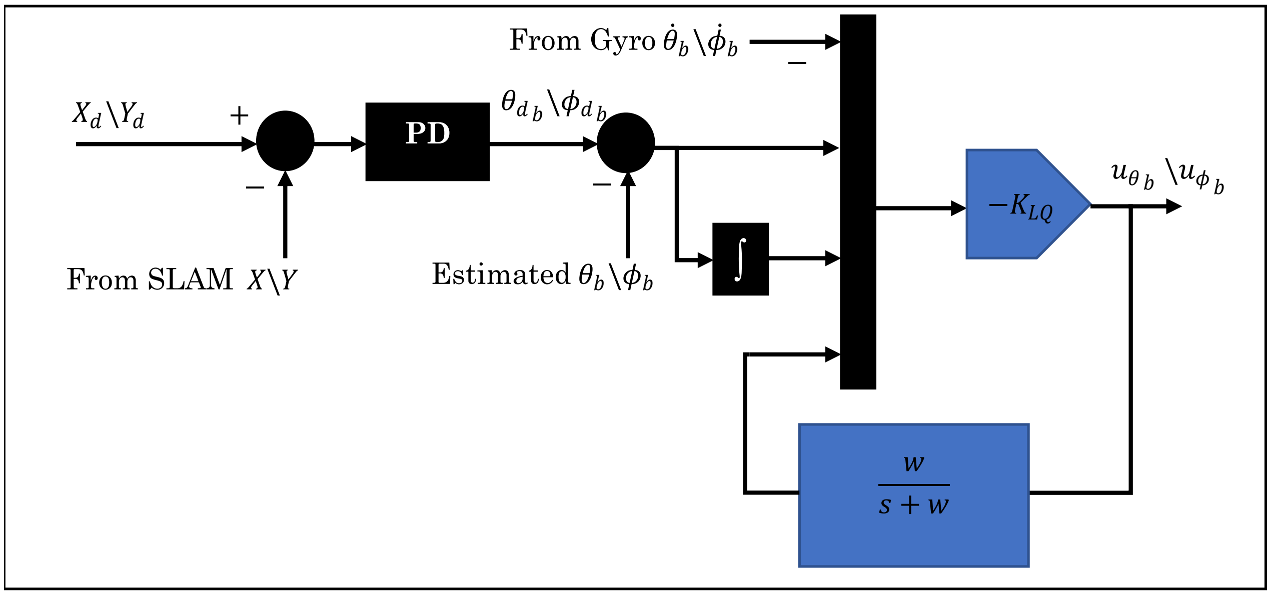

From Gyro $\dot{\theta}_b\backslash\dot{\phi}_b$
$-$
$X_d\backslash Y_d$
$+$
$-$
PD
$\theta_{d\,b}\backslash\phi_{d\,b}$
$-$
From SLAM $X\backslash Y$
Estimated $\theta_b\backslash\phi_b$
$\int$
$-K_{LQ}$
$u_{\theta\,b}\backslash u_{\phi\,b}$
$\frac{w}{s+w}$

Figure 4.4 Quanser's Trajectory and Attitude Controllers

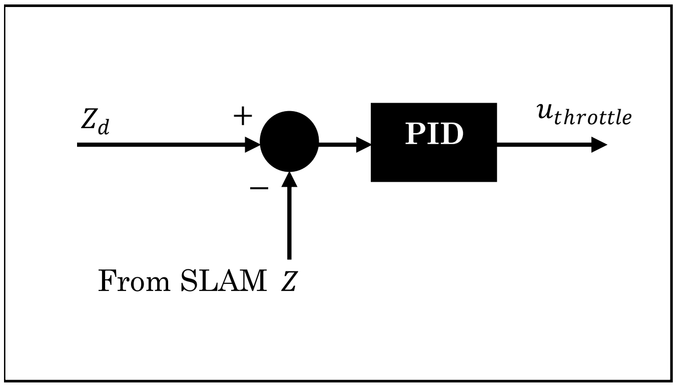


Figure 4.5 Quanser's Altitude Controller

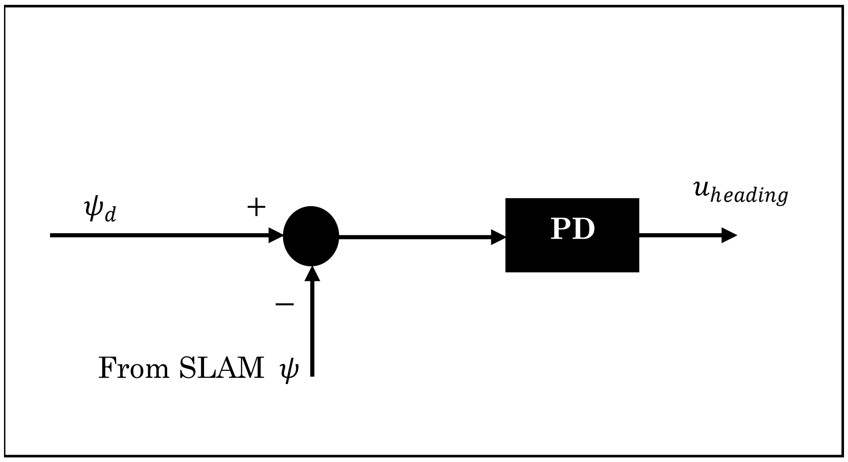


Figure 4.6 Quanser's Heading Controller

4.2 TRAJECTORY MEASUREMENT

Mobile robots position measurement is an active research area [83] [84]. Solutions for more precision, reliability, and efficiency are sought to control a wide variety of mobile robots for an even wider spectrum of applications. Sensors including Light Distance and Ranging (LIDAR), Sound Navigation and Ranging (SONAR), Inertial Measurement Units (IMUs), wheel encoders, Radio Distance and Ranging (RADAR), Infrared Ranging (IR), Global Positioning Systems (GPS), Optical Flow, Ultra-Wideband Range Localization, Optical Tracking, and Simultaneous Localization and Mapping (SLAM), and combination of many sensors together have been used in mobile robot systems for better localization.

In this research, SLAM is used for trajectory measurement. SLAM is a fundamental solution for mobile robot's navigation that allows for localization using primarily vision. Its main purpose is to allow mobile robots to incrementally build a consistent map for the environment while simultaneously determining their location within this map, when randomly placed at an unknown environment and unknown location [85]. The theoretical and conceptual level of SLAM is considered a solved problem [85]. However, sustainable issues remain in the practical realization of the algorithm.

SLAM has been widely implemented in different domains including outdoor, indoor, underwater, and airborne robot systems. There are many developed SLAM algorithms with open source codes available for research and development [85] [86].

Vision based sensors are found more accurate and reliable than GPS in outdoor domains, and, more importantly, they provide a good solution for GPS denied domains. The major advantage of SLAM is that it increases the level of independency of mobile robots on external hardware, and allows to build completely autonomous systems, in contrast to other methods. However, there are some practical issue that researchers are still working on to increase the robustness of SLAM. It is important to mention that the application of UAV imposes tougher requirements for trajectory measurement, as UAVs are inherently unstable and contain very rapid dynamics, accordingly, localization has to be very fast and accurate to avoid eminent damage.

In this research, the Parrot S.L.A.M Dunk developer's kit, shown in Figure 4.7, is used to measure the Qball trajectory. Parrot is a French manufacturing company, specialized in technologies involving voice recognition and signal processing for embedded products as well as remotely controlled drones [87].

The SLAM Dunk developer's kit is an integrated hardware and software for advanced navigation applications, such as drones and robotics. The Parrot SLAM Dunk allows to access integrated sensors optimized to delivering synchronized data through the standard Robotics Operating System (ROS) framework. It utilizes two wide angle stereo cameras to generate a depth map for the environment, in real-time, and integrates that with the measurement of IMU embedded sensors, using a SLAM algorithm to compute the position in space in real-time, without GPS, and to map the environment in three dimensions in point cloud [88].

SLAM Dunk comes with an onboard computer with NVIDIA Tagra K1 microprocessor that runs on Ubuntu with ROS compatible Software Development Kit (SDK). This allows for the development of complex algorithms within the SLAM Dunk, without the need for extra computer.

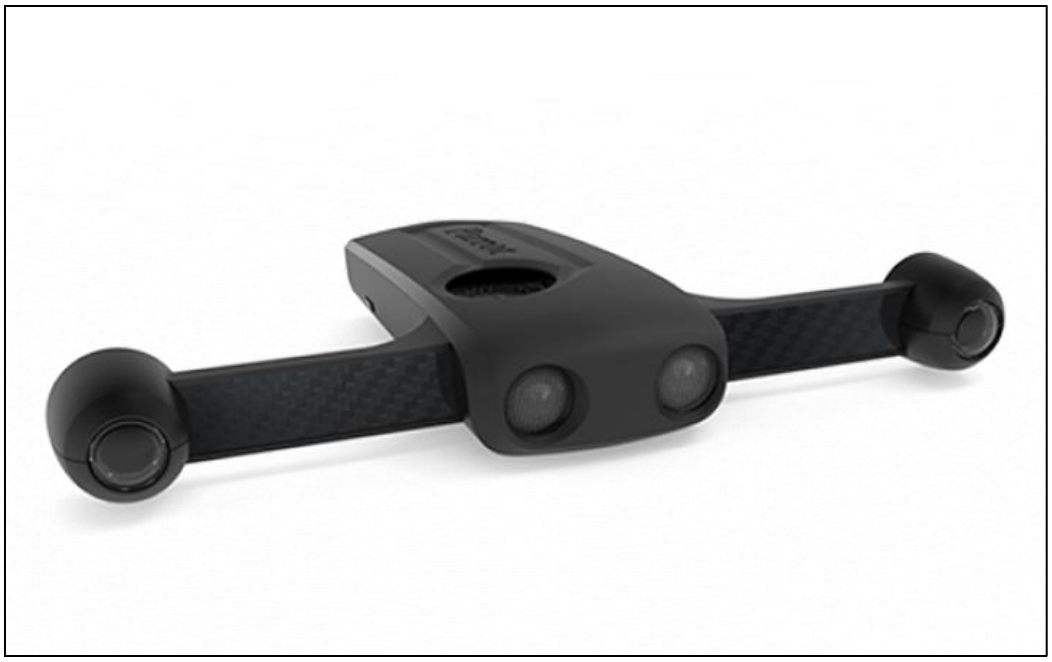


Figure 4.7 Parrot S.L.A.M Dunk

Communication with SLAM Dunk could be established through serial or WiFi connection, as a ROS node.

Due to the limited accessibility to the Qball onboard computer, the serial communication was not possible. Accordingly, the Host computer is connected to the SLAM Dunk through WiFi, and Simulink's ROS toolbox is used to create a 'subscribing node' to acquire position data. This Simulink/ROS model is integrated with the Qball Joystick module, so as to send the data one-time from the Host computer to the target computer. Figure 4.8 shows the modified Qball Joystick SLAM dunk model: Host_Joystick_TYPE_B_AhmedSlam_Dunk_Rate_Sync. In doing this merge, the following challenges had to be addressed:

- Position information is sent through WiFi communication, which introduces the problem of lost data packets, and accordingly leads to performance glitch. This issue is managed by optimizing the WiFi connection.
- For the two models to work together, namely SLAM Dunk and Joystick models, the 'normal' simulation mode was necessary. Which introduces the problem of execution-time's synchrony with real-time; as the simulation computational-time is completely different than real-time, usually faster depending on the complexity of the Simulink code and the computer computational capability. For this reason, a *Time Pacer* block is used in the model to synchronize the simulation-time with real-time.
- The sampling rate for the SLAM algorithm is set at 60 Hz, which is much slower than the Joystick's model sampling frequency of 100 Hz. This difference in sampling rate is resolved by using Simulink's '*enable*' block which allows to hold the previous value until it is updated at the 60 Hz frequency.

SLAM Dunk sensor's power supply is directly connected to the power distribution board of the Qball. This adds more power requirement to the Qball batteries, accordingly, Qball's batteries were upgraded to 3000 mAh from 2700 mAh. Notice that the battery upgrade adds more payload to the Qball in addition to what the SLAM Dunk sensor and its holder add, which exceeds the limit for the Qball payload. Accordingly, the run time is significantly reduced to about 5 minutes, from the original 10 minutes. However, no actuator saturation is observed, which could have led to results invalidation. A holder is 3D printed to mount the SLAM Dunk sensor right below the batteries compartment.

Figures 4.8 and 4.9 show the modified joystick module and the SLAM dunk measurement module respectively. The *SLAM dunk Position Measurement* block is added to the original joystick module to connect with Parrot's SLAM dunk and acquire trajectory measurements. This block creates a Subscribing "ROS" node that communicates between the host computer and the Parrot SLAM dunk through WiFi. The *Topic* subscribed for is *Pose,* which provides Parrot's SALM Dunk inertial 3D position, and inertial attitude angles about the inertial coordinate system. X, Y, Z, and ψ are the only measurements used for the Qball control. The remaining attitude measurements are acquired at a higher sampling rate and higher precision from the onboard IMU. It is important to note that the SLAM algorithm runs at 60 Hz sampling rate, while the Qball controller runs at 200 Hz on the onboard computer.

The *Set Parameter* block is used to set the *Video Mode* for the Parrot sensor to 60 fps, as it is defaulted at 30 fps. This set up is chosen for a quicker position sensitivity. The *Rate Sync* block synchronizes simulation's run-time with Real-Time, as these two times are different when Simulink runs on *normal* mode, as previously mentioned.

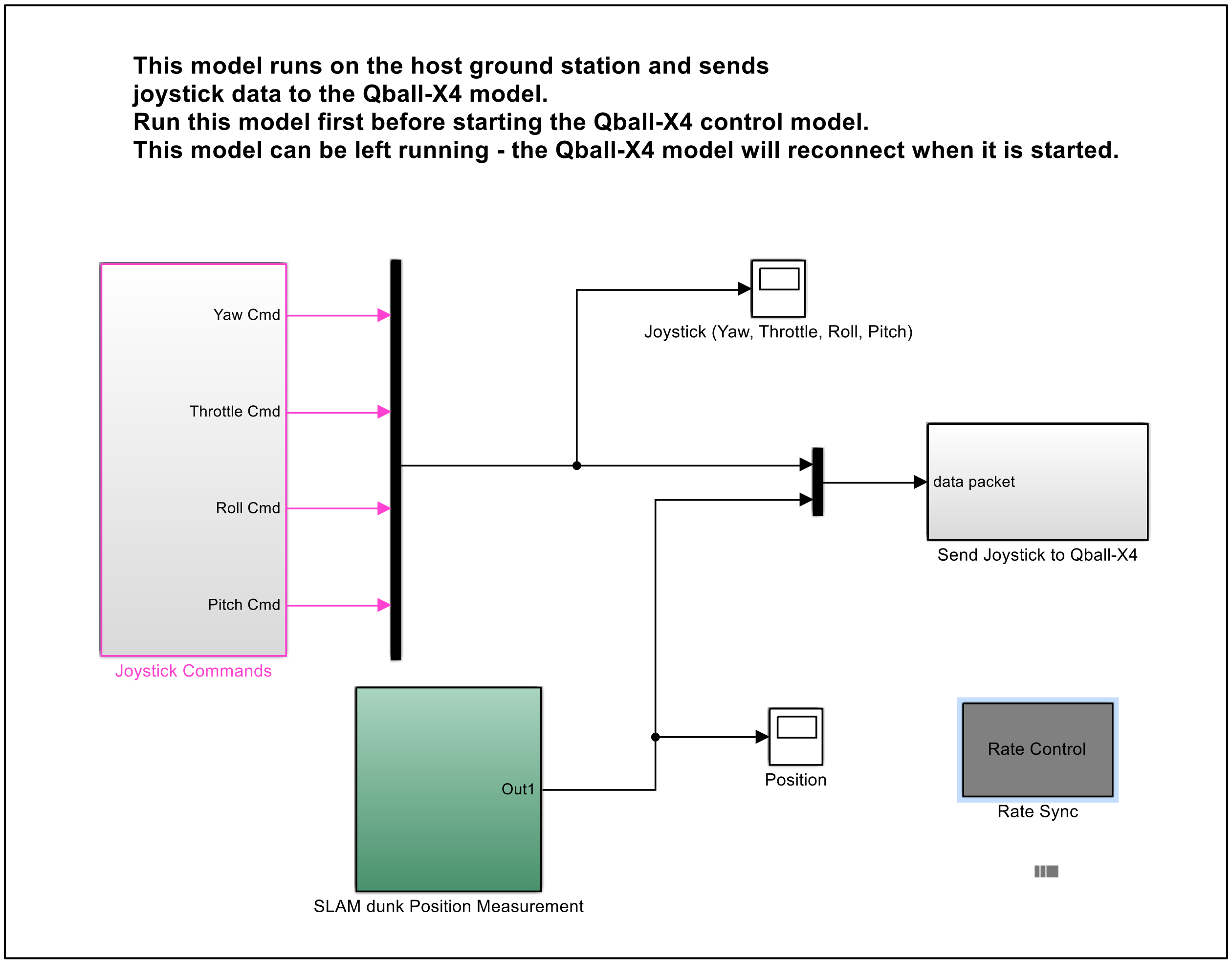

This model runs on the host ground station and sends joystick data to the Qball-X4 model.
Run this model first before starting the Qball-X4 control model.
This model can be left running - the Qball-X4 model will reconnect when it is started.
Yaw Cmd
Throttle Cmd
Roll Cmd
Pitch Cmd
Joystick Commands
Joystick (Yaw, Throttle, Roll, Pitch)
data packet
Send Joystick to Qball-X4
Out1
SLAM dunk Position Measurement
Position
Rate Control
Rate Sync

Figure 4.8 Modified Joystick Module

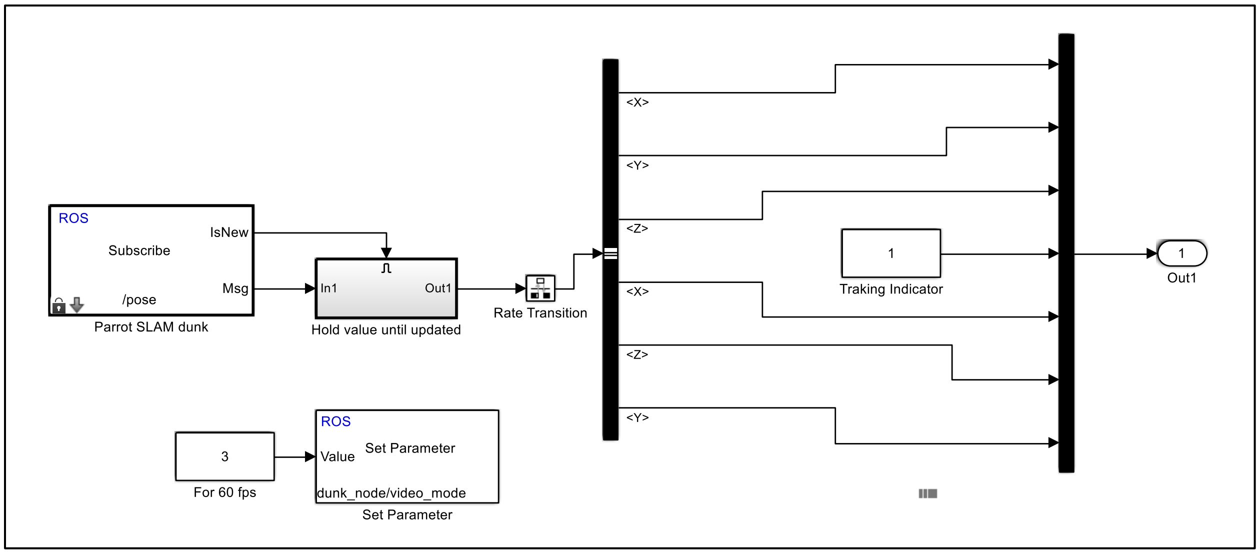


Figure 4.9 SLAM Dunk Position Measurement Submodule

The coordinate system used for the OptiTrack tracking, shown in Figure 4.10 (a), is different than that of the Qball, shown in Figure 4.10 (b). Accordingly, the Qball-X4 controller module is modified to match tracking coordinates with Qball's system.

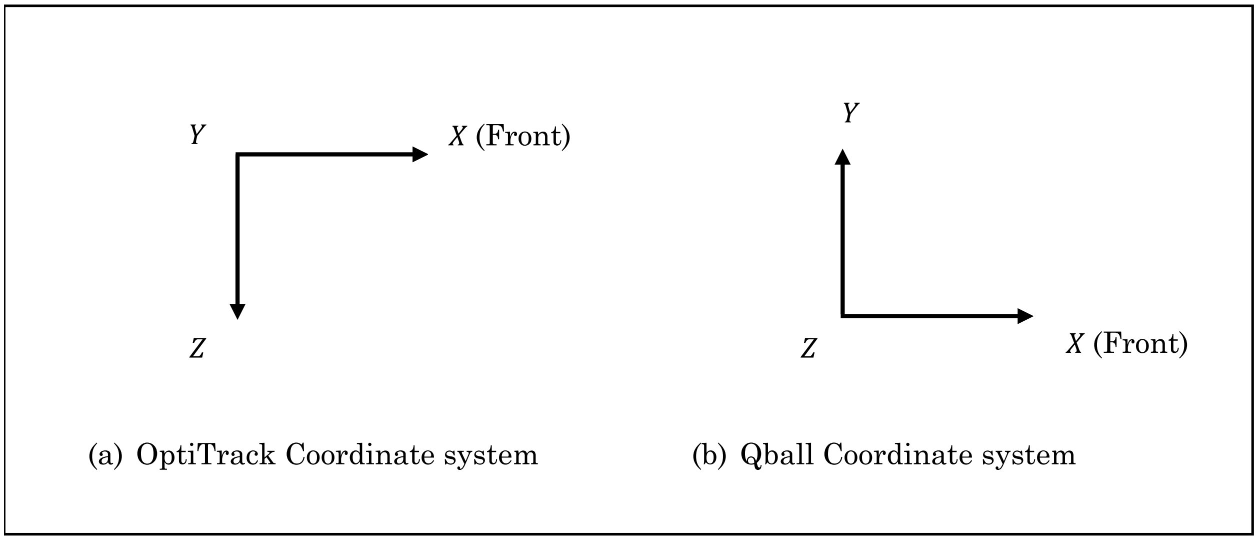


Figure 4.10 OptiTrack vs Qball Coordinate System

4.3 QBALL ANN CONTROLLER MODULE

The developed Qball ANN controller module is shown in Figure 4.11. The three trajectory controller blocks and the three attitude controller blocks are replaced with *Mekky's ANN Controller*. Unlike the original controller, quadrotors states are vectored, making ζ for trajectory states and η for attitude states, then, the ANN controller is implemented in two loops; an outer trajectory loop, and an inner attitude loop, as depicted in Figure 3.3.

The ANN controller design parameters for the experiments are chosen as: $\Lambda_1 = diag(0.72, 0.72, 0.4615)$, $K_{r_1} = diag(4.7174, 4.7174, 2.4)$, $K_{i_1} = diag(1.4, 1.4, 1.68)$, $\Lambda_2 = diag(5.2025, 5.2025, 2.1070)$, $K_{r_2} = diag(0.624, 0.624, 0.24)$, and $K_{i_2} = diag(0.0083, 0.0083, 0.1)$. ANN parameters are selected as: $Q_{r_1} = Q_{r_2} = 1.2$, $Q_{i_1} = Q_{i_2} = 1$, $F_1 = diag(0.063 ,0.063 ,0.036)$, $F_2 = diag(0.009,0.009,0.009)$ and $\kappa_1 = \kappa_2 = 0.02$. Notice that these values satisfy the stability conditions in Eq. 3-47.

As shown in chapter 3, attitude commands are corrected for the heading angle, then the designed controller for attitude angles calculated based on the inertial frame is used to control attitude angles based of the body frame.

The small angle assumption is used for the calculation of ϕ_d and θ_d, which is typical for the Qball, as it is designed for small attitude angles with limits at $\pm 0.08\, rad$, attitude commands on the body frame become:

$$\phi_{c_b} = \phi_{c_i} * \cos(\psi) - \theta_{c_i} * \sin(\psi) \tag{4-1}$$

And

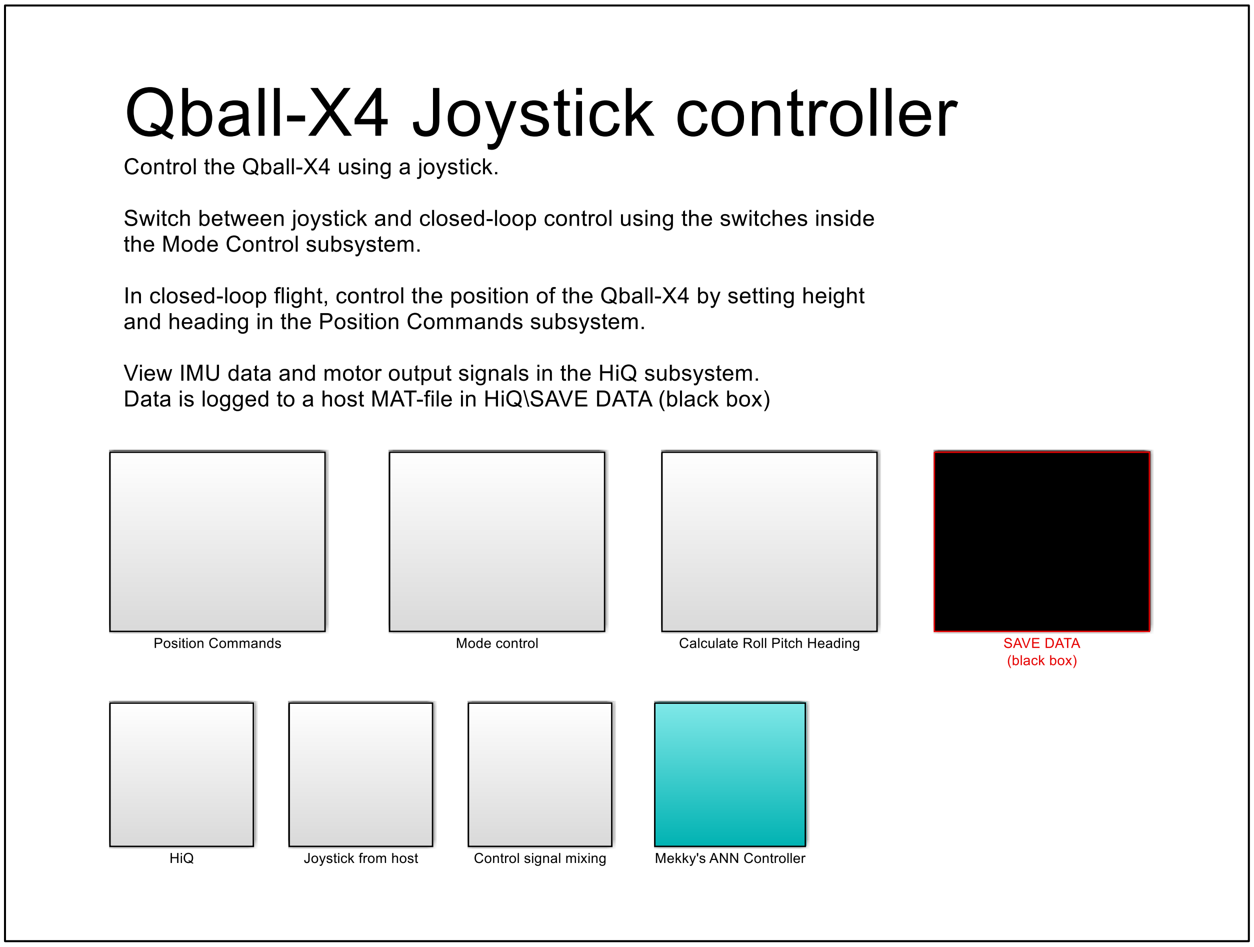
$$\theta_{c_b} = \theta_{c_i} * \cos(\psi) + \phi_{c_i} * \sin(\psi) \tag{4-2}$$

Small angle assumption is not applicable for the heading angle as it could vary for a wide range of values depending on commanded yaw angle ψ_c.

The challenges faced when implementing the ANN controller are:

- Computational capacity of the embedded computer: it was very challenging to get the ANN controller coded on the Qball's computer, as its processing power is limited. Accordingly, the coding of the controller had to be optimized for the least computational effort. Therefore, it was important to study alternative ways to code the same operation and chose the least computationally extensive method. On the other hand, the sampling rate had to be reduced to 80 Hz from the original 200 Hz on the PID controller. The reduction in the sampling rate limits the capability of the ANNs, as ANNs learn at each sample step, especially for the fast attitude dynamics. To achieve higher sampling rates, and accordingly, better performance, a more powerful onboard computer will be required.

- Motors saturation limit, and propellers thrust capacity: These factors lead to the limitation on the controller design parameters. And accordingly, smaller gain values were necessary, thus slower design dynamics could be achieved, in contrast to simulation results. To achieve high levels of performance, motor and propeller combination should be upgraded.
- Sensor noise: measurement noise was a very important issue to deal with. This could be the reason why online trained ANN controllers have not been experimentally successful. Here, Low pass filters were used when calculating derivative of state. The cut off frequencies of the filters had to be carefully chosen, such that, interference with actual dynamics is minimized. Better, more expensive, sensors could lead to better performance, as low pass filters introduce delays that interfere with performance. The tradeoff between noise interference and filter dynamics influence is crucial to control performance.



Qball-X4 Joystick controller

Control the Qball-X4 using a joystick.

Switch between joystick and closed-loop control using the switches inside the Mode Control subsystem.

In closed-loop flight, control the position of the Qball-X4 by setting height and heading in the Position Commands subsystem.

View IMU data and motor output signals in the HiQ subsystem.
Data is logged to a host MAT-file in HiQ\SAVE DATA (black box)

Figure 4.11 Qball ANN Controller Module

4.4 SIMULATION RESULTS

A nonlinear simulation is carried out for the proposed controller when applied to the Qball-X4 quadrotor. This simulation is developed for the purpose of testing the universality of the proposed ANN structure and the developed learning algorithm. Therefore, some ideal conditions are assumed in carrying out the simulation to put extra emphasizes on the tested properties. These assumptions include:

1- Inertial attitude angles and attitude rates are available for measurement.

2- Absence of measurement noise.

3- Sampling rate of 100 Hz.

4- Position measurements are available at 100 Hz.

Quadrotor parameters are shown in Table 4-2. Five cases are reported in this dissertation to show the robustness of the developed learning controller; first: when no aerodynamic uncertainties are applied, second: in the presence of model uncertainty, assuming that the mass of the quadcopter is roughly estimated, and the actual quadrotor is 1 kg heavier than the estimated value, third: when time dependent aerodynamic forces are applied, with different magnitudes and frequencies in the X, Y, and Z directions, fourth: when state dependent aerodynamic forces are applied, and finally: when a fixed wind force, with different X and Y components, is applied. The desired trajectory is generated using the method followed in [53]; which is developed in [89] so as to reduce jerk. The results of the five cases are discussed in the following.

Controller Design Parameters are chosen as: $\Lambda_1 = diag(20, 20, 20)$, $K_{r_1} = diag(58, 58, 58)$, $K_{i_1} = diag(29, 29.29)$, $\Lambda_2 = diag(10, 10, 10)$, $K_{r_2} = diag(30.4, 30.4, 30.4)$, and $K_{i_2} = diag(15.2, 15.2, 15.2)$. ANN parameters are selected as: $Q_{r_1} = Q_{r_2} = 1.2$, $Q_{i_1} = Q_{i_2} = 1$, $F_1 = F_2 = diag(0.9,0.9,0.9)$, and $\kappa_1 = \kappa_2 = 0.15$.

Case 1: No unknown aerodynamics or model uncertainties applied to the quadrotor dynamics. The dynamic response of the quadrotor is shown in Figure 4.12. The first ANN has learned some function Figure 4.12 (d), however, the force values are very small and accordingly, they are negligible. On the other hand, the attitude ANN have calculated fairly considerable values; that corresponds to the complex dynamics shown in Eq. 3-20 with $A_{\phi\theta\psi} = 0$. Figure 4.12 (f) shows the boundedness of attitude ANN weights. The weights of the translational ANN are not presented for their very small values.

Case 2: Model uncertainty, mass error. In this case, the quadrotor is 1 kg heavier than its assumed value for controller design. Figure 4.13 shows the dynamic response of the quadrotor in this case. Notice that the force ANN, shown in Figure 4.13 (d), have captured effectively the model uncertainty of -9.81 N (Red dashed line) in the Z direction. The force ANN weights are shown in Figure 4.13 (f). Furthermore, the trajectory error, shown in Figure 4.13 (a), quickly recovers from its high initial value, compared with case 1.

Case 3: Sinusoidal time dependent unknown aerodynamics. Three sinusoidal waves with different magnitudes and frequencies in the X, Y and Z directions are applied. The actual applied aerodynamics are given by $A_{xyz} = [0.5 * \sin(3 * t)\,, 1.5 * \sin(t)\,,\ \sin(0.5 * t)]^T N$. The performance of the quadrotor with this type of uncertainty is shown in Figure 4.14. Notice that the unknown aerodynamics are effectively captured by the ANN, as shown in Figure 4.14 (d). The force ANN weights are bounded as shown in Figure 4.14 (f). Despite the absence of unknown moment aerodynamics, the second ANN estimates the complex terms in the attitude dynamic equation.

Case 4: State dependent unknown aerodynamics. The applied aerodynamic forces are given by $A_{xyz} = \left[2 * \dot{X}, 0.5 * \dot{Y}, \dot{Z}\right]^T N$. Figure 4.15 shows the dynamic performance of the quadrotor with such aerodynamic forces. The force ANN has captured the unknown aerodynamics, as shown in Figure 4.15 (d), with bounded weights, as shown in Figure 4.15 (f).

Case 5: Fixed wind force in X and Y directions. The applied force in this case is $A_{xyz} = [-1, 0.5, 0]^T N$. Figure 4.16 shows the performance of the quadrotor. Notice, that unknown forces are effectively estimated by the ANN, as shown in Figure 4.16 (d), with bounded weights.

In general, the overall performance in all cases is very good, including in the cases where the unknown part of the dynamics has no straightforward relationship with system's states. These results show the effectiveness of the proposed ANN structure and the developed learning algorithm.

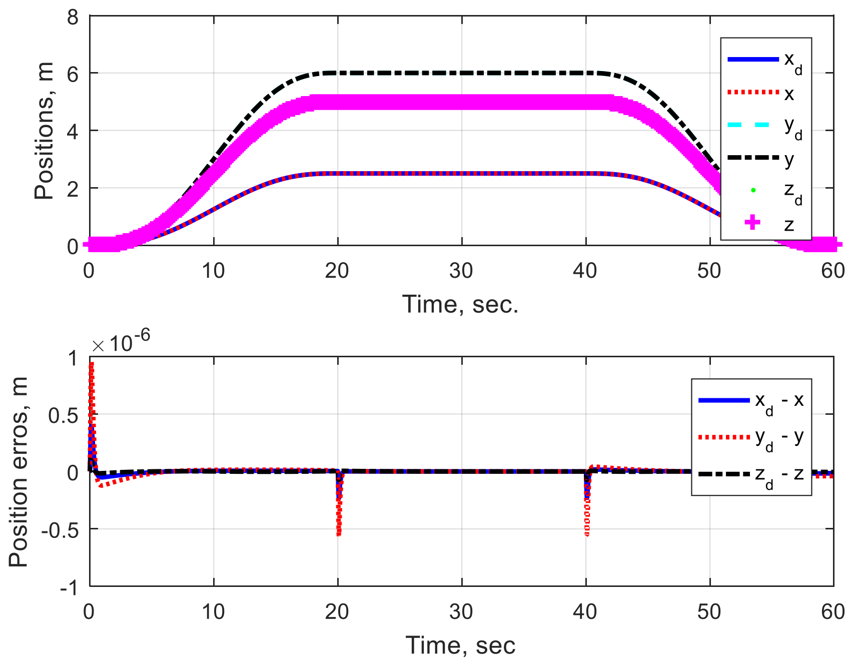


Figure 4.12 (a) Simulation Results for Case 1: Translational Response

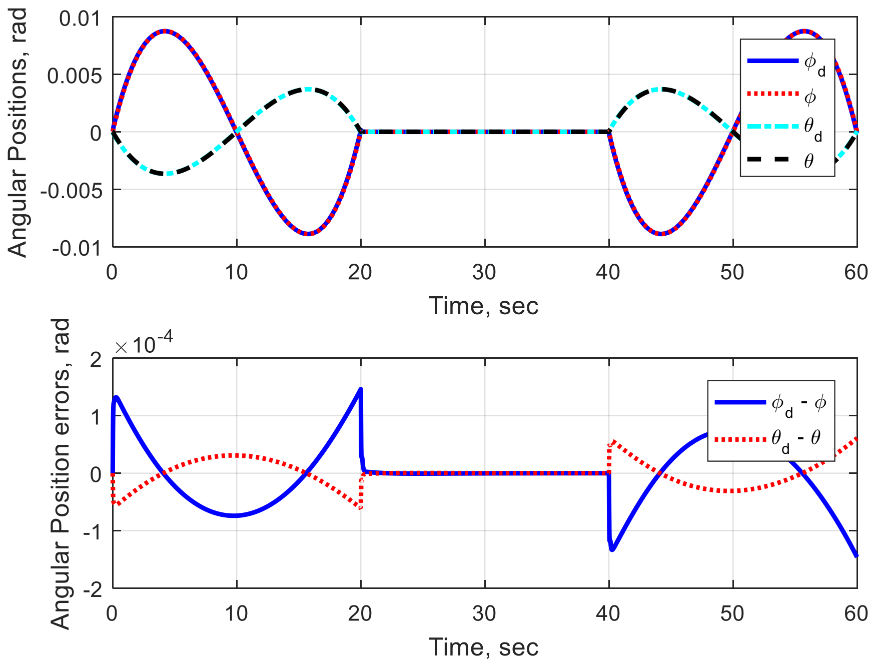


Figure 4.12 (b) Simulation Results for Case 1: Attitude Response

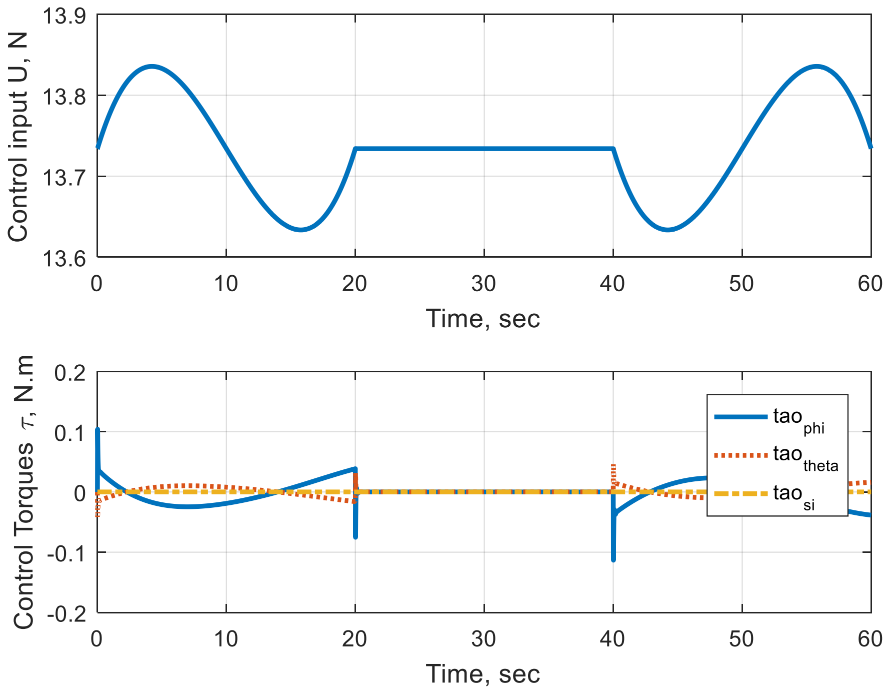


Figure 4.12 (c) Simulation Results for Case 1: Control Force and Torque Inputs

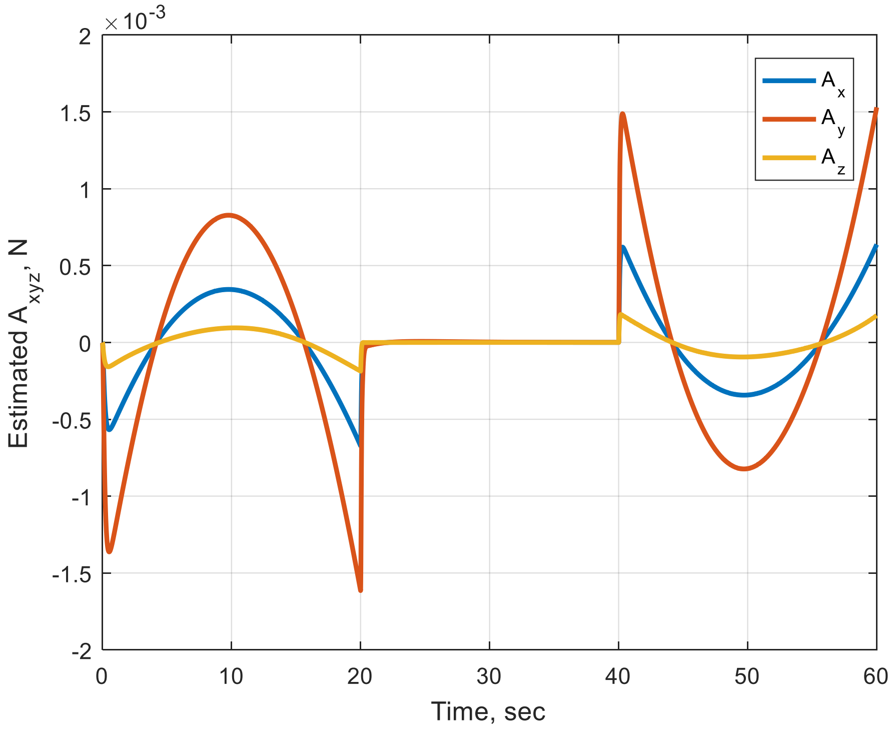


Figure 4.12 (d) Simulation Results for Case 1: (d) ANN Force Realization

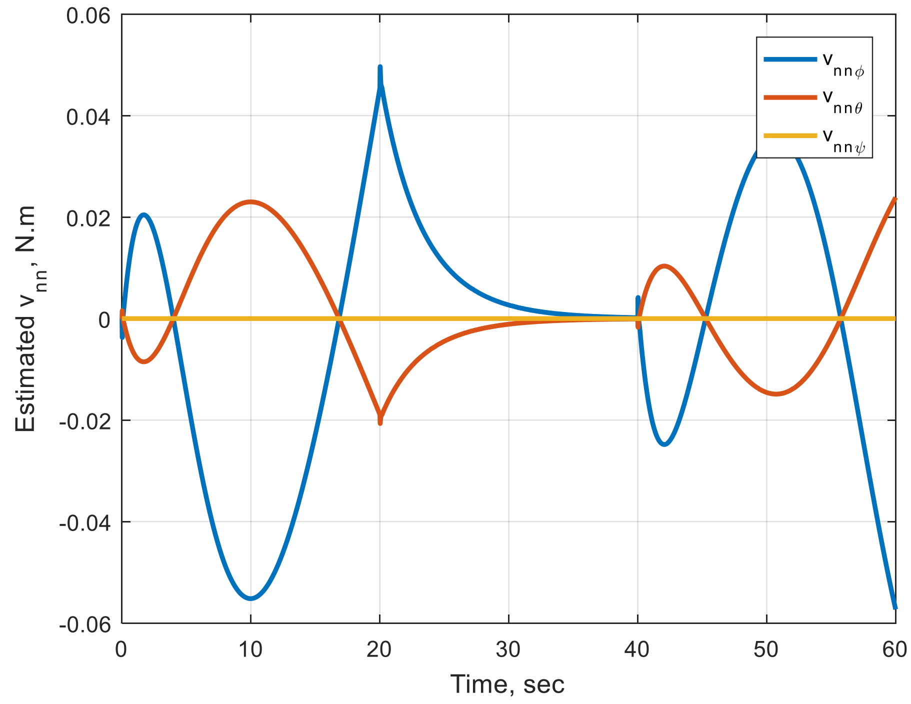


Figure 4.12 (e) Simulation Results for Case 1: ANN Moment realization

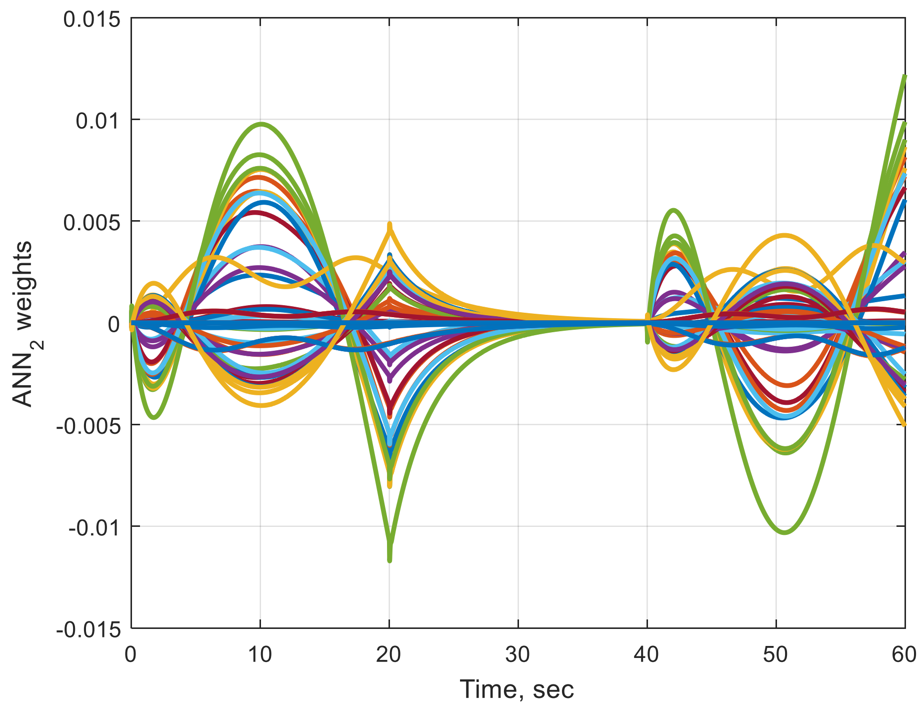


Figure 4.12 (f) Simulation Results for Case 1: Attitude ANN Weights

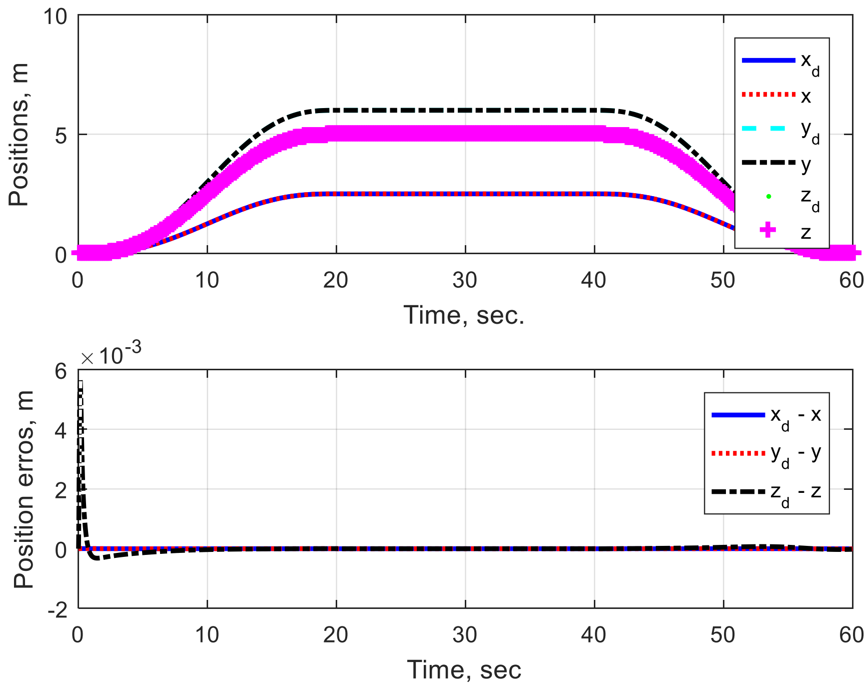


Figure 4.13 (a) Simulation Results for Case 2: Translational Response

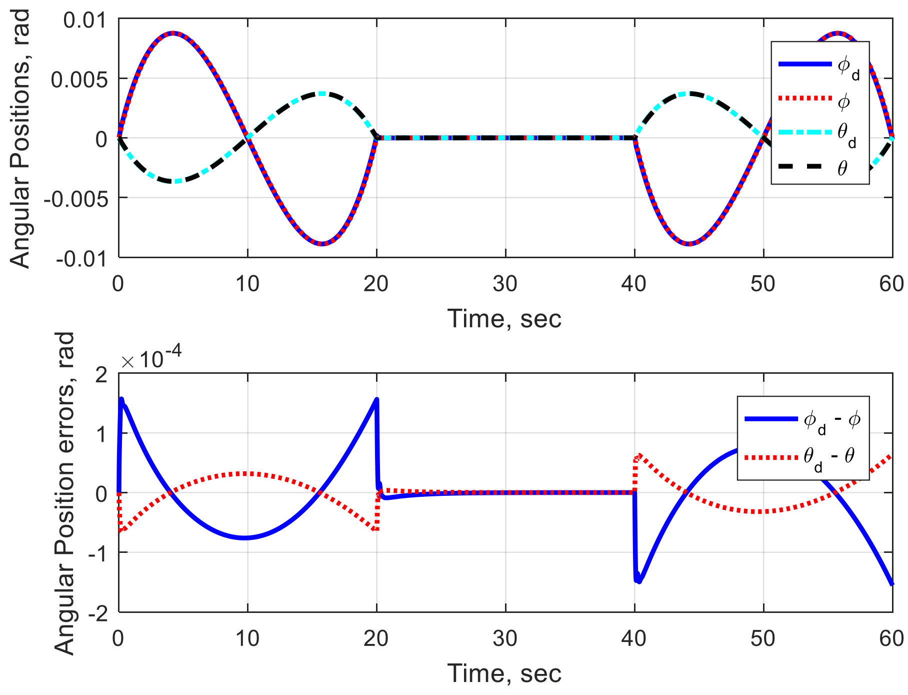


Figure 4.13 (b) Simulation Results for Case 2: Attitude Response

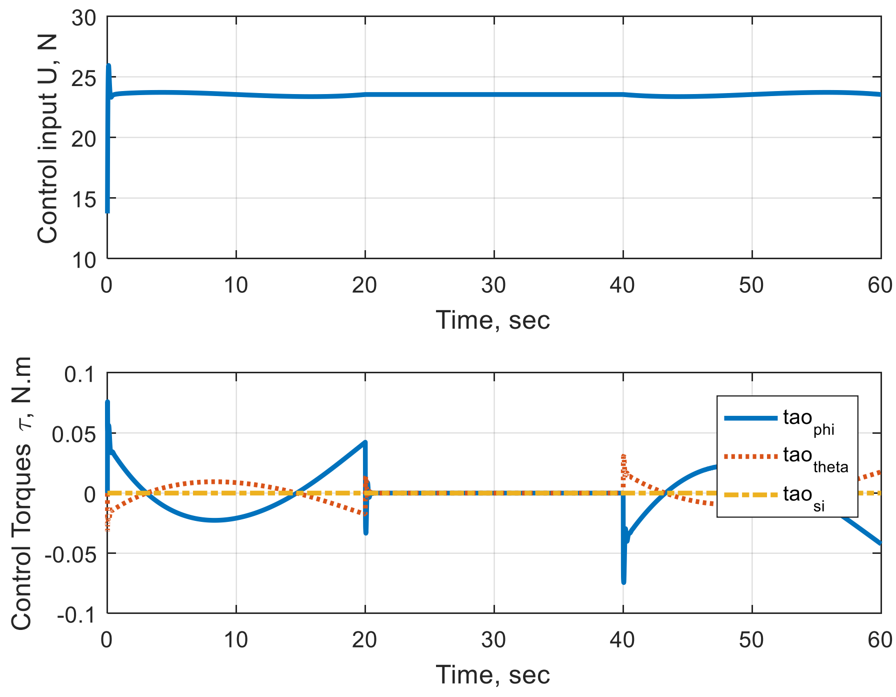


Figure 4.13 (c) Simulation Results for Case 2: Control Force and Torque Inputs

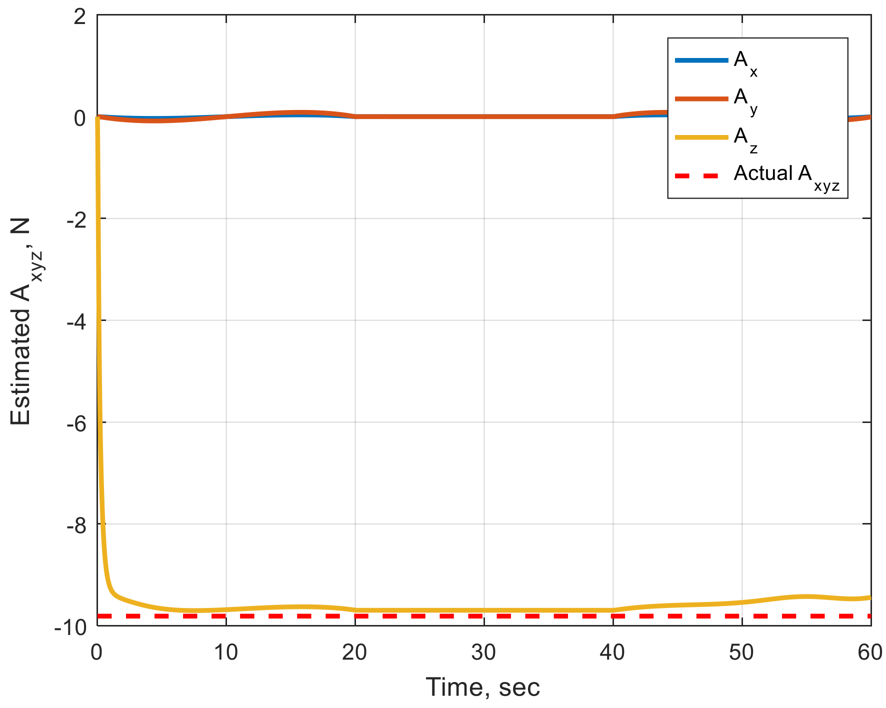


Figure 4.13 (d) Simulation Results for Case 2: ANN Force Realization

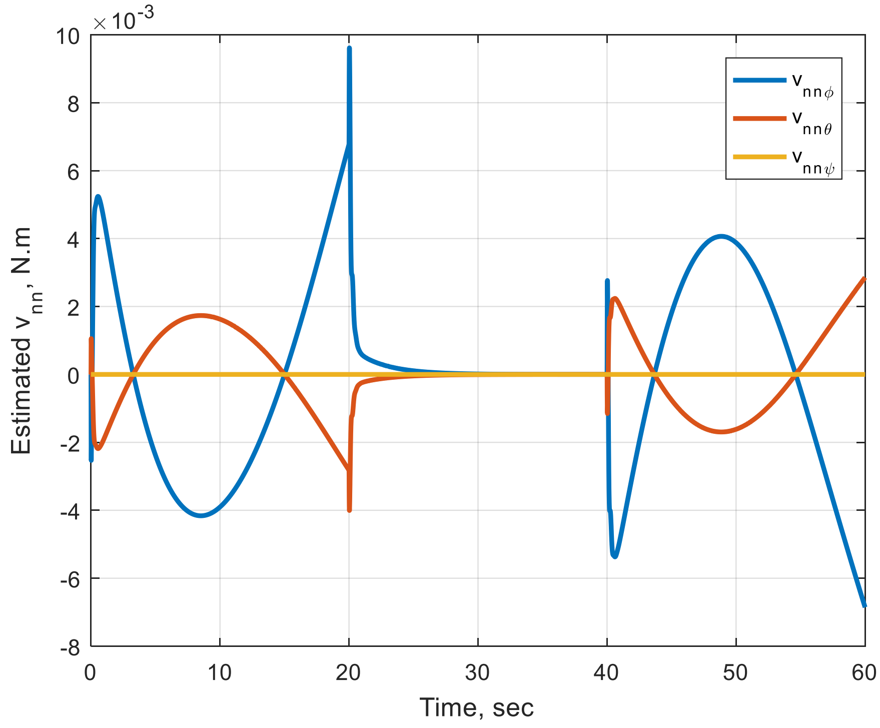


Figure 4.13 (e) Simulation Results for Case 2: ANN Moment Realization

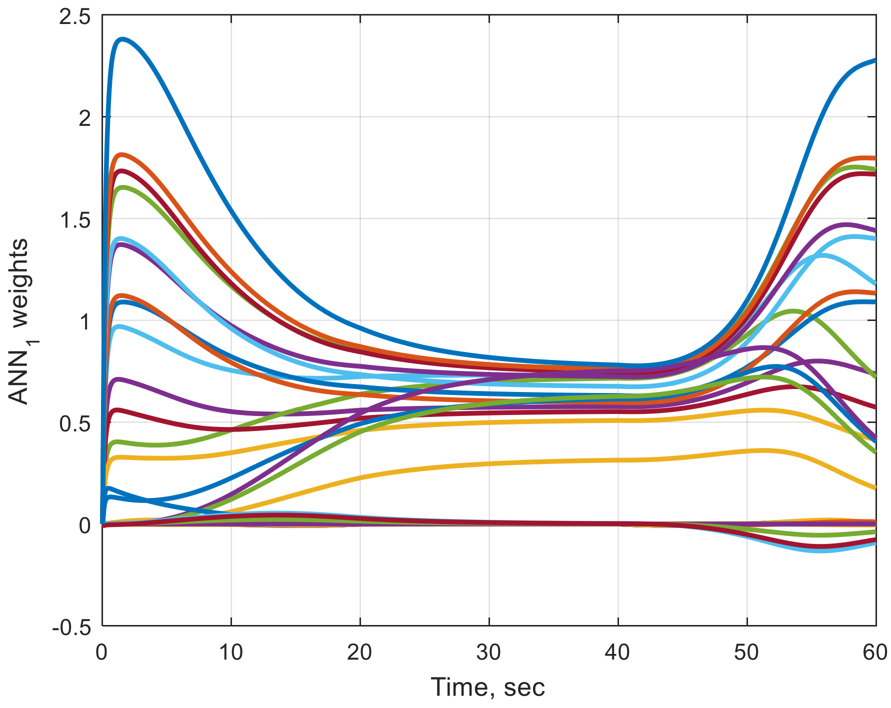


Figure 4.13 (f) Simulation Results for Case 2: Force ANN Weights

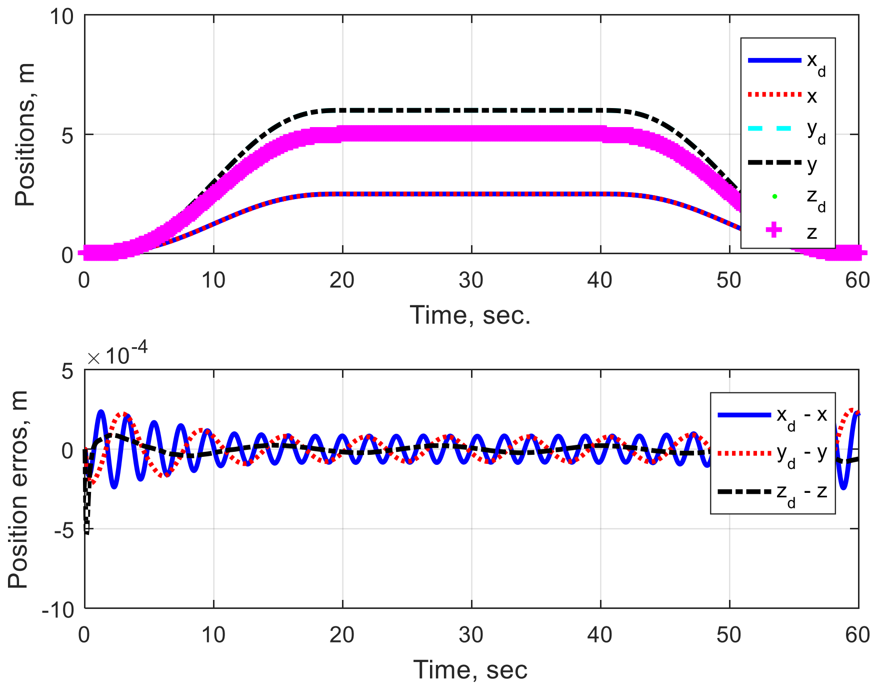


Figure 4.14 (a) Simulation Results for Case 3: Translational Response

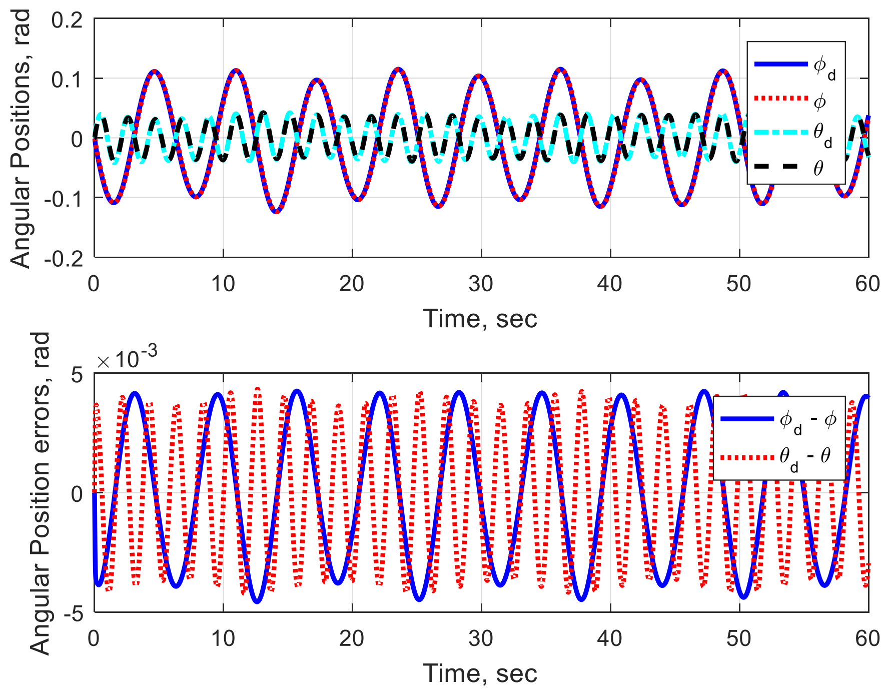


Figure 4.14 (b) Simulation Results for Case 3: Attitude Response

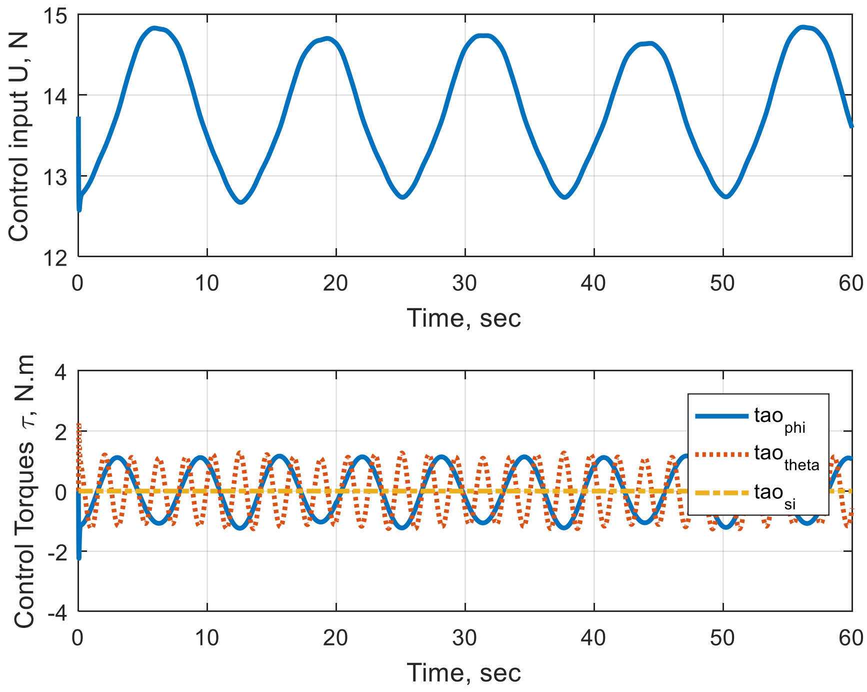


Figure 4.14 (c) Simulation Results for Case 3: Control Force and Torque Inputs

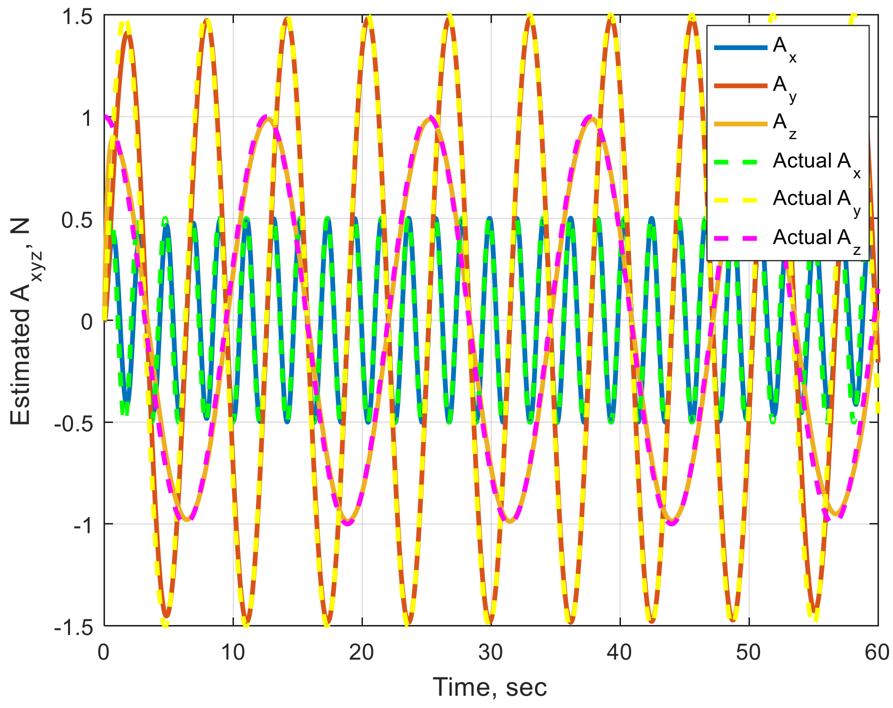


Figure 4.14 (d) Simulation Results for Case 3: ANN Force Realization

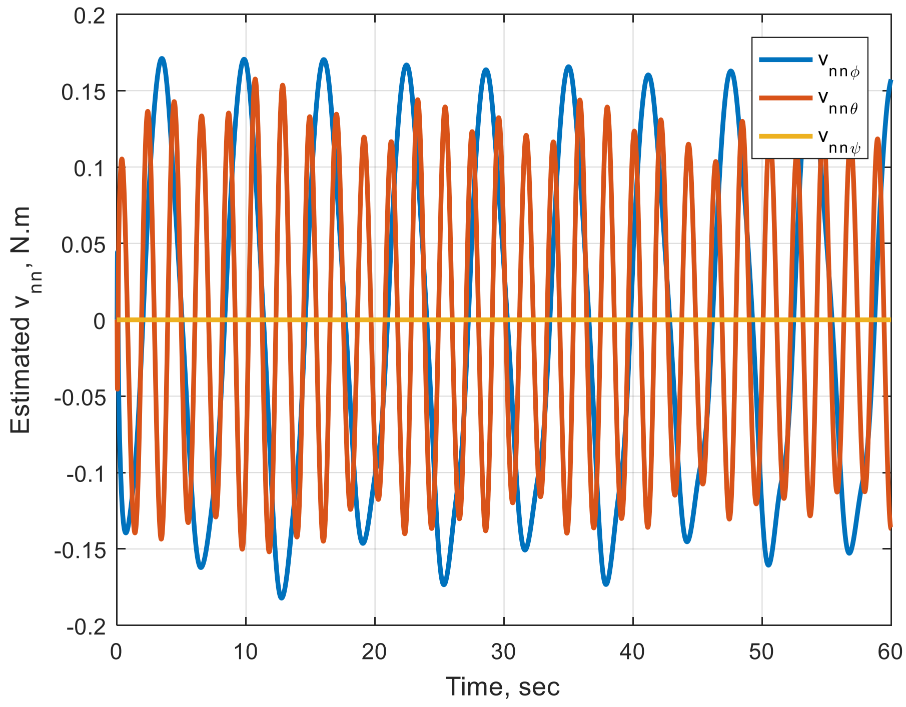


Figure 4.14 (e) Simulation Results for Case 3: ANN Moment Realization

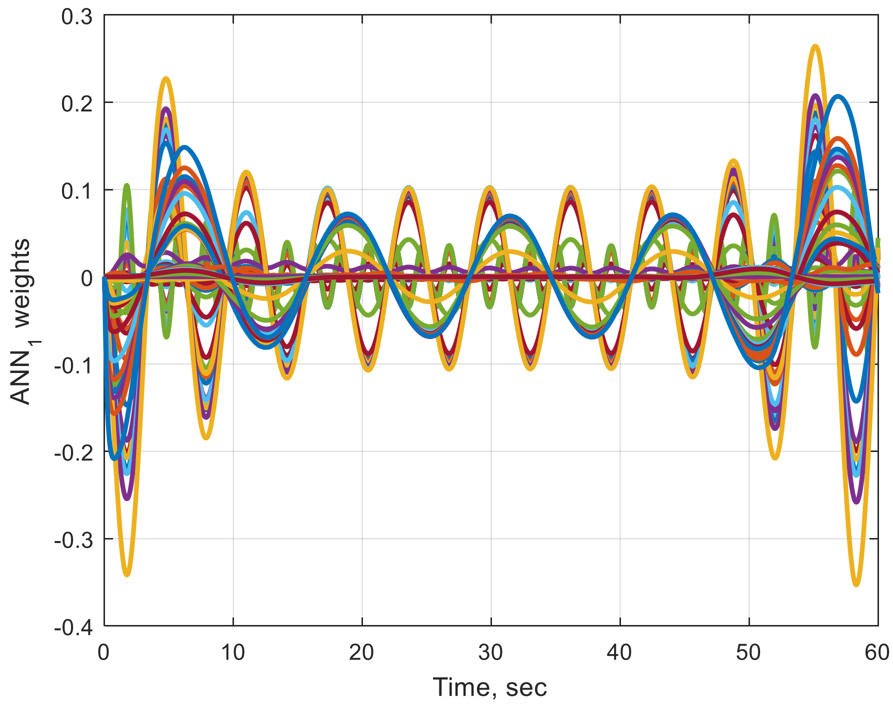


Figure 4.14 (f) Simulation Results for Case 3: Force ANN Weights

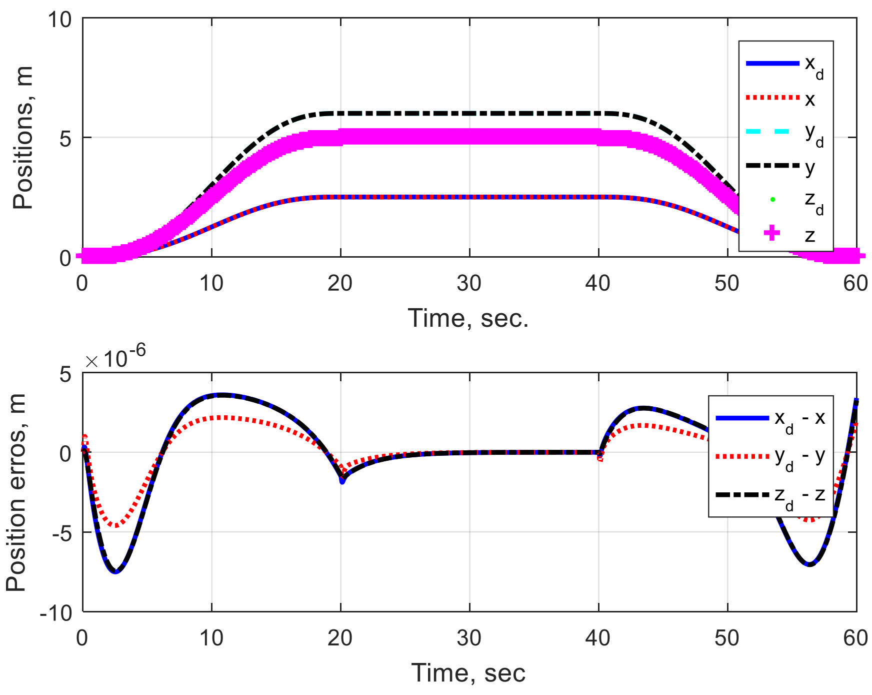


Figure 4.15 (a) Simulation Results for Case 4: Translational Response

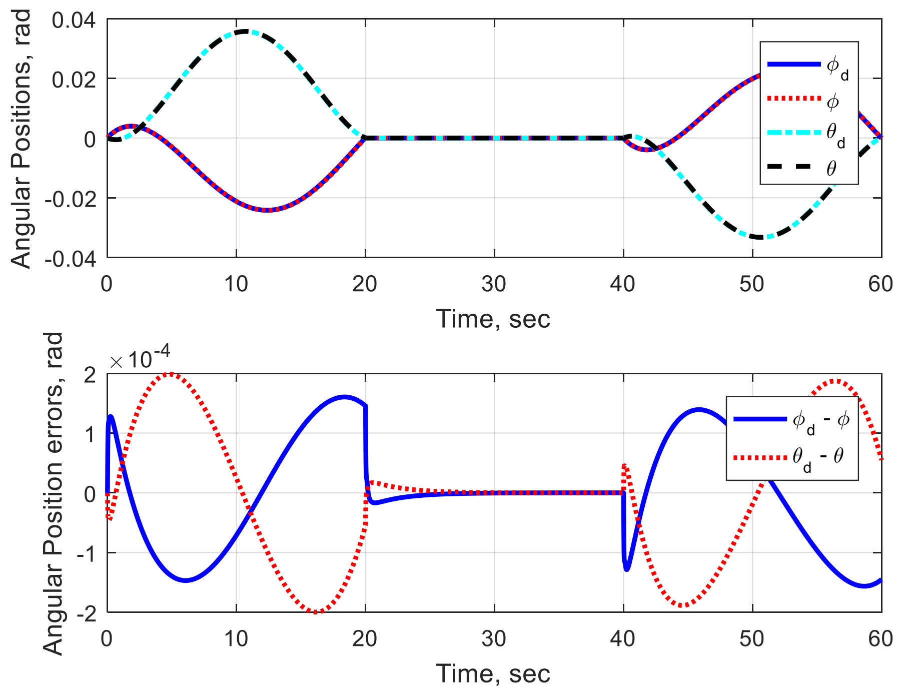


Figure 4.15 (b) Simulation Results for Case 4: Attitude Response

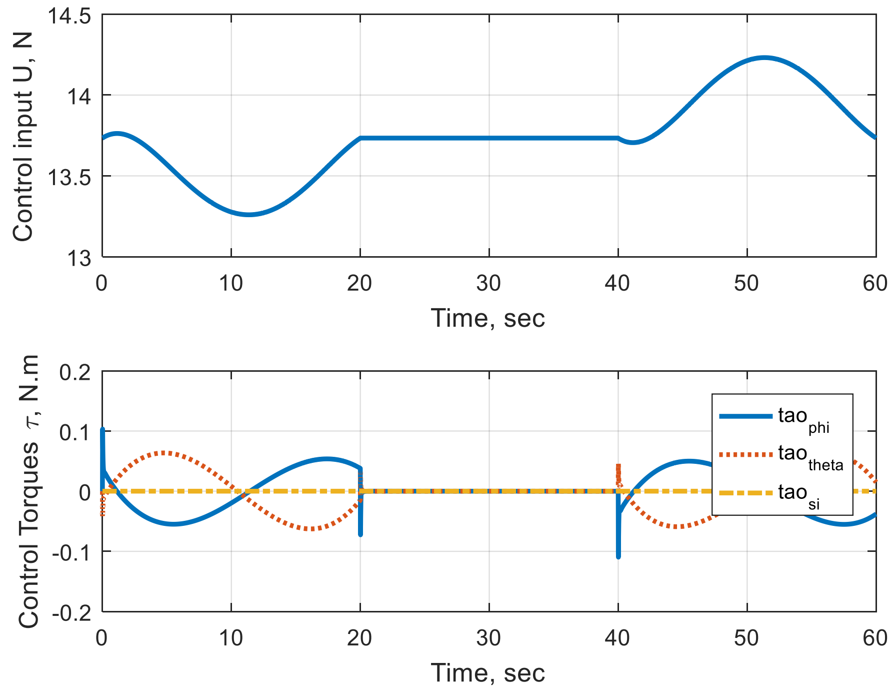


Figure 4.15 (c) Simulation Results for Case 4: Control Force and Torque Inputs

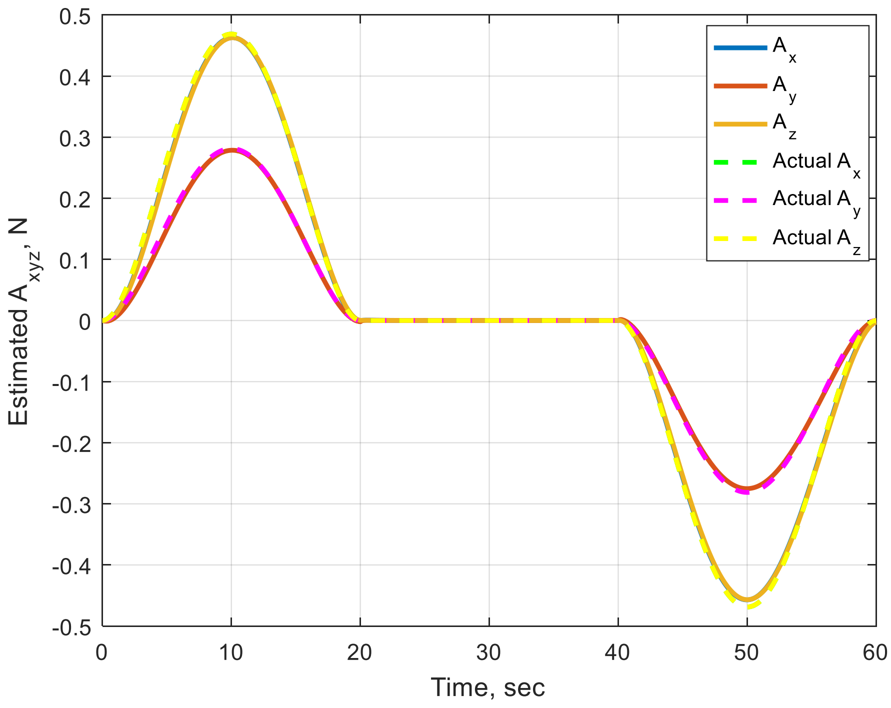


Figure 4.15 (d) Simulation Results for Case 4: ANN Force Realization

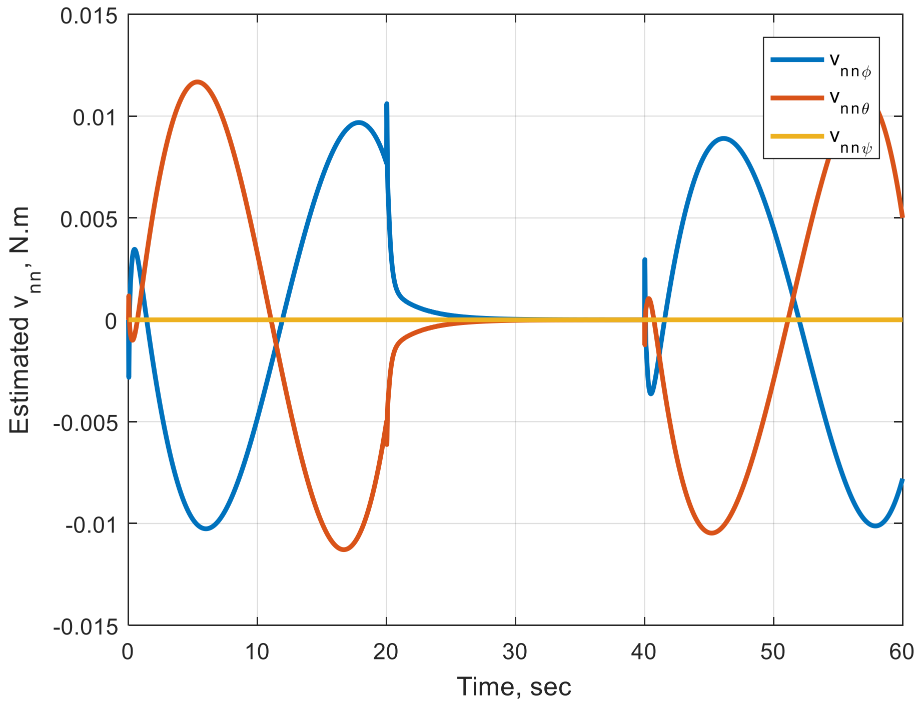


Figure 4.15 (e) Simulation Results for Case 4: ANN Moment Realization

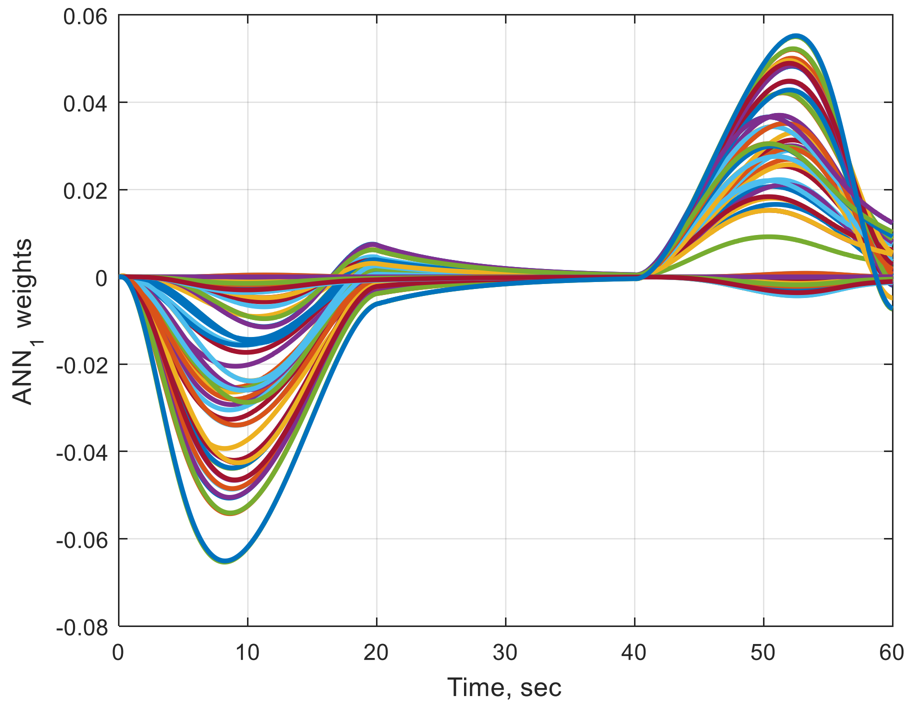


Figure 4.15 (f) Simulation Results for Case 4: Force ANN Weights

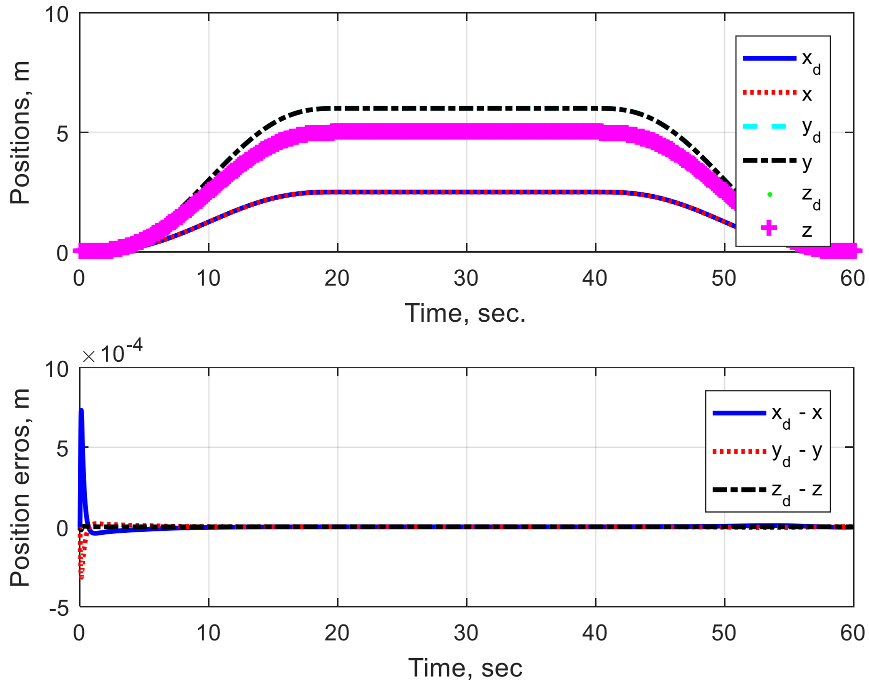


Figure 4.16 (a) Simulation Results for Case 5: Translational Response

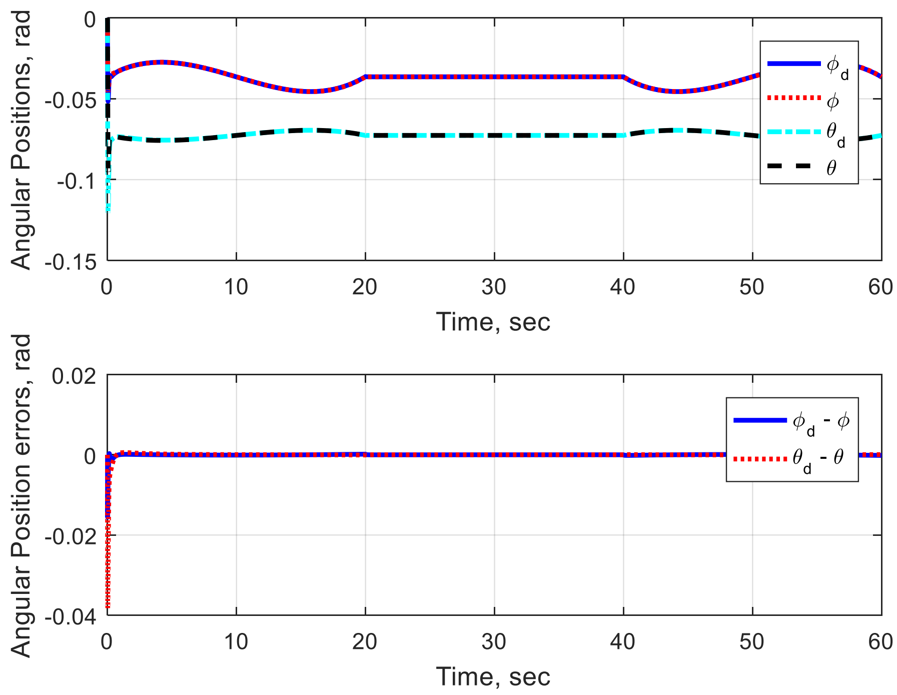


Figure 4.16 (b) Simulation Results for Case 5: Attitude Response

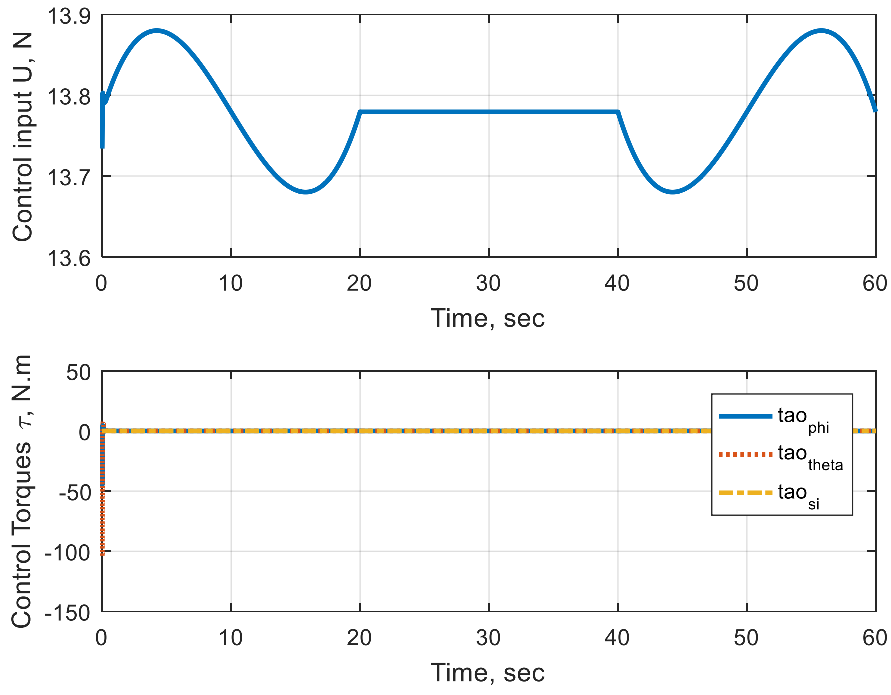


Figure 4.16 (c) Simulation Results for Case 5: Control Force and Torque Inputs

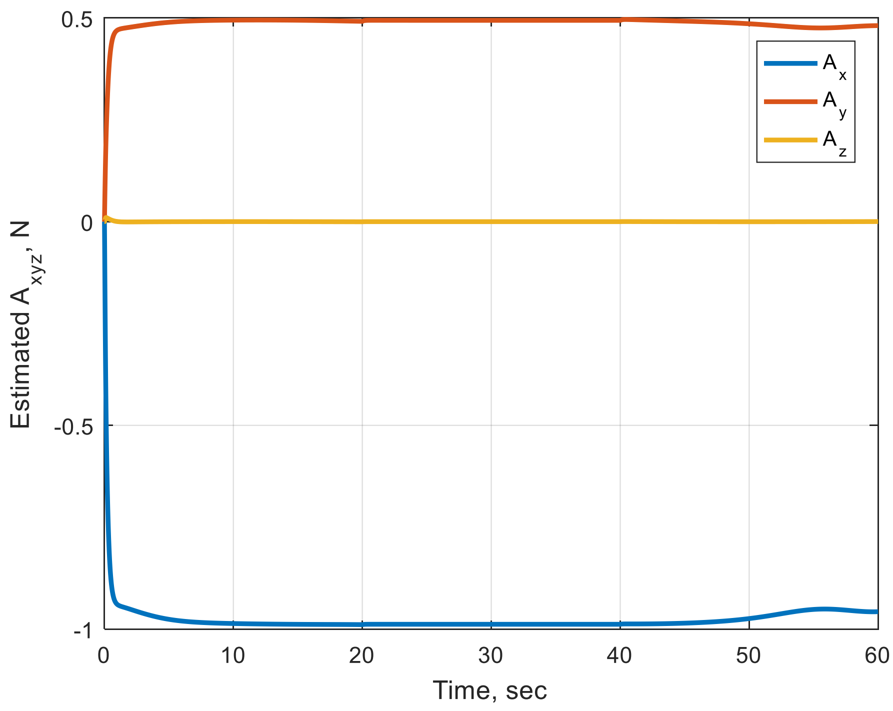


Figure 4.16 (d) Simulation Results for Case 5: ANN Force Realization

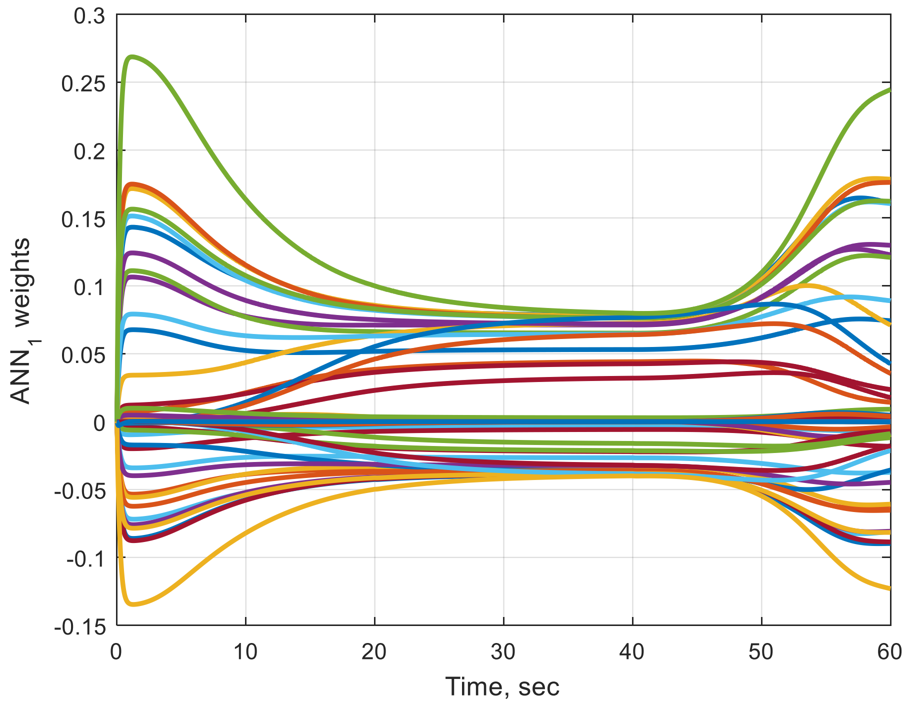


Figure 4.16 (e) Simulation Results for Case 5: Force ANN Weights

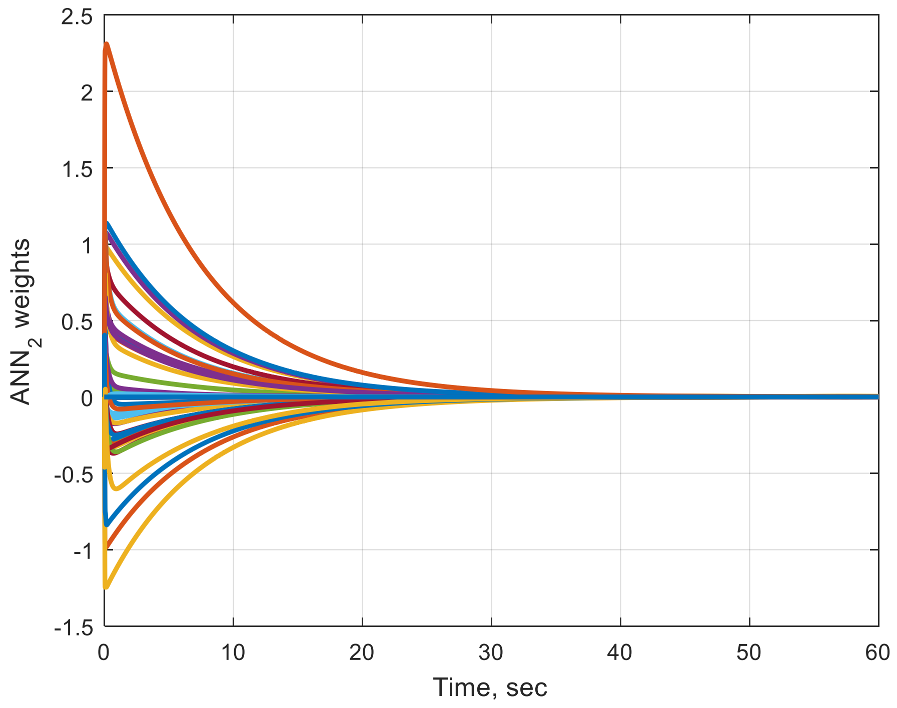


Figure 4.16 (f) Simulation Results for Case 5: Moment ANN Weights

4.5 EXPERIMENTAL RESULTS

The laboratory in which experiments were conducted represents an inherent confined environment, as it was not designed for UAV research. In addition, experiments were performed with a focus on low altitude to increase exposure to ground effects. Therefore, even when Qball is flying away from walls, it still experiences aerodynamics uncertainties associated with confined environment.

It is important to mention that the original Qball PID controller is designed for the original hardware, which is modified by the upgrade of the batteries (weight), and the Parrot SLAM dunk sensor (weight plus CG shift). This hardware modification introduces two factors that have not been accounted for in the controller design: Weight uncertainty, and center of gravity location uncertainty. Therefore, Qball's performance under the PID controller is expected to deteriorate from its designed performance. Despite the fact that an improvement in performance could be achieved by adjusting controller parameters, the original Qball controller gains were not optimized for these modifications. Thus, the author wants to clarify that the original controller is used here as baseline controller to evaluate the performance of ANN, rather than to evaluate its design.

Experiment results presented here are designed to push the quadrotor to its physical limits to test stability, robustness, and effectiveness of the ANN controller. Eight cases were conducted, and the results are summarized in Table 4-4, and are detailed as follows:

4.5.1 No Walls Nearby

The environment setup in this case is shown in Figure 4.17. Qball is commanded to hover at different altitudes when there are no nearby walls. This description is loosely used here to describe the situation when there is no barrier within about 2 m around the Qball. Figure 4.18 shows the response in this case when the ANN controller is used.

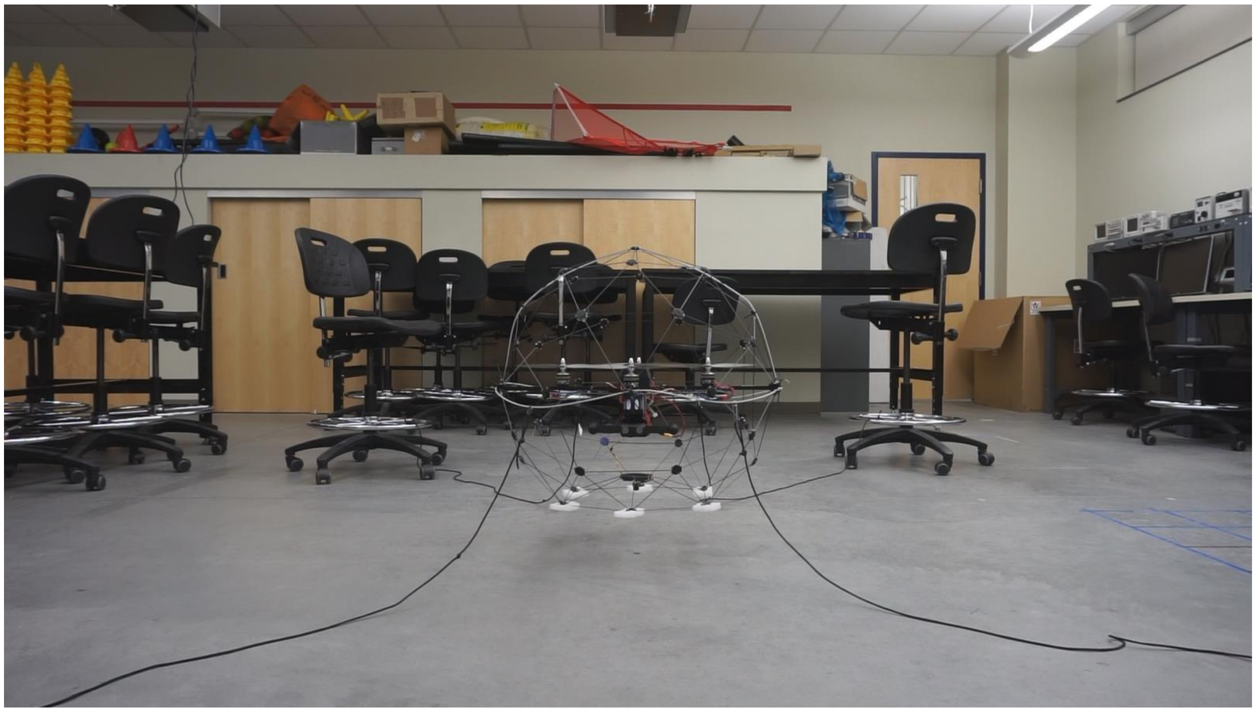


Figure 4.17 Environment Setup for Case 1: No Nearby Wall

It can be seen, in Figure 4.18 (e), that the X force identified by the ANN is shifted towards a positive value, with a mean of $0.62\ N$. This shift could be caused by either the CG shift resulting from the SLAM sensor and the batteries, or because of an asymmetry on the longitudinal thrust mechanism. The Y component of the identified force is shifted towards a negative value, with a mean of $-0.26\,N$, which could be caused by the same reasons causing the longitudinal shift. The realized Z force indicates that the Qball is heavier than expected. This result was expected because of the additional weight added by the SLAM sensor, SLAM's holder and the batteries upgrade. Nevertheless, it should be noted that this realized weight difference is not the exact weight change, as the integral term in the altitude controller should have compensated for some of the weight difference. The cyclic nature of the three identified forces is an indication for the presence of aerodynamic fluctuations associated with being in a confined environment, despite the fact that there are no nearby barriers.

The torque ANN has captured a negative pitch torque and a positive roll torque, these are attributed to the IMU misalignment, CG shift, and thrust system asymmetry. As it is observed that Roll and Pitch measurements deviate from zero when the Qball is leveled.

Notice that the errors in X and Y decrease with time, as the ANNs learn more with time. Also notice the shift in the pitch and roll responses which is corrected through time.

This environment almost represents the ideal condition for PID controller. However, two factors make it not ideal:

1- The hardware changes.

2- The laboratory environment. As typical UAV research laboratories have higher roofs and bigger empty arenas.

Figure 4.19 shows Qball's response for the first case when using the PID controller. The performance is very good, however, pitch and roll commands needed to bang between their limits to maintain the position. In addition, notice the shift in pitch and roll response, which align with the identified torques by the ANN's. Trajectory tracking errors are oscillating at a fixed magnitude, because the PID controller is not adaptive.

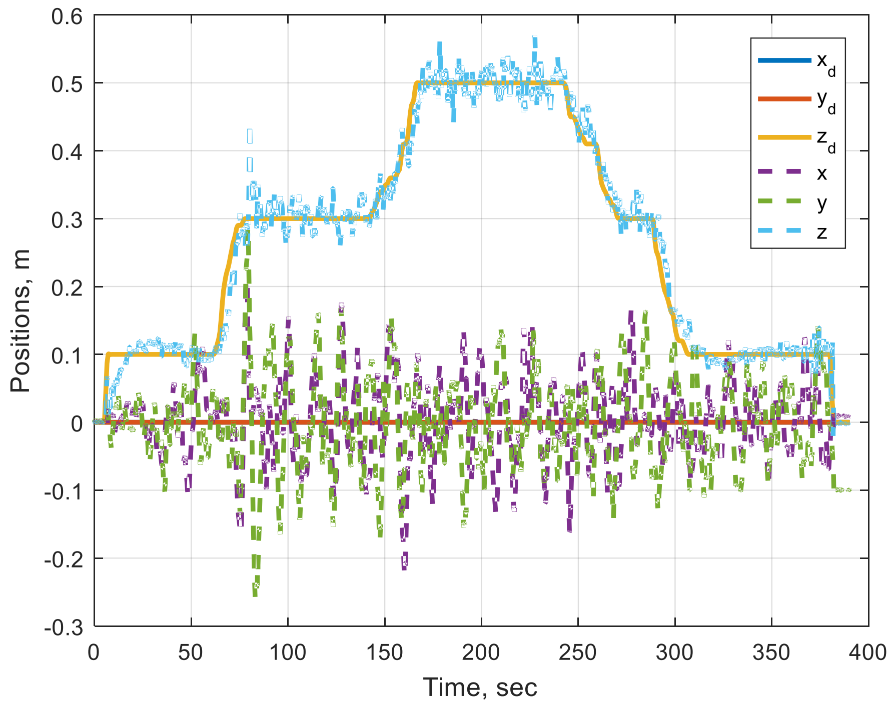


Figure 4.18 (a) Experimental Results for Case 1 Using ANN Controller: Translational Response

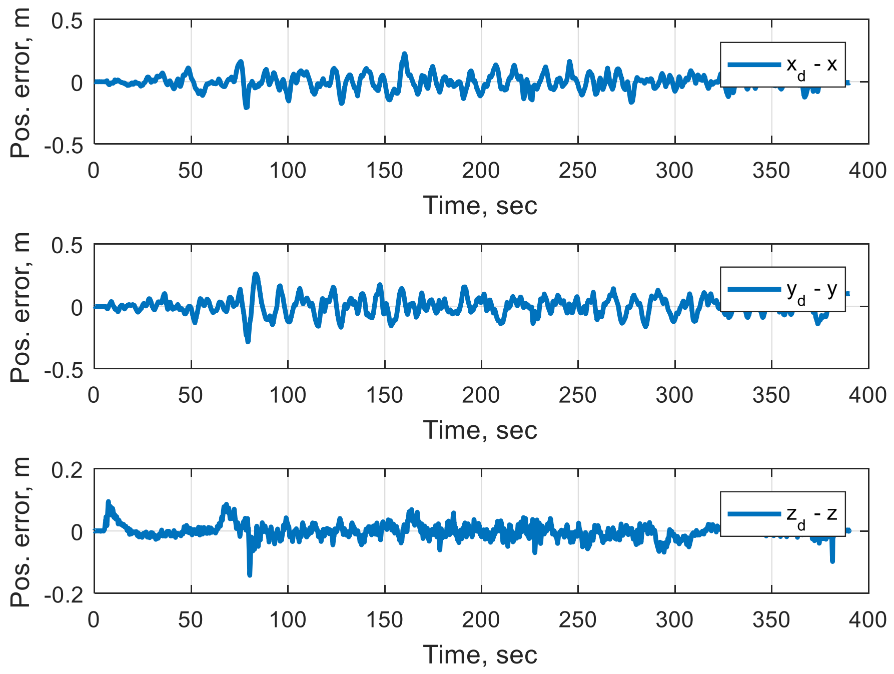


Figure 4.18 (b) Experimental Results for Case 1 Using ANN Controller: Position Errors

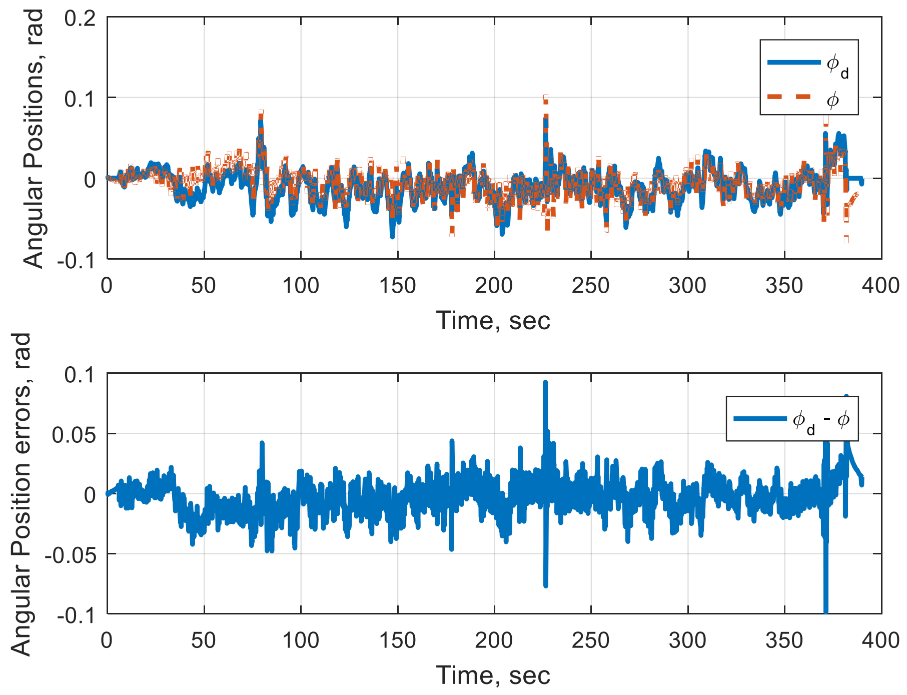


Figure 4.18 (c) Experimental Results for Case 1 Using ANN Controller: Roll Angle Response

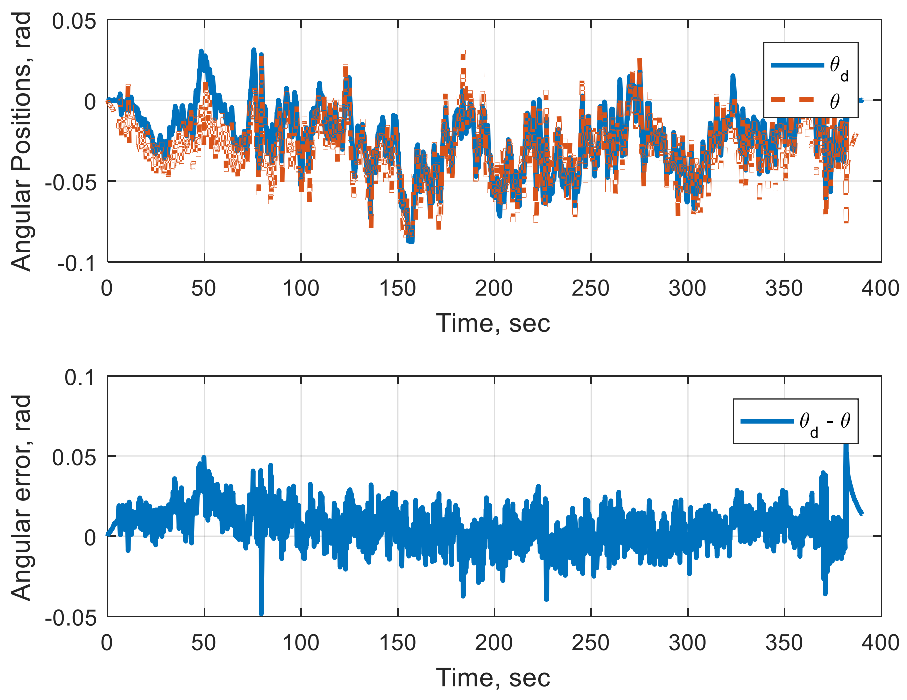


Figure 4.18 (d) Experimental Results for Case 1 Using ANN Controller: Pitch Angle Response

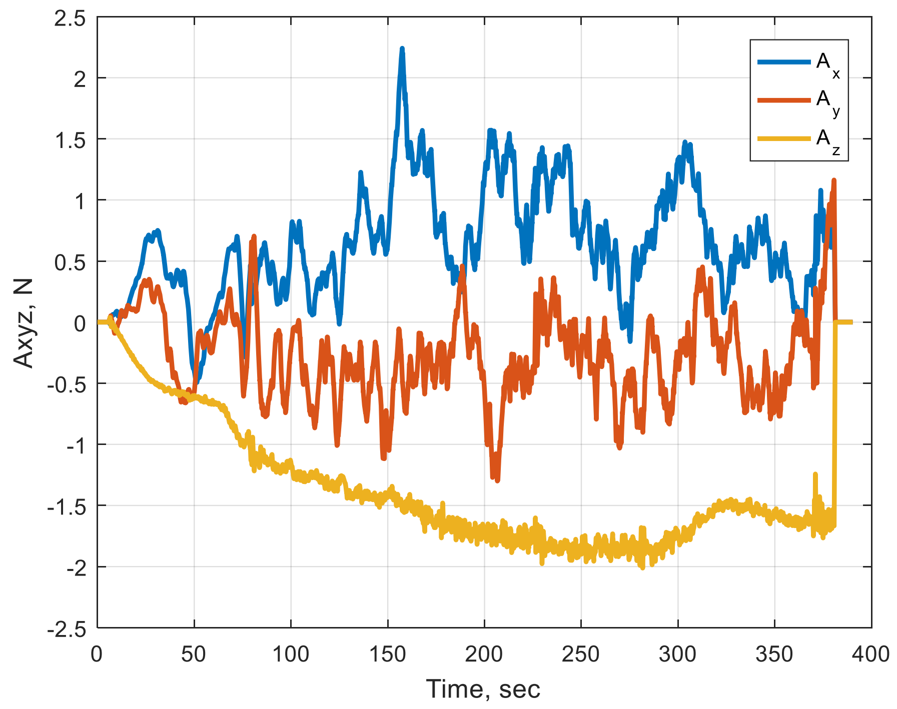


Figure 4.18 (e) Experimental Results for Case 1 Using ANN Controller: ANN Realized Forces

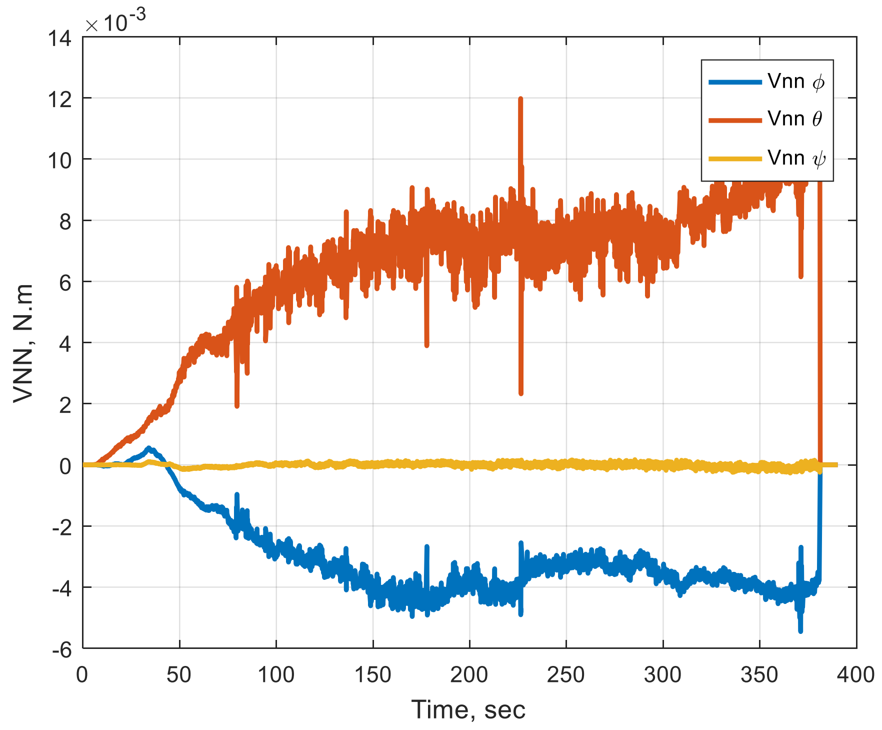


Figure 4.18 (f) Experimental Results for Case 1 Using ANN Controller: ANN Realized Moments

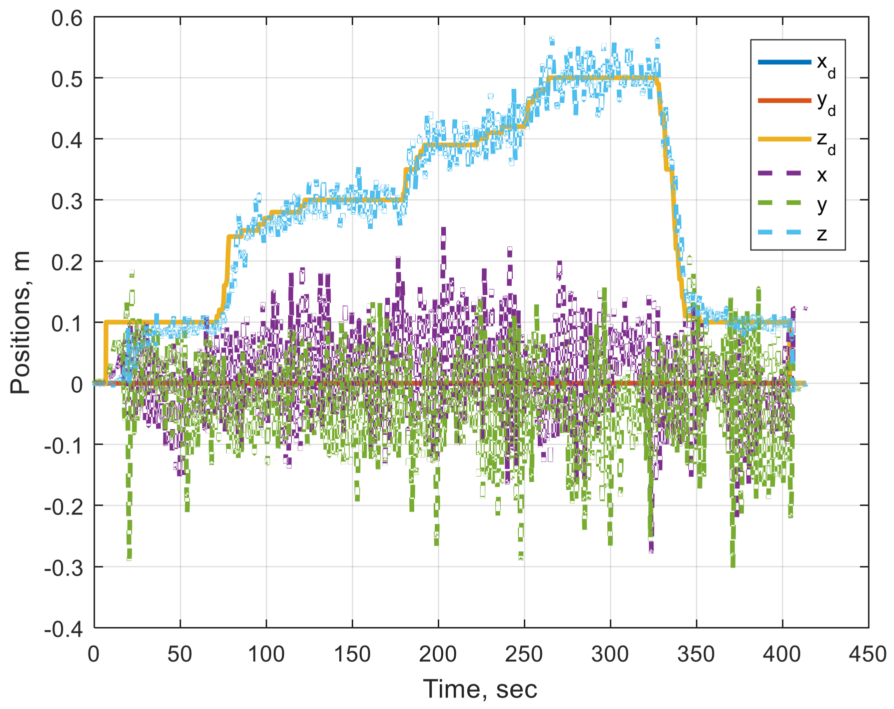


Figure 4.19 (a) Experimental Results for Case 1 Using PID Controller: Translational Response

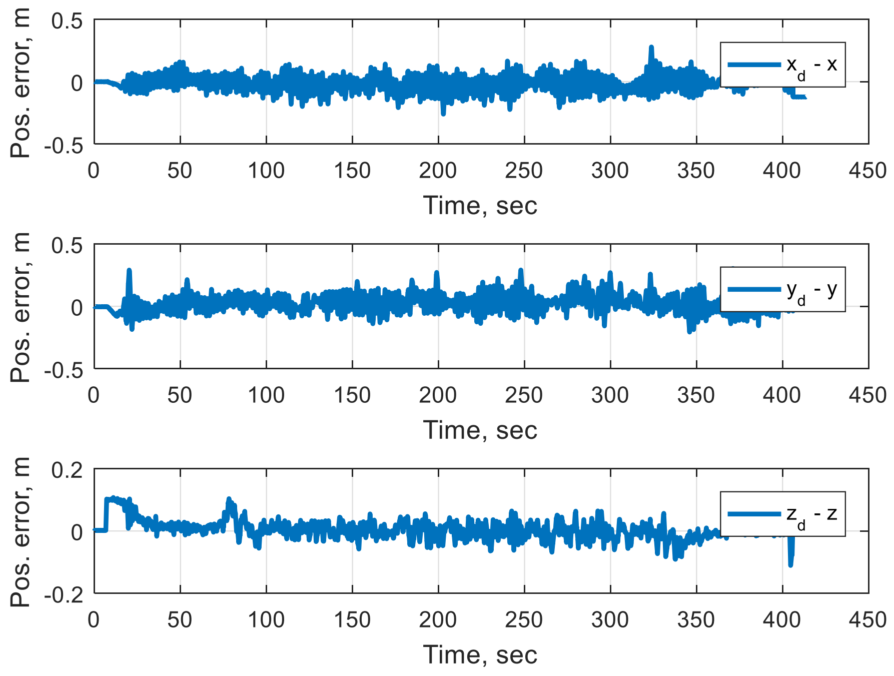


Figure 4.19 (b) Experimental Results for Case 1 Using PID Controller: Position Errors

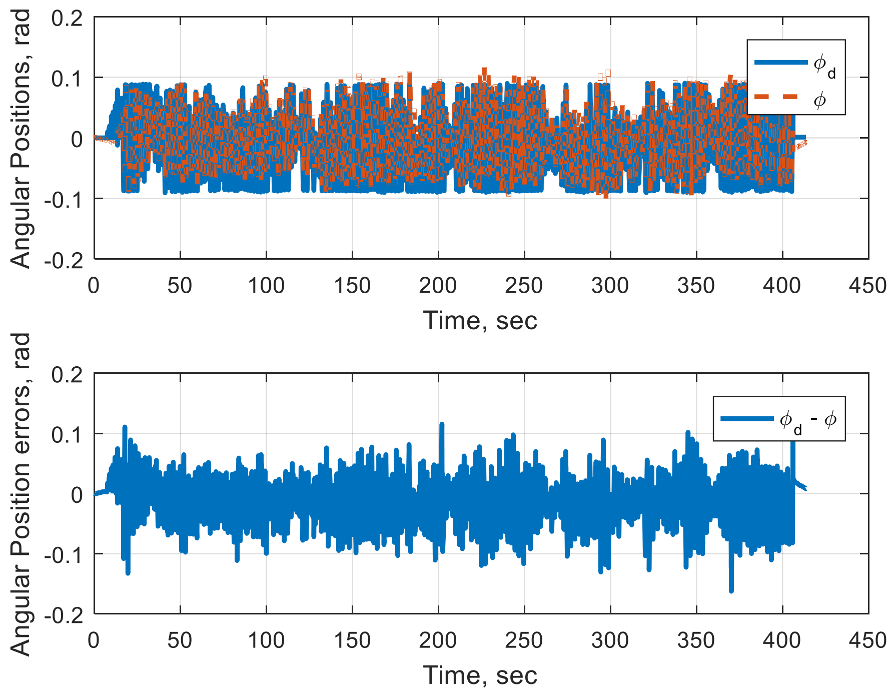


Figure 4.19 (c) Experimental Results for Case 1 Using PID Controller: Roll Angle Response

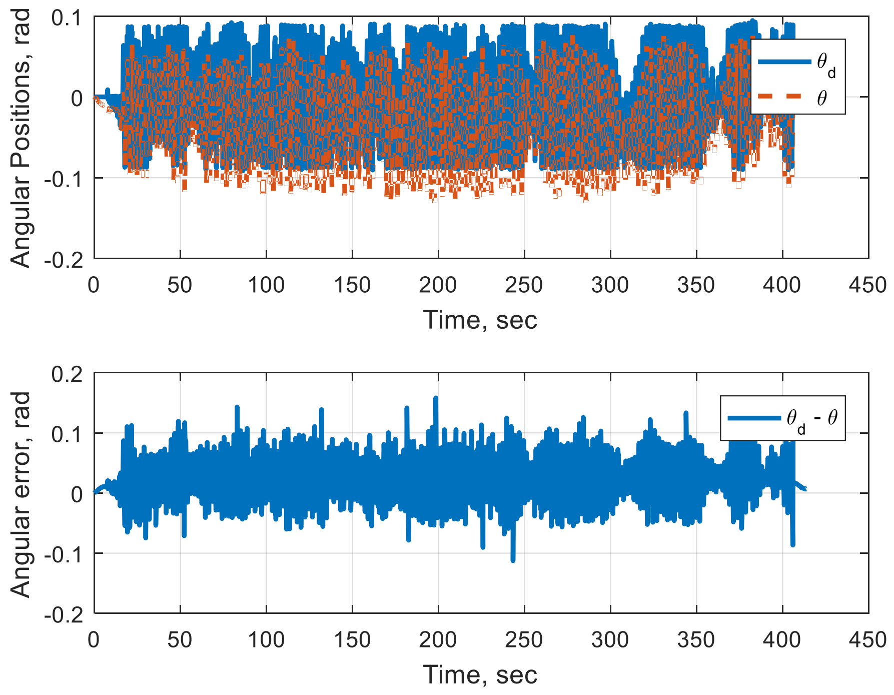


Figure 4.19 (d) Experimental Results for Case 1 Using PID Controller: Pitch Angle Response

4.5.2 One Wall Nearby

The setup in this case is shown in Figure 4.20. The Qball is required to hover close to a wall at a low altitude. It is observed that this case represents the second most challenging to control the Qball. Figures 4.21 and 4.22 show the time response of the Qball in this case. The PID controller has failed to maintain stability for 180 seconds, as the Qball became unstable and crash on the wall. While, the ANN controller could maintain stability for about 500 seconds. Notice that the error in Y direction is reducing significantly with time. Force ANNs identified forces in the X, Y, and Z directions. The Y component of the force has a greater magnitude, with a mean of $-0.3665\,N$, towards the wall with more oscillations compared with the *No-wall* case; this indicates that the wall is pulling the quadrotor towards it. The reason for the exponential increase in the Z force is the safety tethers, as one of them was trapped under the wall (table) and pulled the Qball downwards. The identified torques in the θ and ϕ have changed directions compared to that of the first case. The roll torque is negative, which means that the Qball is trying to roll away from the wall. This could be caused by the reflected air at the corner of the wall and the ground, as Qball is flying at low altitude, which creates more thrust on the right propeller when the Qball rolls.

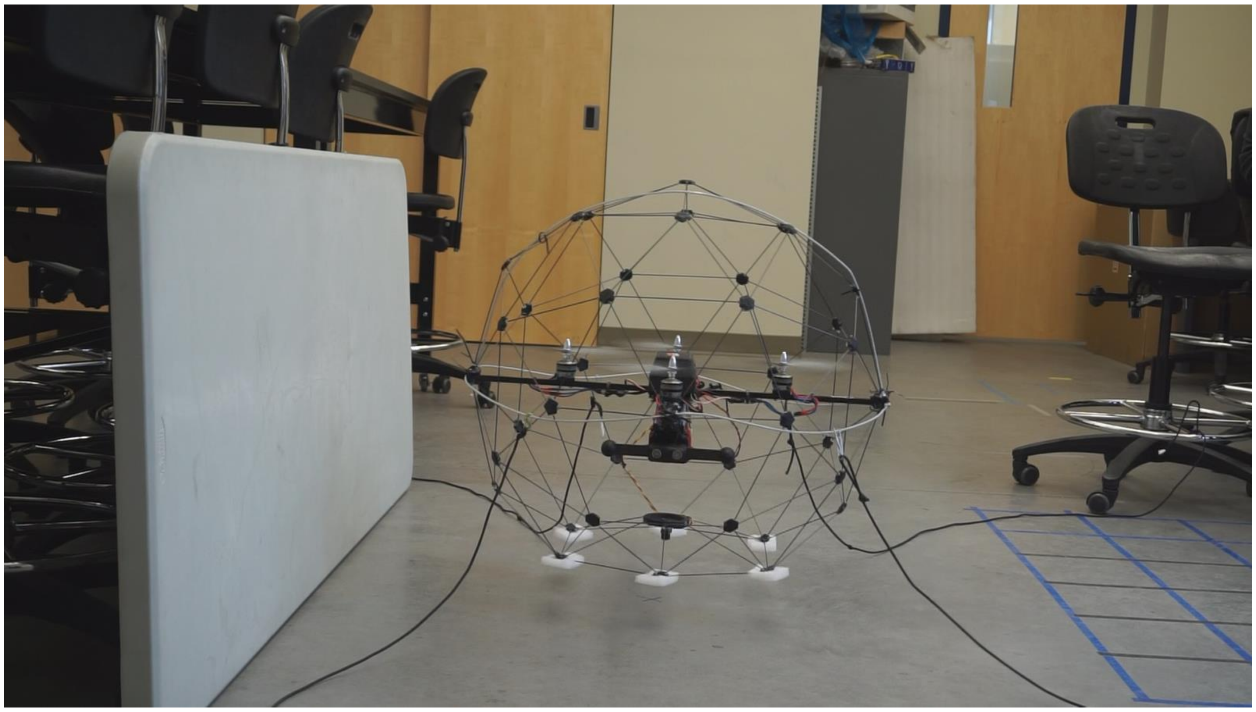


Figure 4.20 Environment Setup for Case 2: One Wall

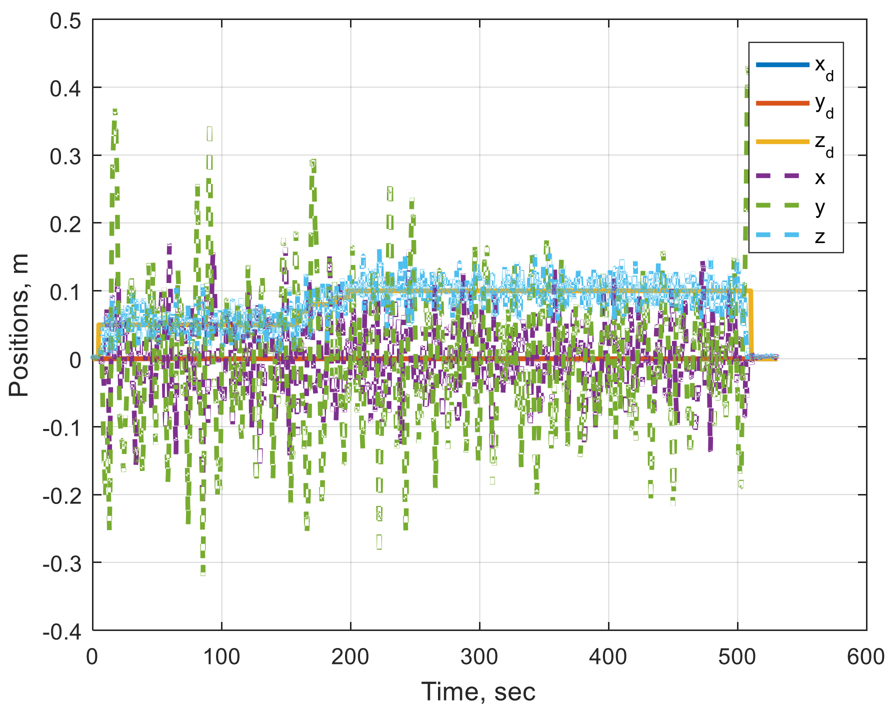


Figure 4.21 (a) Experimental Results for Case 2 Using ANN Controller: Translational Response

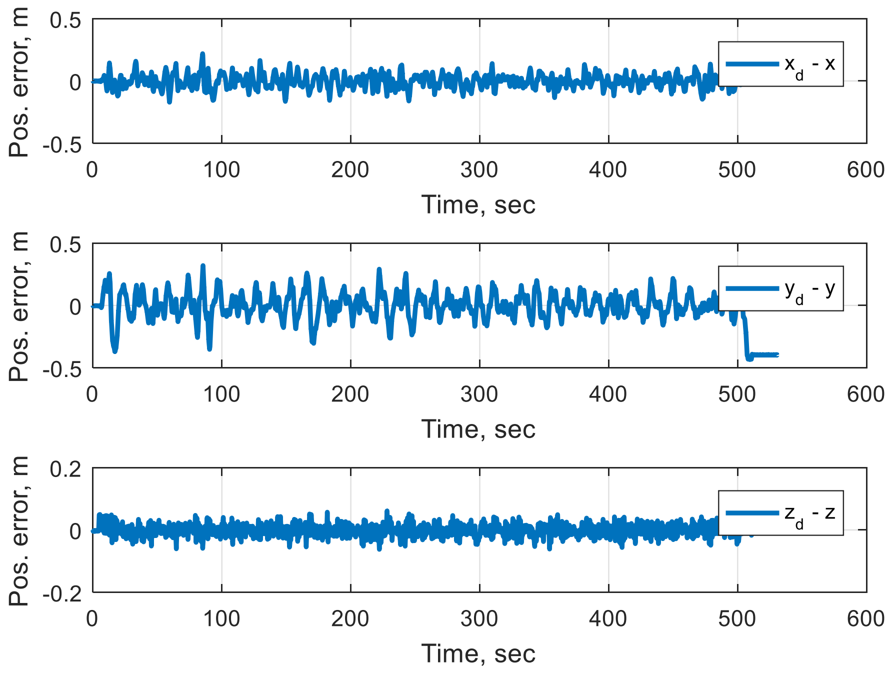


Figure 4.21 (b) Experimental Results for Case 2 Using ANN Controller: Position Errors

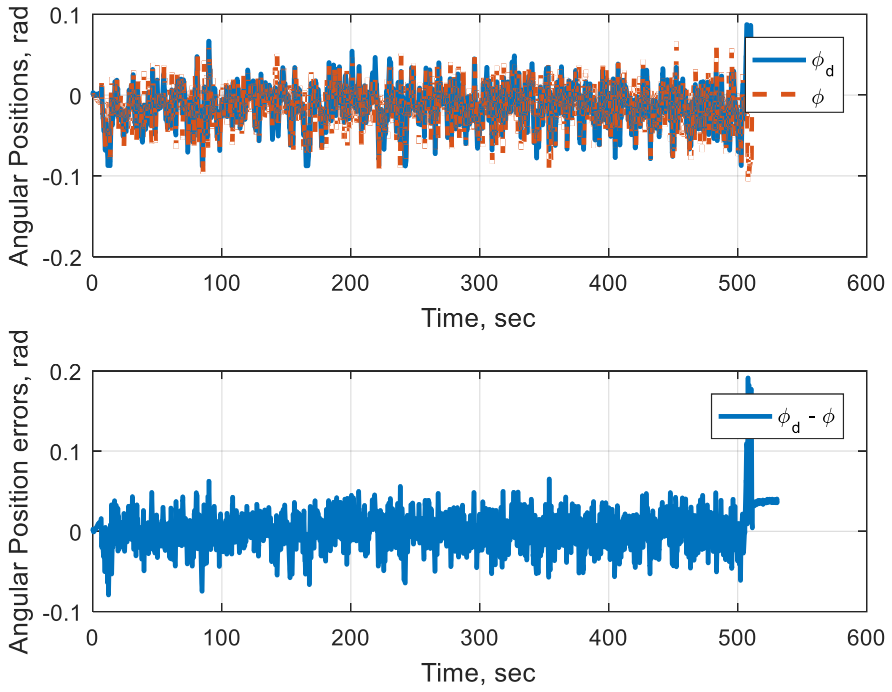


Figure 4.21 (c) Experimental Results for Case 2 Using ANN Controller: Roll Angle Response

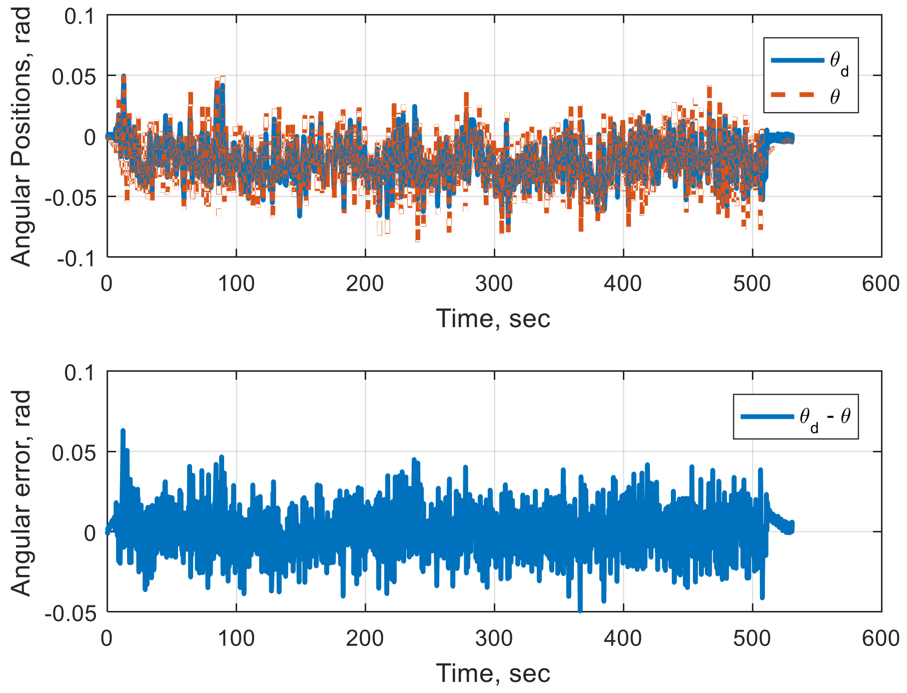


Figure 4.21 (d) Experimental Results for Case 2 Using ANN Controller: Pitch Angle Response

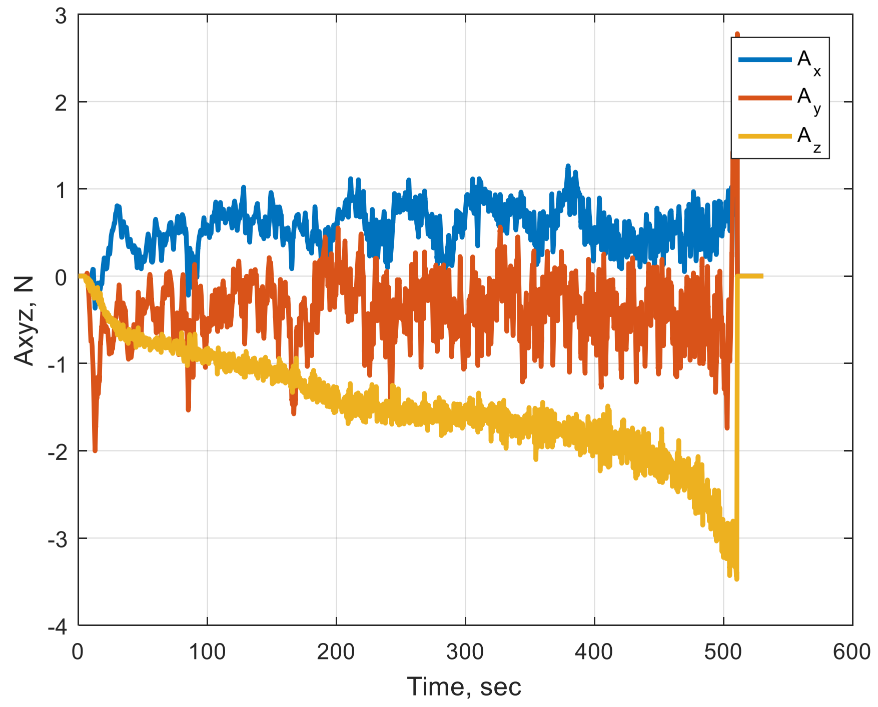


Figure 4.21 (e) Experimental Results for Case 2 Using ANN Controller: ANN Realized Forces

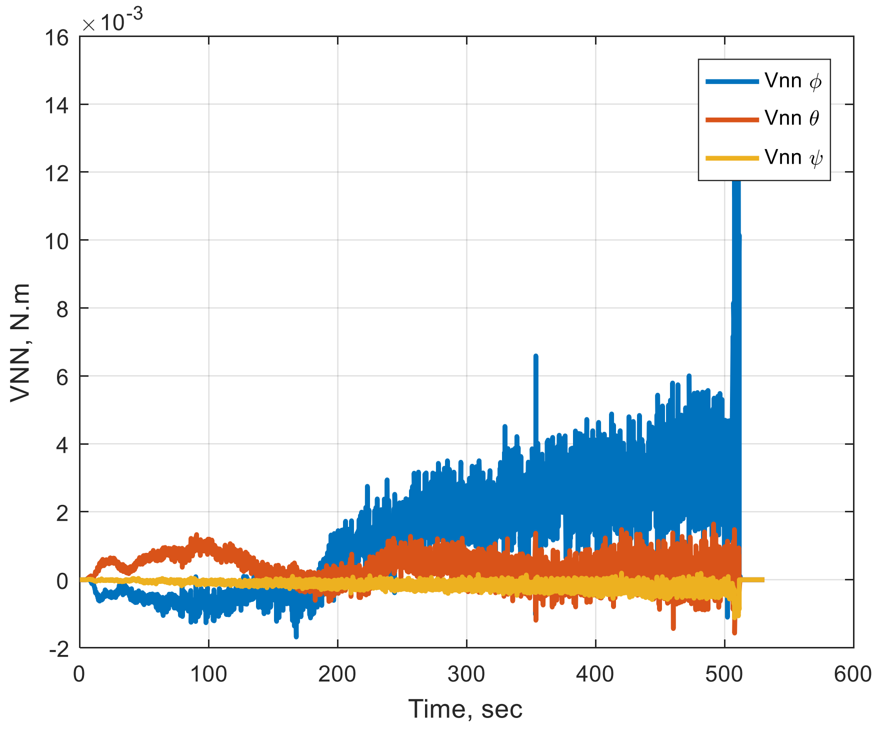


Figure 4.21 (f) Experimental Results for Case 2 Using ANN Controller: ANN Realized Moments

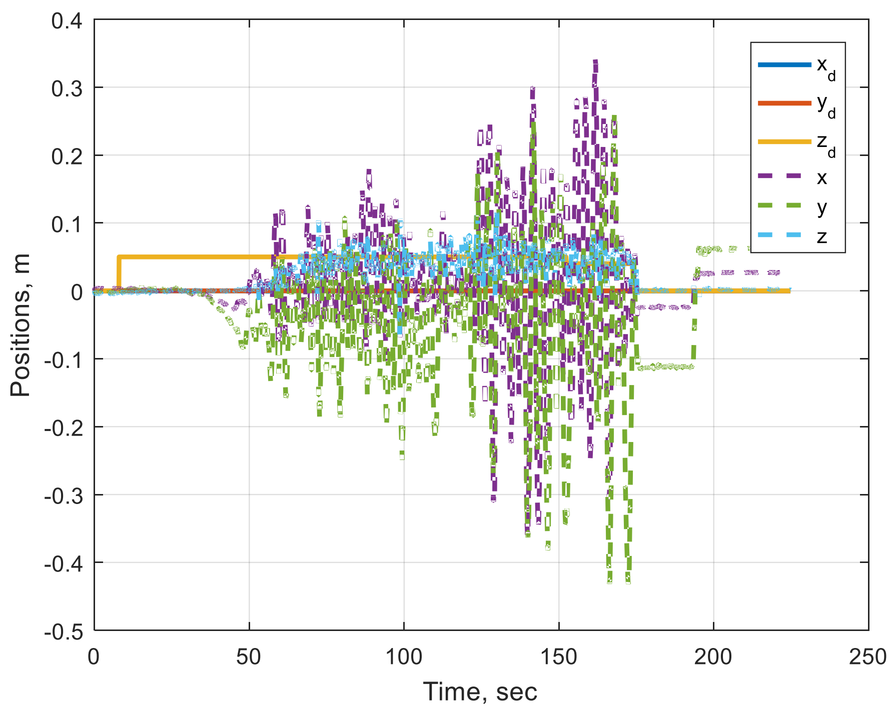


Figure 4.22 (a) Experimental Results for Case 2 Using PID Controller: Translational Response

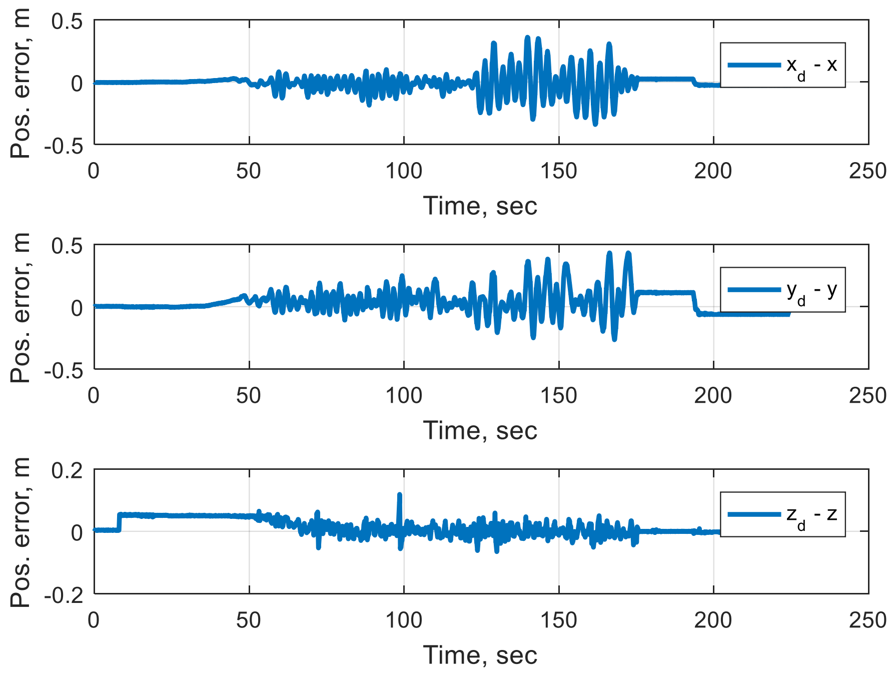


Figure 4.22 (b) Experimental Results for Case 2 Using PID Controller: Position Errors

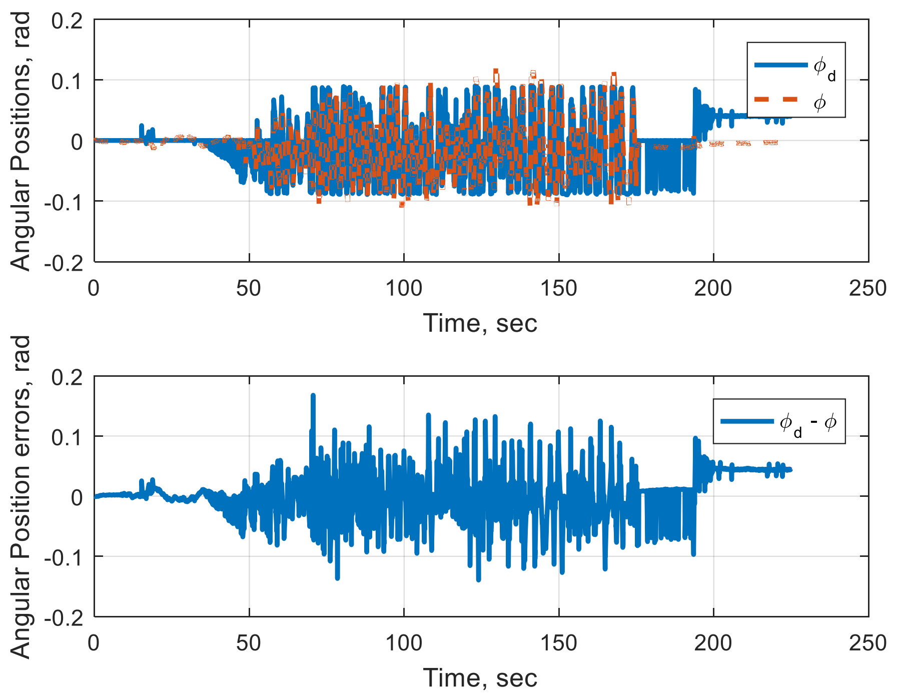


Figure 4.22 (c) Experimental Results for Case 2 Using PID Controller: Roll Angle Response

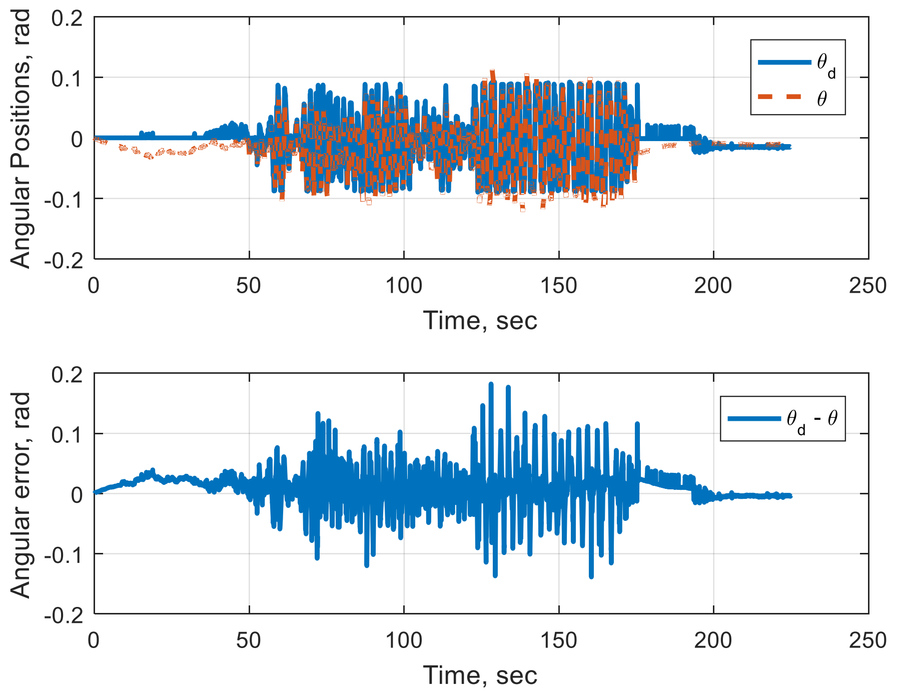


Figure 4.22 (d) Experimental Results for *Case* 2 U*sing* PID C*ontroller*: Pitch Angle Response

4.5.3 Corner of Two Walls

In this case, the Qball is operated in a corner of two walls as shown in Figure 4.23. Figures 4.24 and 4.25, show the hover response when flying in a low altitude. The ANN controller could maintain stability and compensate for aerodynamics uncertainties. This fact is reflected by the fewer oscillations on the X, Y directions when compared with that of the PID controller. It is seen too, that the ANN required attitude commands well below saturation limits to maintain position, while the PID controller required attitude angles beyond saturation limits. Notice that force and torque ANNs have captured different behaviors for aerodynamics uncertainties than in previous cases. The smaller X force is an indication that the wall behind the quadrotor is applying a negative force that balanced, to some extent, the positive force caused by CG shift and thrust asymmetry which was identified in previous runs. Also, it is seen that the identified Y force has less fluctuations compared with the one wall case. Figure 4.24 (f) shows how the pitch torque in this case has a less negative value compared with the no wall case, as a result of the back wall. On the other hand, roll torque agrees with that of the one wall case, and is in a different direction compared with no wall case. Notice that in the case of PID attitude controller, Figure 4.25 (c) and (d), the roll command saturates more on the negative side, away from the wall, which indicates that the Qball is constantly drifting towards the wall. While the pitch command is almost balanced, as the CG shift which is pushing the Qball forward balances the negative force applied by the back wall which tends to pull the Qball backwards.

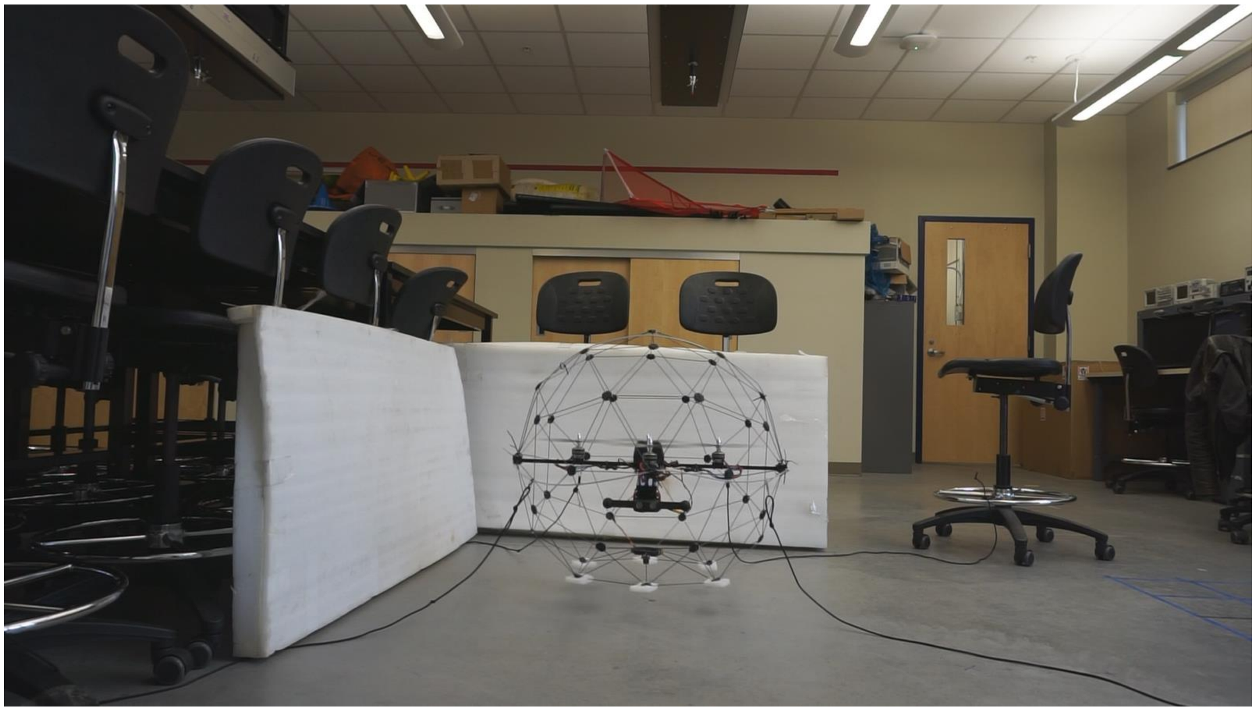


Figure 4.23 Environment Setup for Case 3: Corner of Two Walls

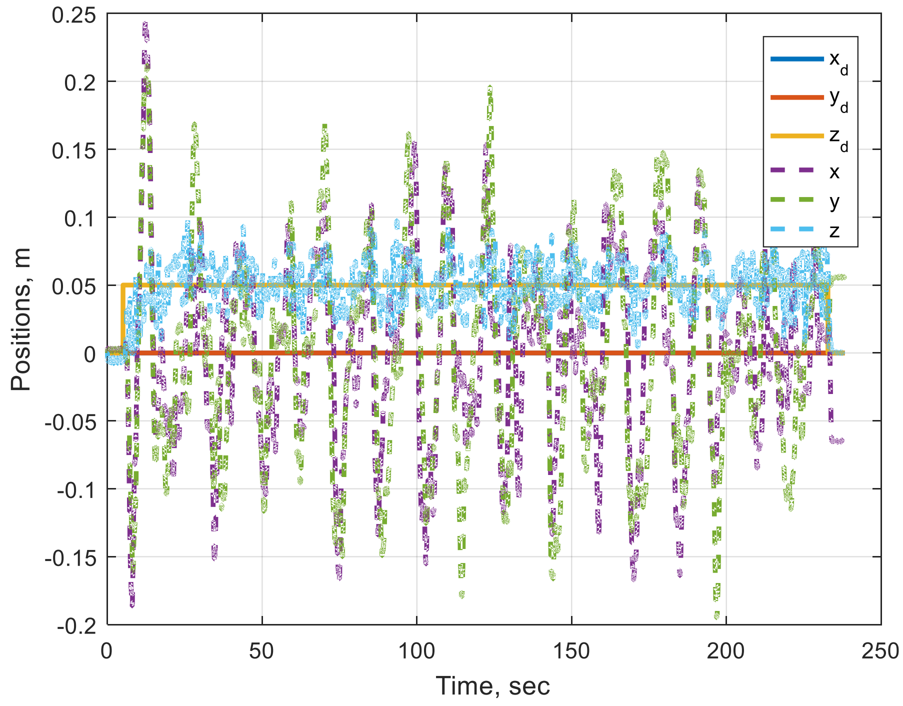


Figure 4.24 (a) Experimental Results for Case 3 Using ANN Controller: Translational Response

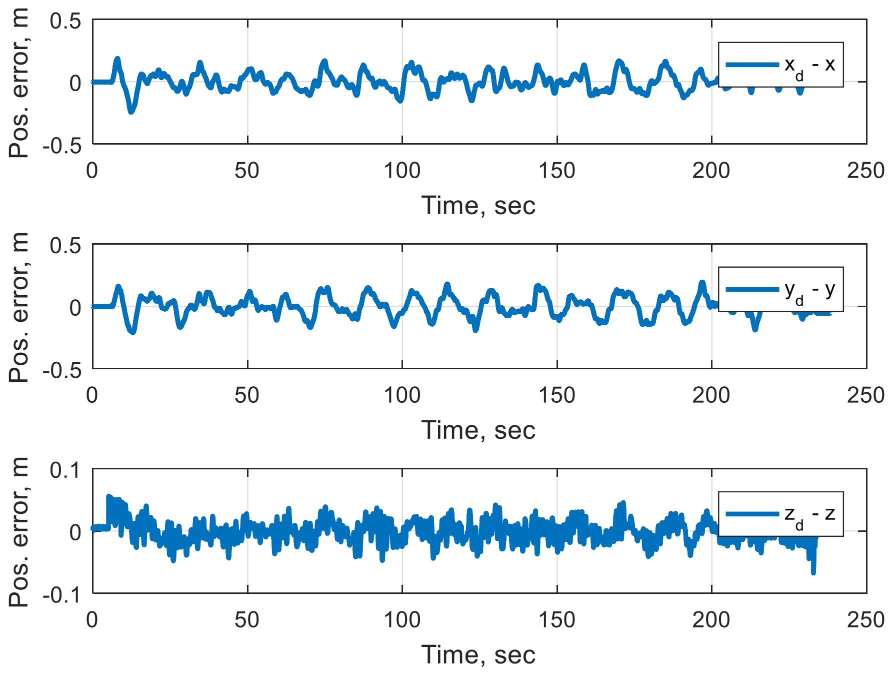


Figure 4.24 (b) Experimental Results for Case 3 Using ANN Controller: Position Errors

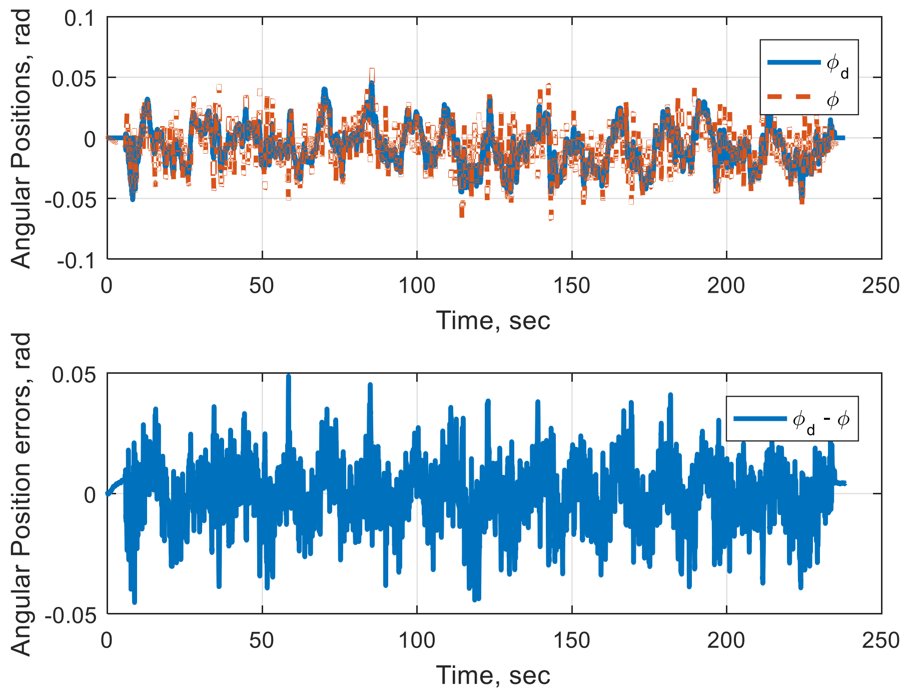


Figure 4.24 (c) Experimental Results for Case 3 Using ANN Controller: Roll Angle Response

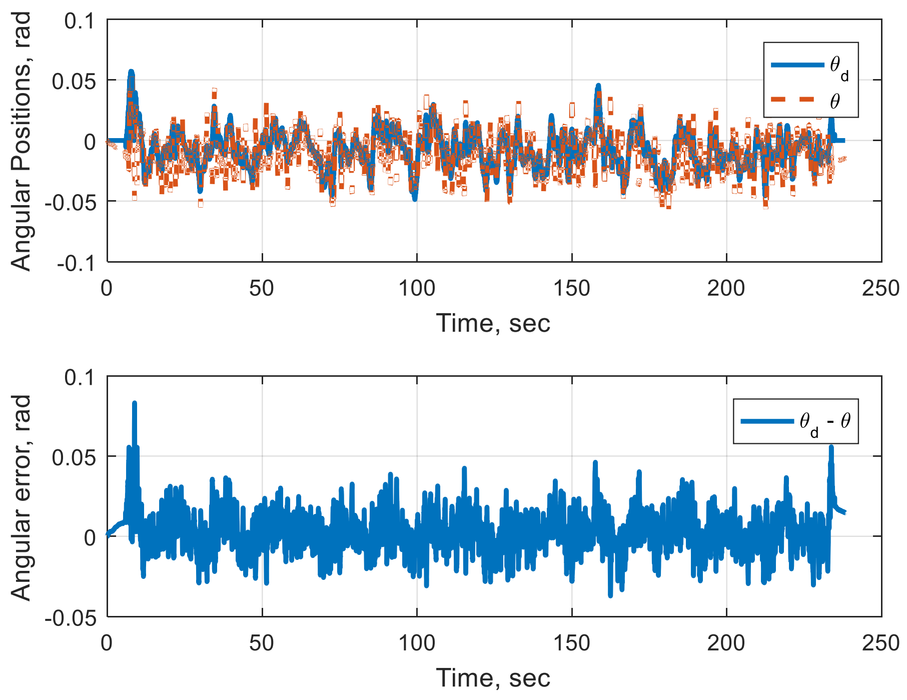


Figure 4.24 (d) Experimental Results for Case 3 Using ANN Controller: Pitch Angle Response

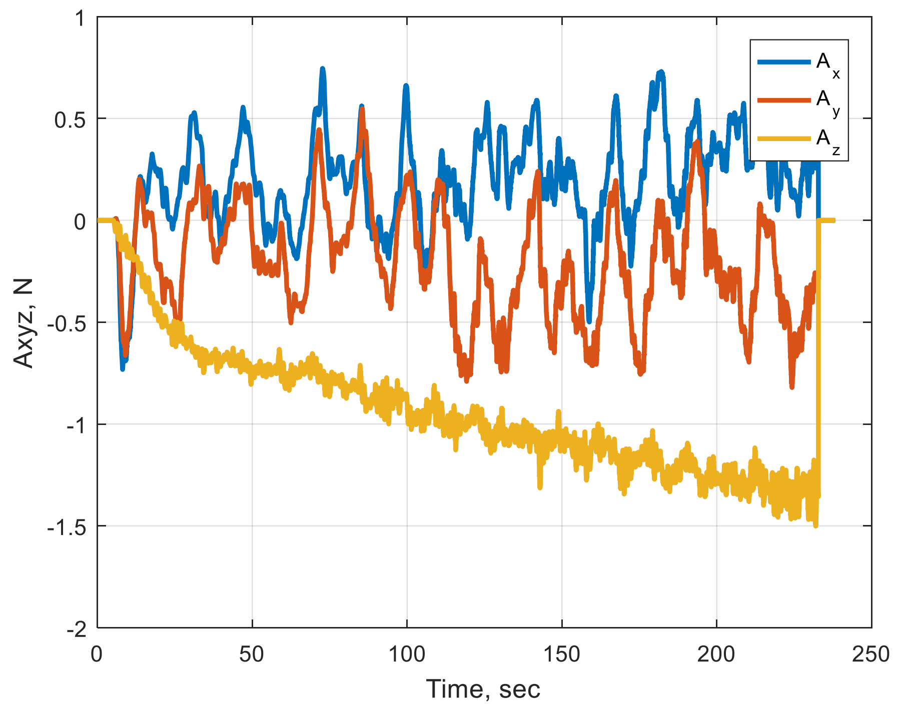


Figure 4.24 (e) Experimental Results for Case 3 Using ANN Controller: ANN Realized Forces

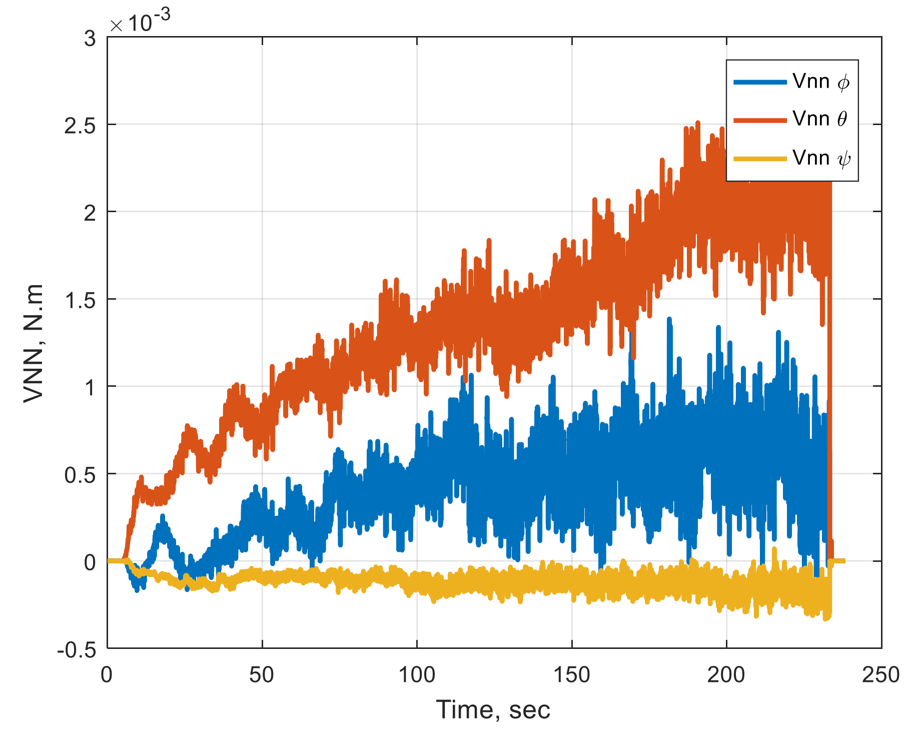


Figure 4.24 (f) Experimental Results for Case 3 Using ANN Controller: ANN Realized Moments

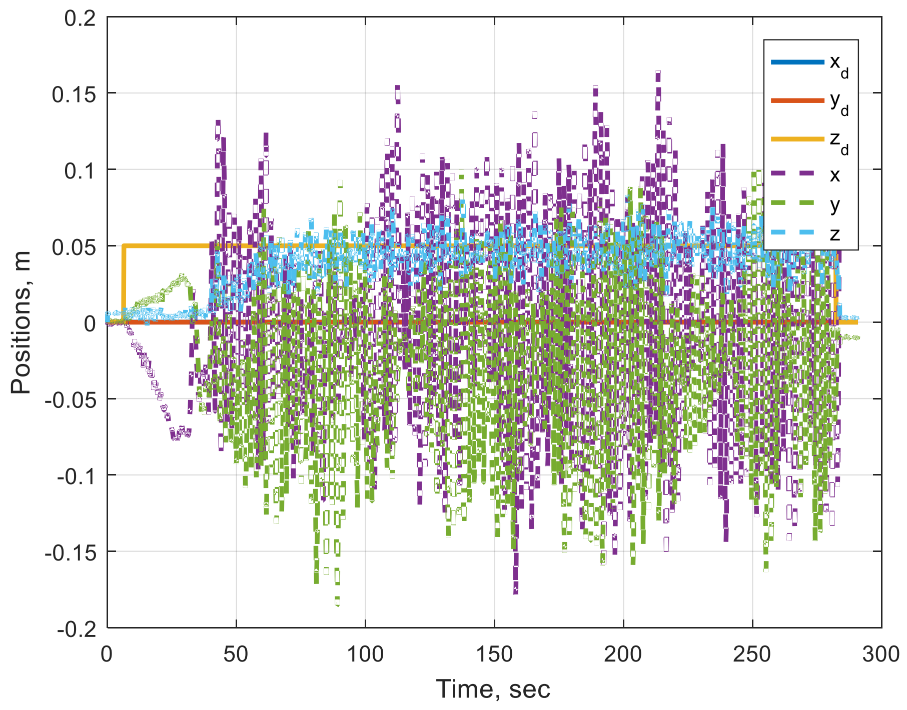


Figure 4.25 (a) Experimental Results for Case 3 Using PID Controller: Translational Response

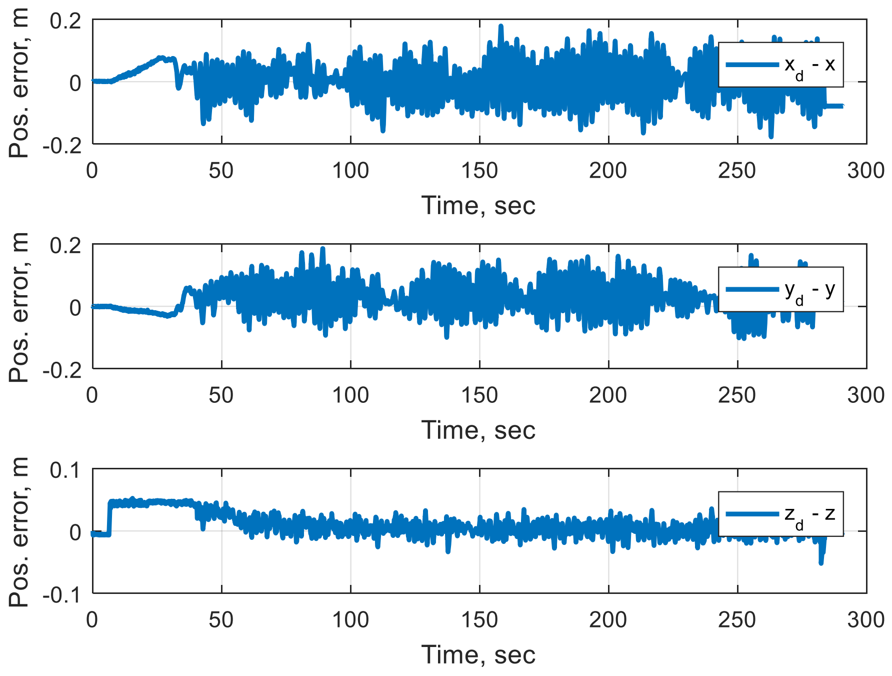


Figure 4.25 (b) Experimental Results for Case 3 Using PID Controller: Position Errors

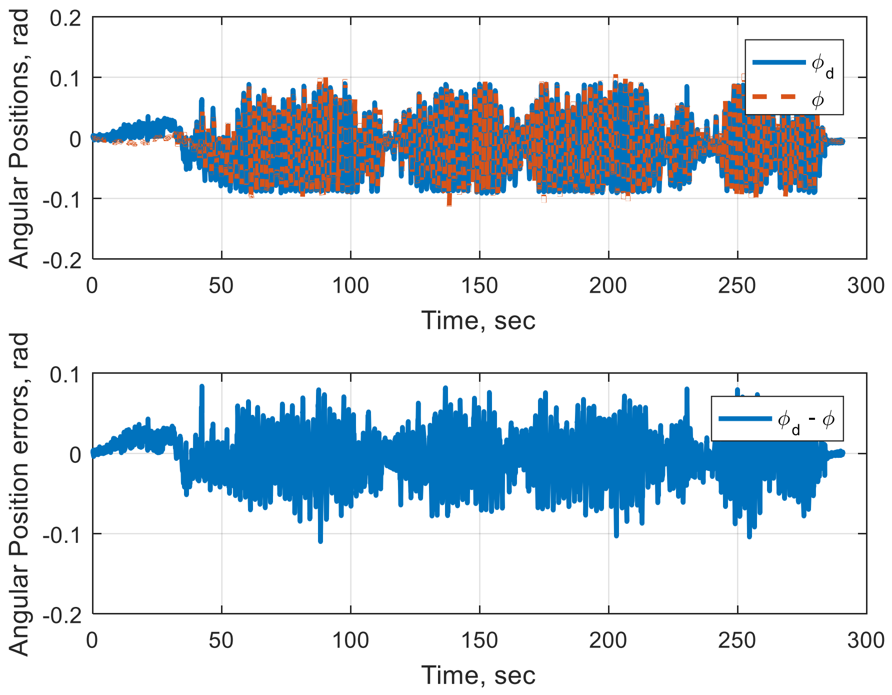


Figure 4.25 (c) Experimental Results for Case 3 Using PID Controller: Roll Angle Response

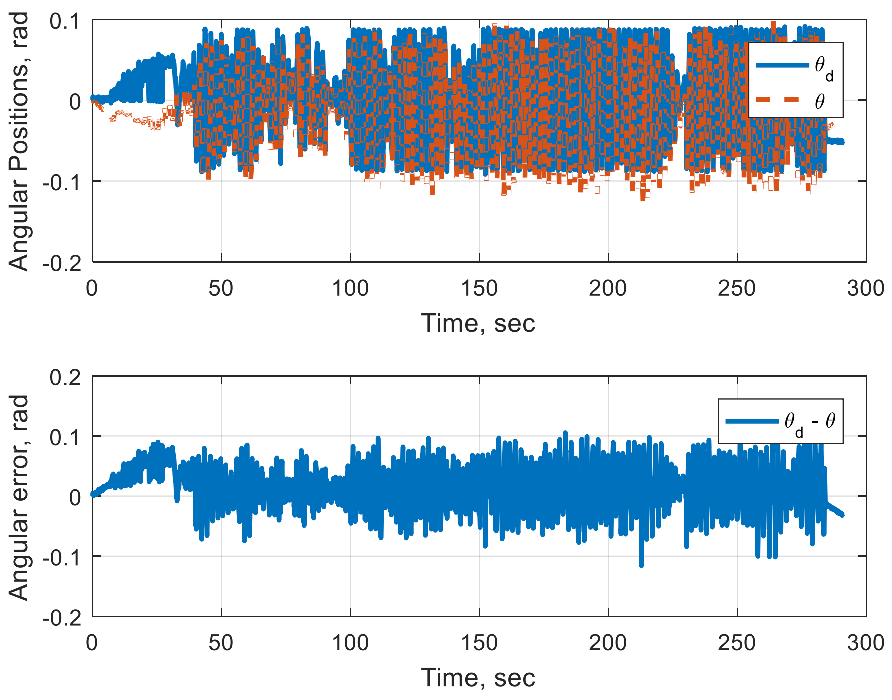


Figure 4.25 (d) Experimental Results for Case 3 Using PID Controller: Pitch Angle Response

4.5.4 Between Chairs (Typical Confined Environment)

The environment setup in this case is shown in Figure 4.26. The Qball is commanded to hover at different altitudes starting at a very low value until it flies higher than the height of the chairs. This case is considered as a typical confined environment, because it represents the case when the quadrotor is flying around objects that are not necessarily solid or symmetric. As a result, air is expected to behave in chaotic nature.

Figures 4.27 and 4.28 show the time response of the Qball when controlled by the ANNs and the PID respectively. Performance is generally very good in both cases. It is seen that the force ANN has captured an X component that is shifted forward, a Y component that is fluctuating about the zero, and a Z component that looks similar to previous cases, this is a very good indication for ANNs consistency. X and Y forces are fluctuating with bigger magnitudes compared with previous runs with a clear irregularity, associated with irregularity of the environment. Notice that torques identified by the ANN change with altitude; which is expected as a result of the change of chairs geometry with altitude.

Notice too, that the PID controller is trapped in a limit cycle to maintain position, with attitude commands saturating back and forth.

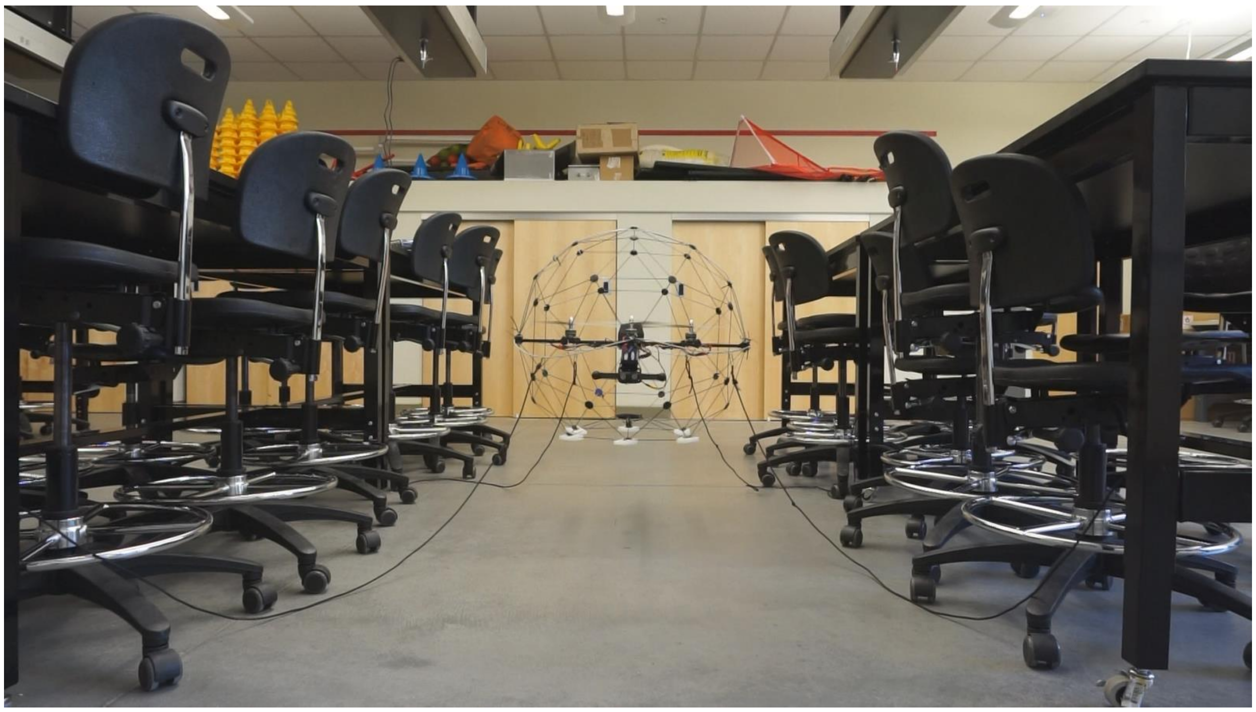


Figure 4.26 Environment Setup for Case 4: Middle of Chair Rows

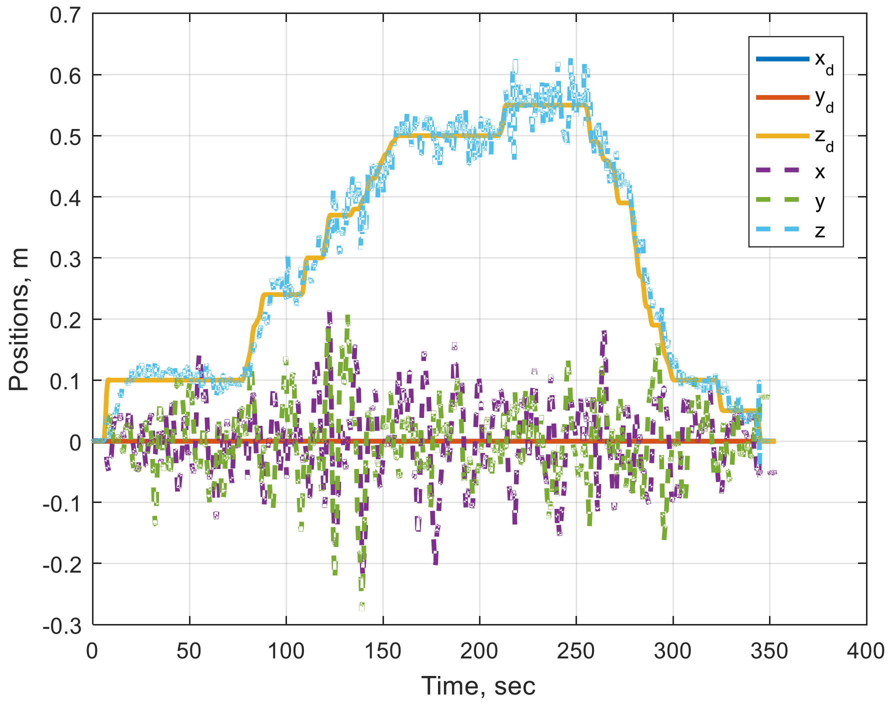


Figure 4.27 (a) Experimental Results for Case 4 Using ANN Controller: Translational Response

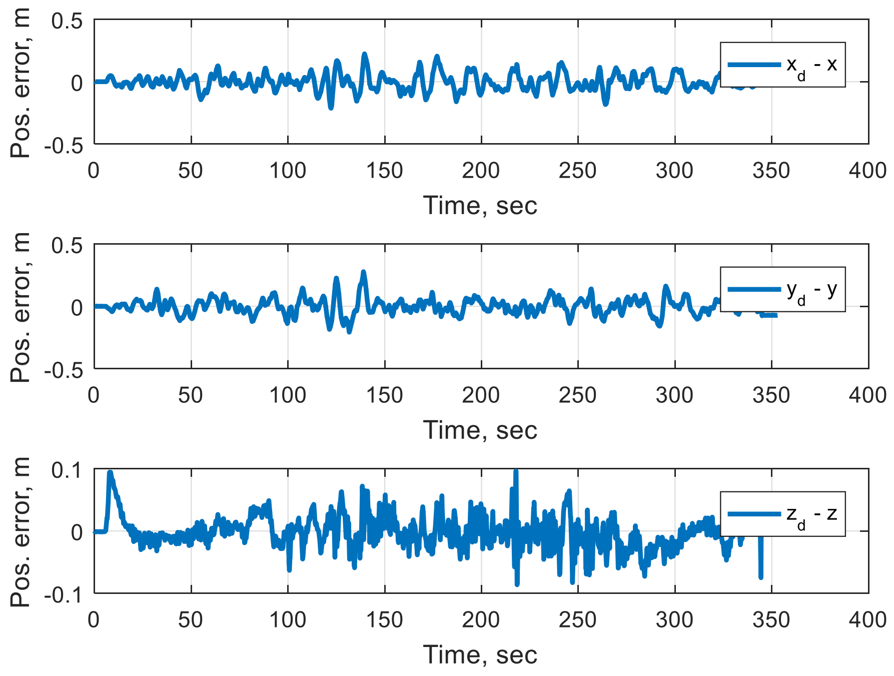


Figure 4.27 (b) Experimental Results for Case 4 Using ANN Controller: Position Errors

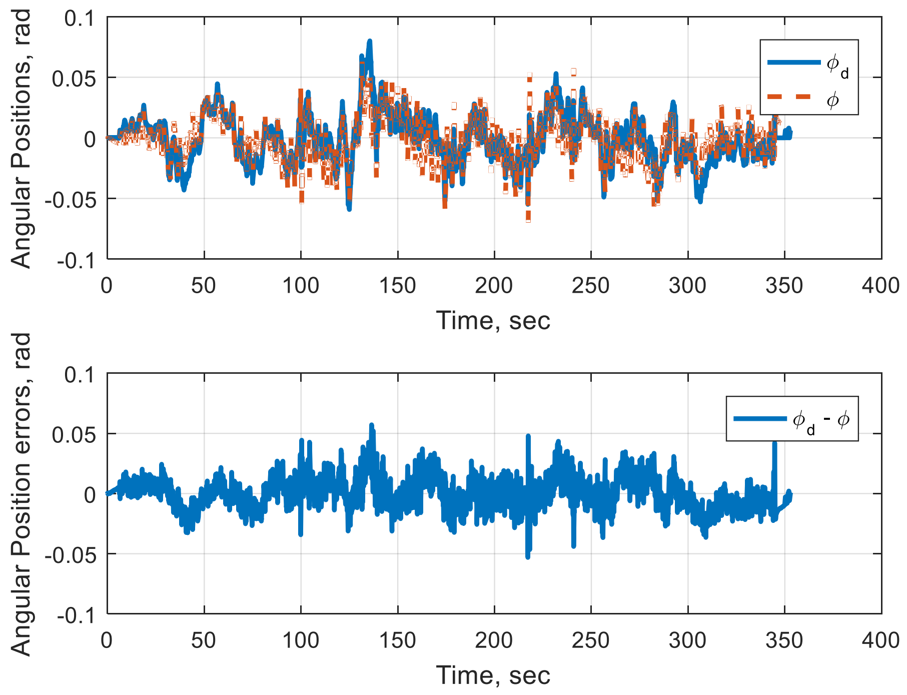


Figure 4.27 (c) Experimental Results for Case 4 Using ANN Controller: Roll Angle Response

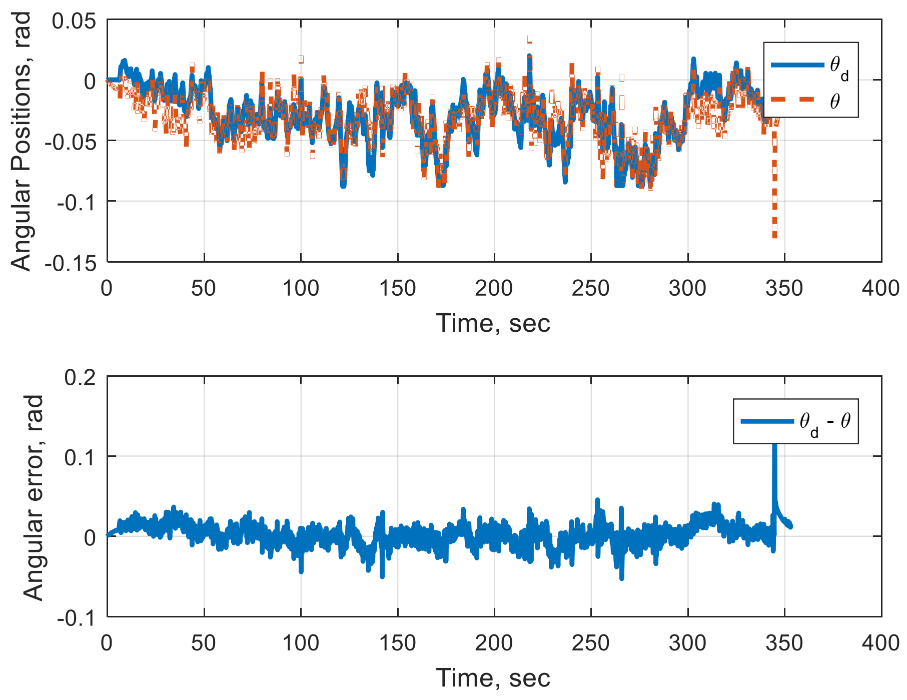


Figure 4.27 (d) Experimental Results for Case 4 Using ANN Controller: Pitch Angle Response

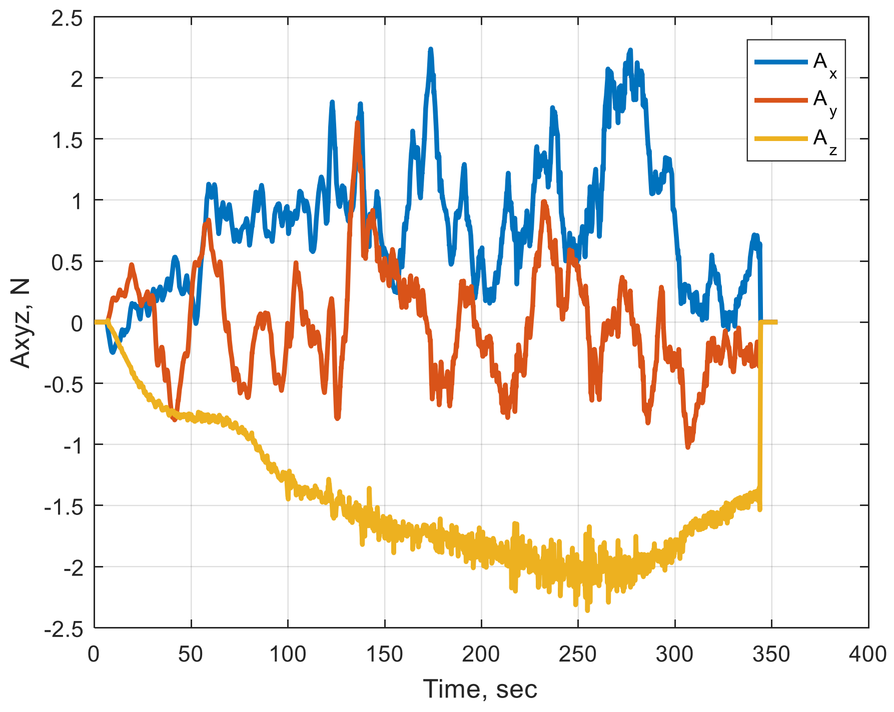


Figure 4.27 (e) Experimental Results for Case 4 Using ANN Controller: ANN Realized Forces

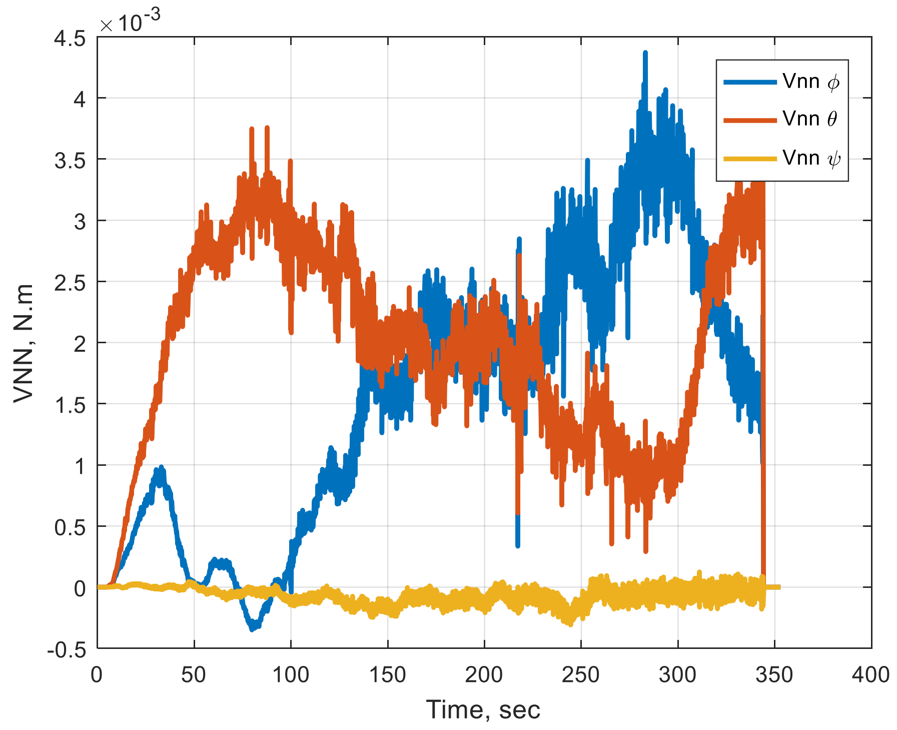


Figure 4.27 (f) Experimental Results for Case 4 Using ANN Controller: ANN Realized Moments

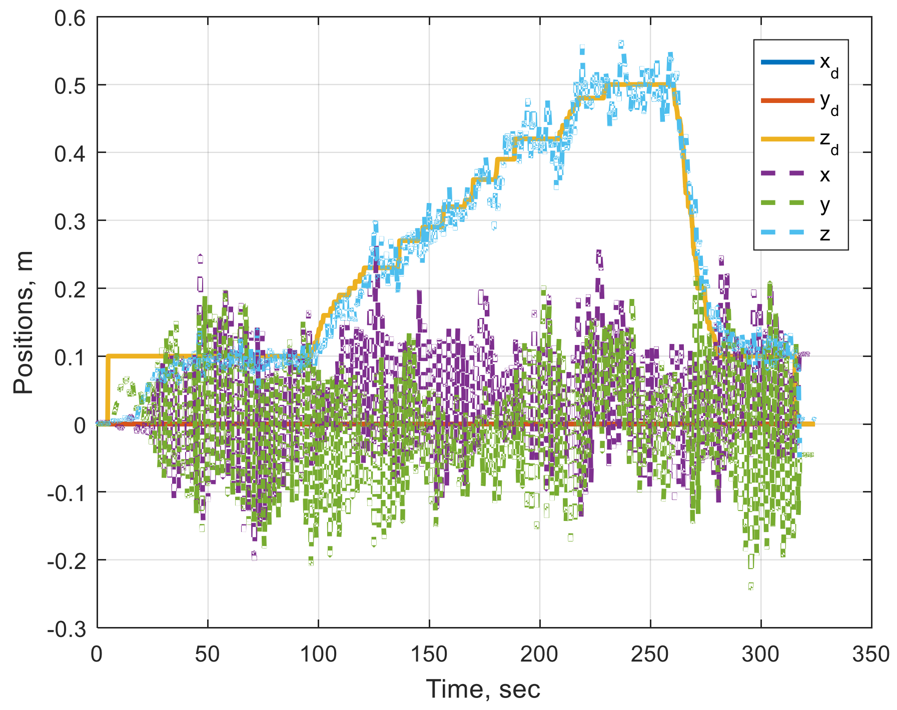


Figure 4.28 (a) Experimental Results for Case 4 Using PID Controller: Translational Response

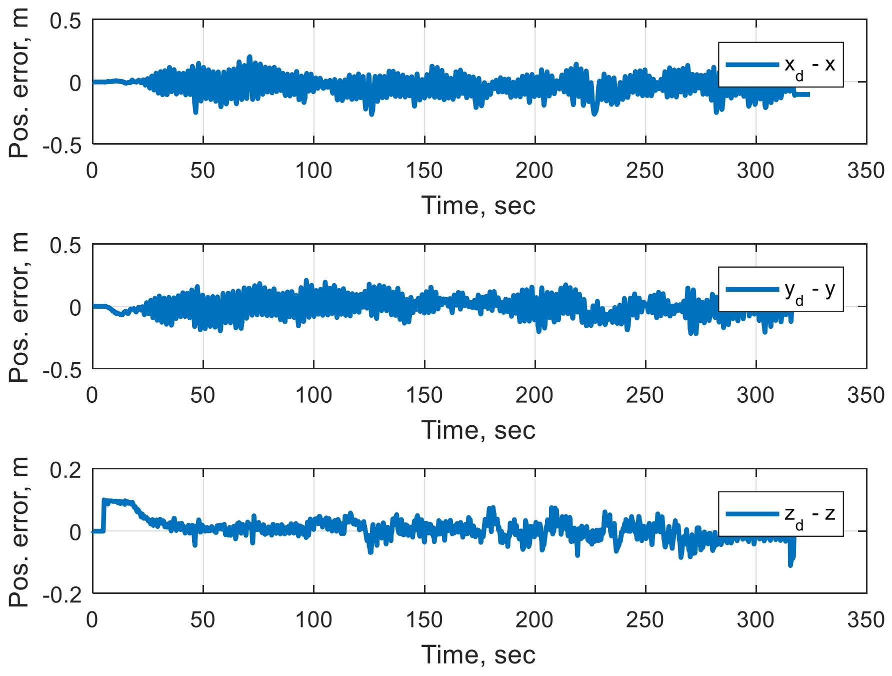


Figure 4.28 (b) Experimental Results for Case 4 Using PID Controller: Position Errors

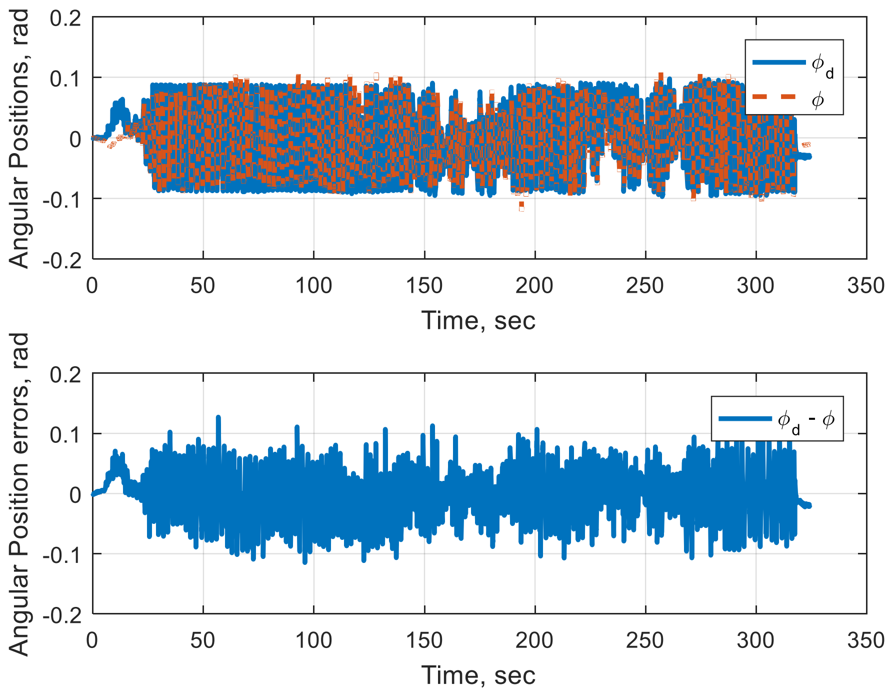


Figure 4.28 (c) Experimental Results for Case 4 Using PID Controller: Roll Angle Response

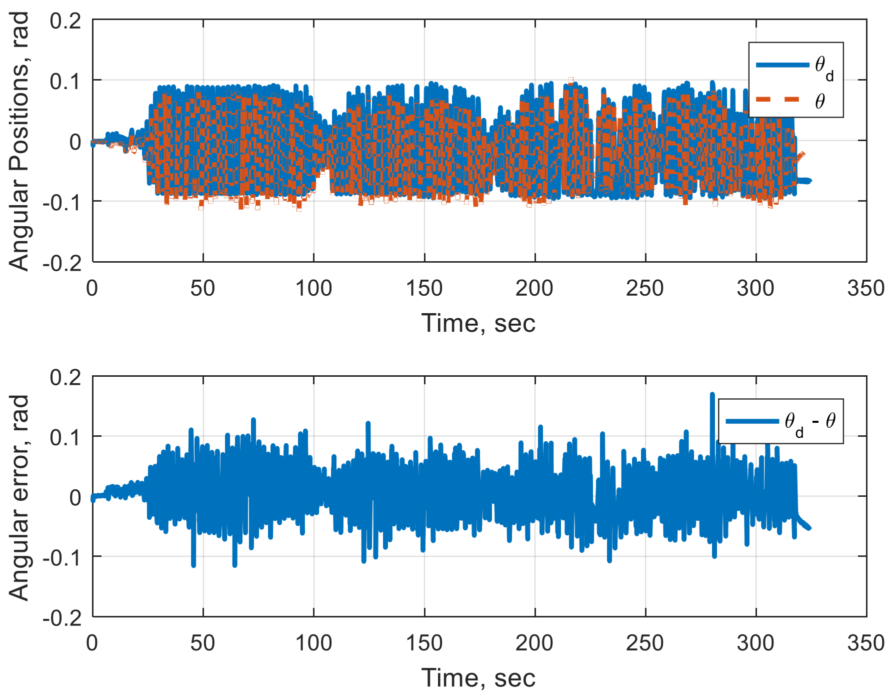


Figure 4.28 (d) Experimental Results for Case 4 Using PID Controller: Pitch Angle Response

4.5.5 Tunnel-Like Environment

This is the most challenging setup performed in this research. Challenges faced were far beyond controller adaptation and uncertainty compensation. The main difficulty was associated with Computer Vision, it was very difficult to create a tunnel path and have the SLAM sensor work properly at the same time. Two problems arose: Light level, and view repetition. Because of those two problems position measurement jump abruptly between values with significant difference, up to meters, which lead to the controller trying to compensate for them and crash the Qball. Figure 4.29 shows the final setup for this case: it is similar to that of the case of *Between-The-Chairs* with the addition of foam boards on the sides of part of the passage. This setup turned out to be a very good scenario, as flight performance could be tested when entering, and exiting the tunnel as well as inside the tunnel. The Qball is first lifted off inside the tunnel (between the boards), let to hover for a while, then an X command is applied to send the Qball away from the boards, and then sent back to the tunnel again.

Figures 4.30 and 4.31 show the response in this setup when using the ANN and PID controllers respectively. This case showed a significant difference between the two controllers. When the PID is used, the Qball could not maintain the X position and was oscillating back and forth with a large magnitude, until it surpasses the boards. However, the Y response looked better because the Qball was not maintained inside the tunnel. When the Qball is sent away from the boards the X response improved, this is a sign of the significant influence of the aerodynamics associated with the tunnel. The Qball crashed on the side of the tunnel at the end of the PID run. The ANN controller was able to maintain the Qball inside the tunnel, compensate for the aerodynamic forces on the X, Y and Z directions, and improve performance with time. Notice the change of ANN forces when the Qball is out of the tunnel, and more importantly, how they maintained their identified functions when quickly sent back in the tunnel. This shows that ANNs provide function estimation rather than instant adaptation for changes, such is a manifestation of the difference between learning and adaptation.

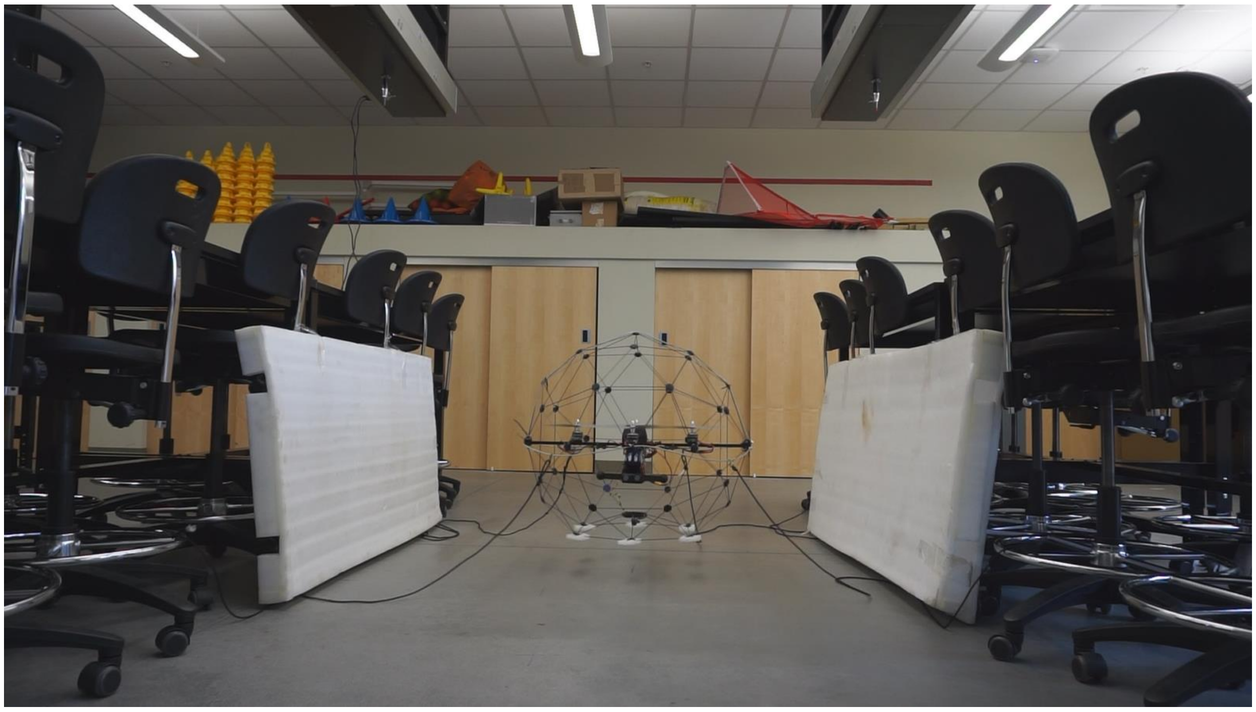


Figure 4.29 Environment Setup for Case 5: Tunnel-Like

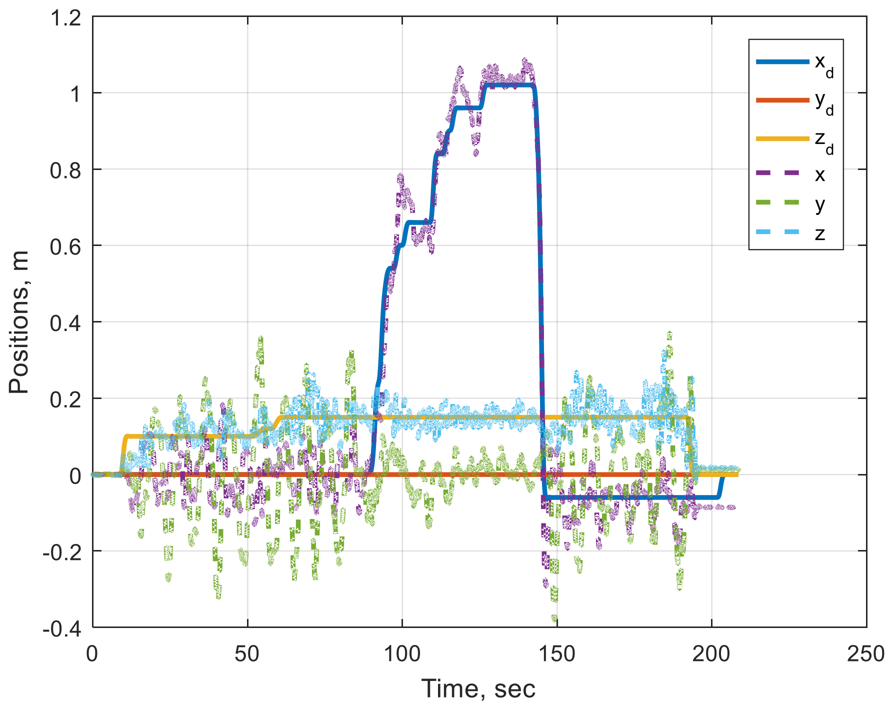


Figure 4.30 (a) Experimental Results for Case 5 Using ANN Controller: Translational Response

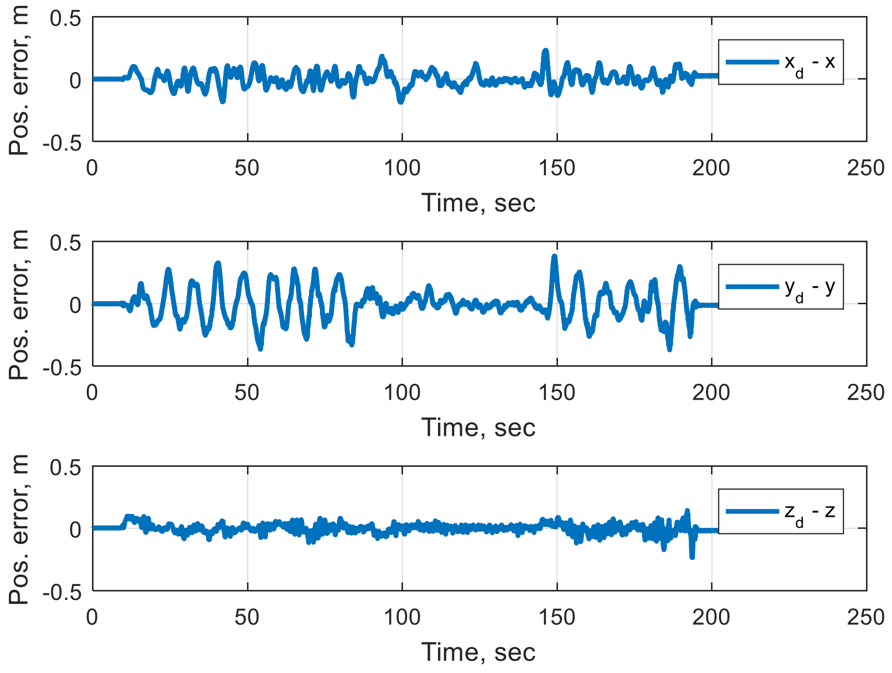


Figure 4.30 (b) Experimental Results for Case 5 Using ANN Controller: Position Errors

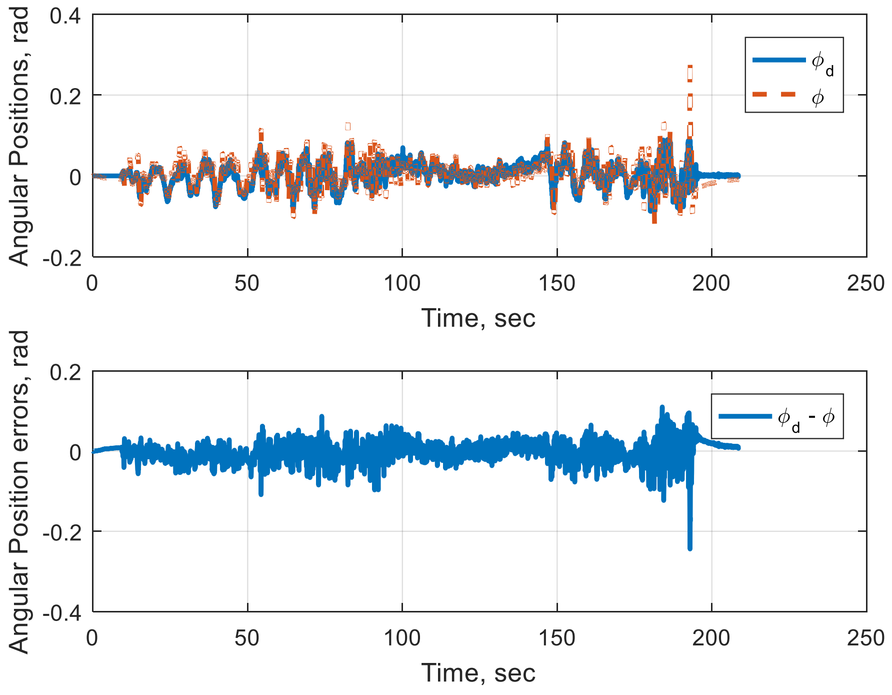


Figure 4.30 (c) Experimental Results for Case 5 Using ANN Controller: Roll Angle Response

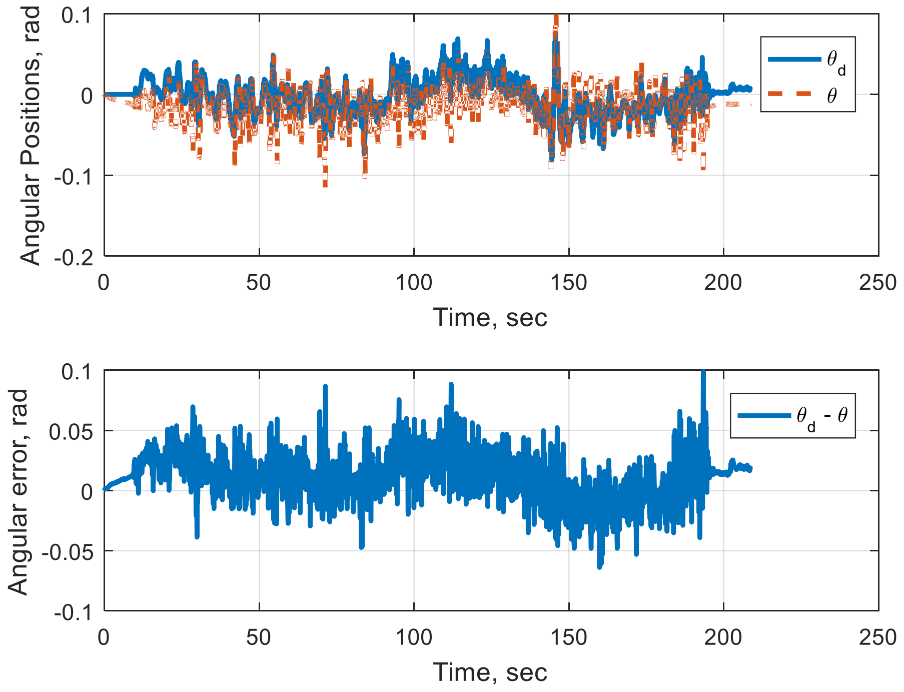


Figure 4.30 (d) Experimental Results for Case 5 Using ANN Controller: Pitch Angle Response

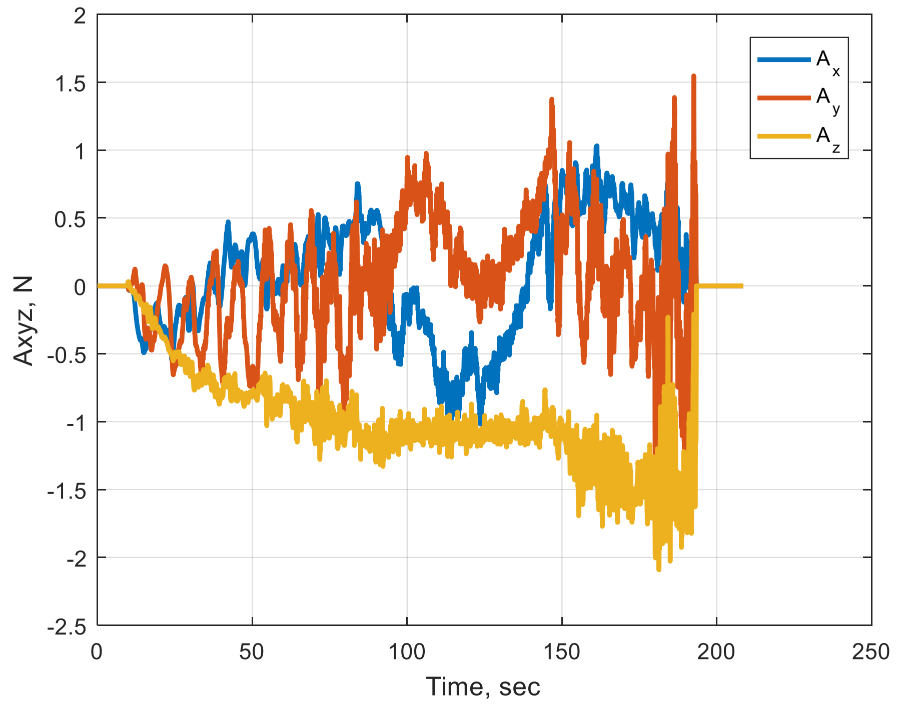


Figure 4.30 (e) Experimental Results for Case 5 Using ANN Controller: ANN Realized Forces

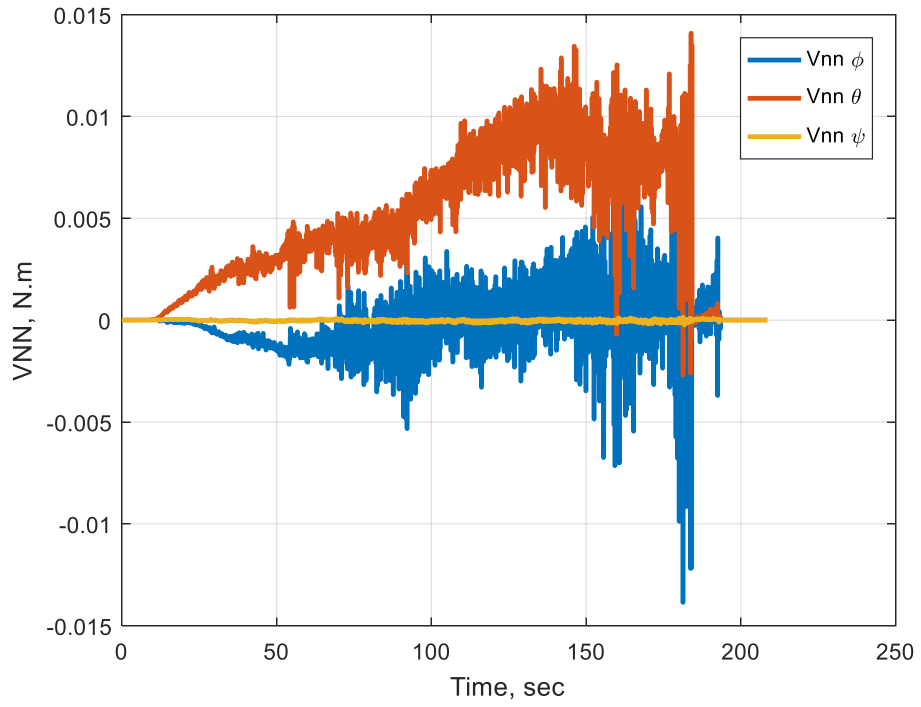


Figure 4.30 (f) Experimental Results for Case 5 Using ANN Controller: ANN Realized Moments

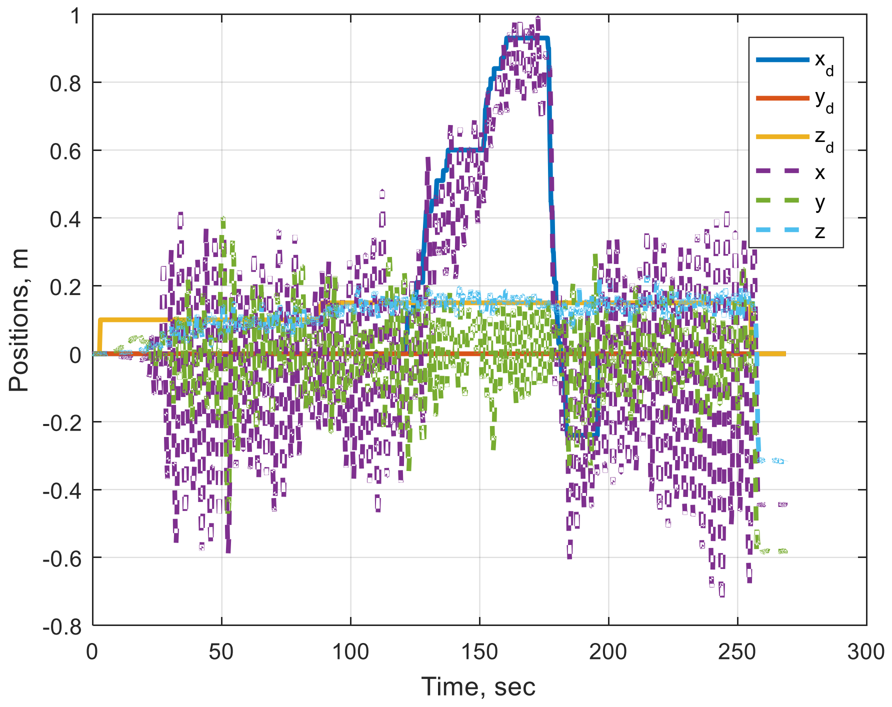


Figure 4.31 (a) Experimental Results for Case 5 Using PID Controller: Translational Response

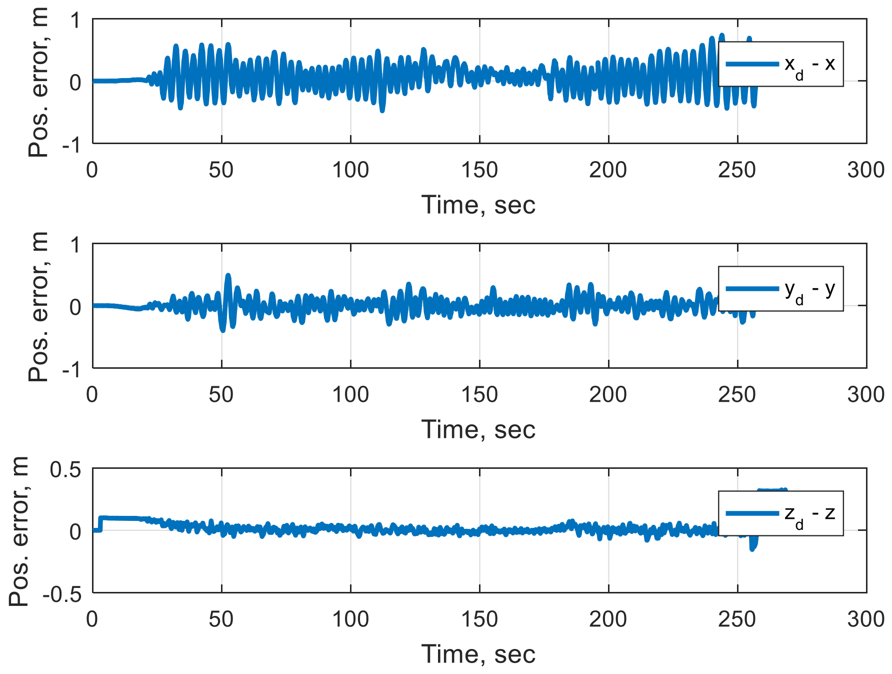


Figure 4.31 (b) Experimental Results for Case 5 Using PID Controller: Position Errors

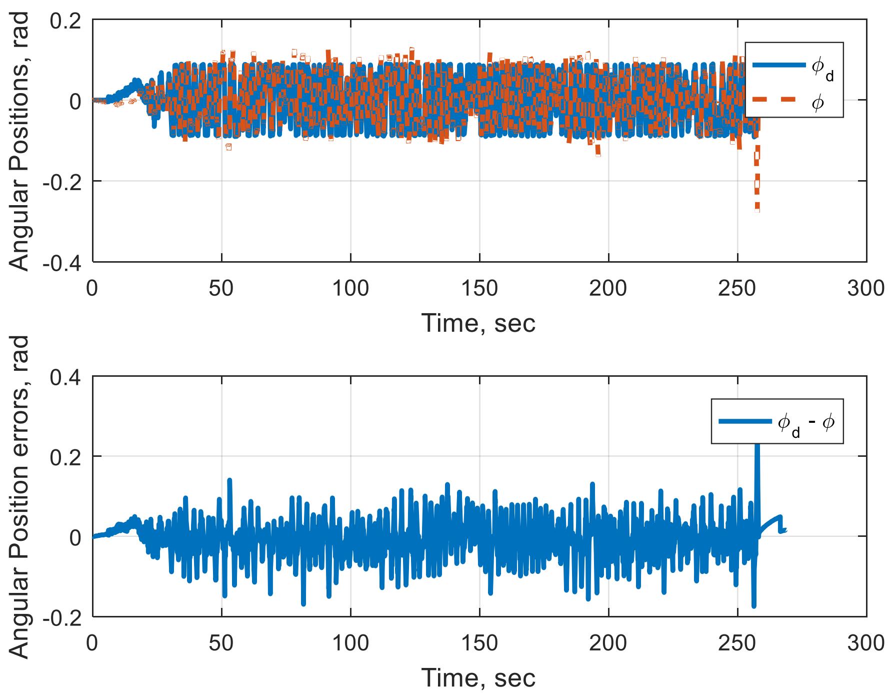


Figure 4.31 (c) Experimental Results for Case 5 Using PID Controller: Roll Angle Response

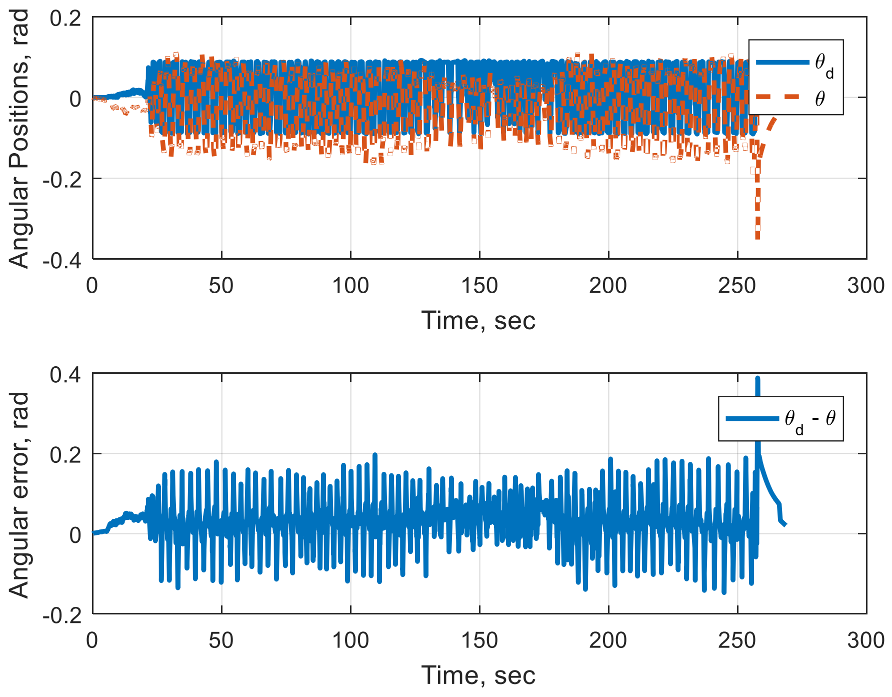


Figure 4.31 (d) Experimental Results for Case 5 Using PID Controller: Pitch Angle Response

4.5.6 Added Actuator Dynamics

It has been observed in previous runs that there are visible ‘cyclic’ fluctuations on the Z response. These fluctuations are partly caused by fluctuations in position measurement, associated with trajectory sensory, and partly by the actuator being slower than the controller. The ANN controller commands forces and expects propellers to be able to instantly deliver these forces. It is well known in the field of nonlinear systems that such delays (nonlinearities) cause limit cycle behavior [90]. The actuator dynamics for the Qball are experimentally identified by Quanser and the model is provided in the Qball manual, Table 4-2. Therefore, it is decided to include actuator dynamics in the controller module to fix these fluctuations. Figures 4.33 shows the time response for the case of *between the chairs* with the inclusion of actuator dynamics, Figure 4.32. Notice the significant reduction in altitude fluctuations.

It is important to mention that the drawback of including actuator model in the altitude controller is that it makes altitude dynamics very responsive and, as a result, very fragile for position sensory glitches. In several occasions when the SLAM sensor loses track of position it holds the previous ‘known’ value until it retrieves position information then a sudden change in altitude happens, and accordingly, the Qball reacts very quickly and crashes. To avoid that, some safety features are added to the ANN controller module.

The general performance in this case is considered satisfactory.

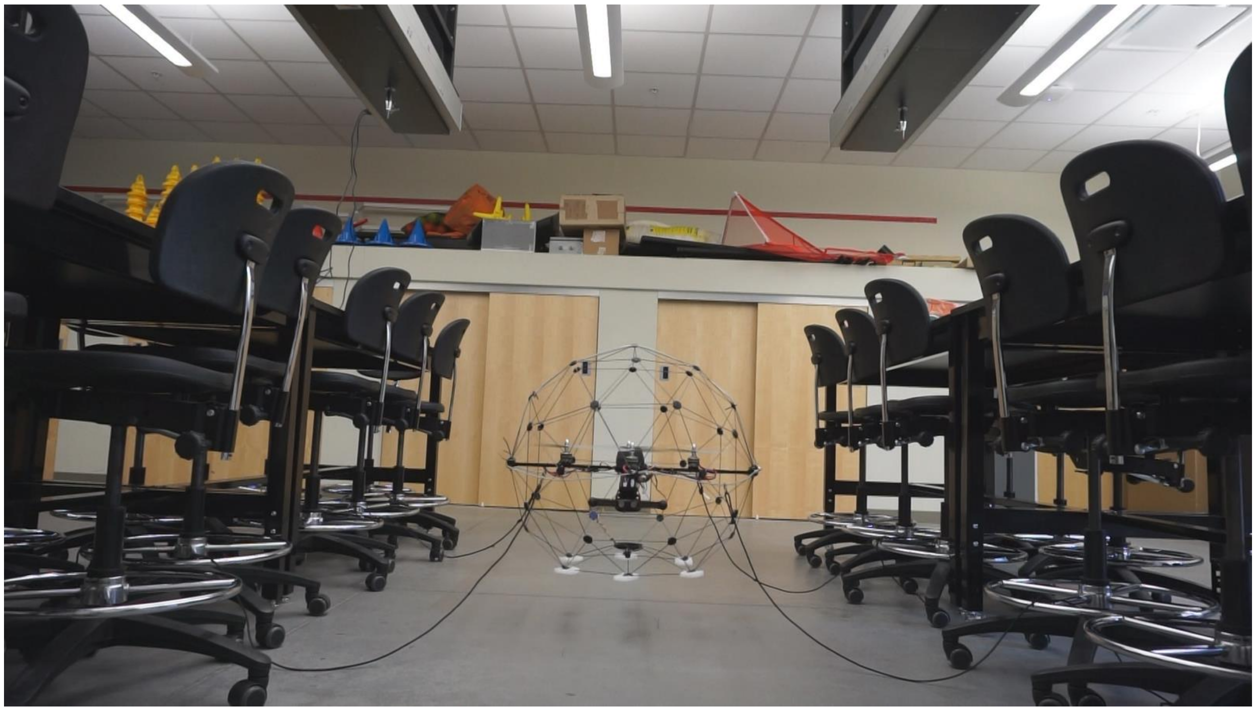


Figure 4.32 Environment Setup for Case 6: Added Actuator Dynamics

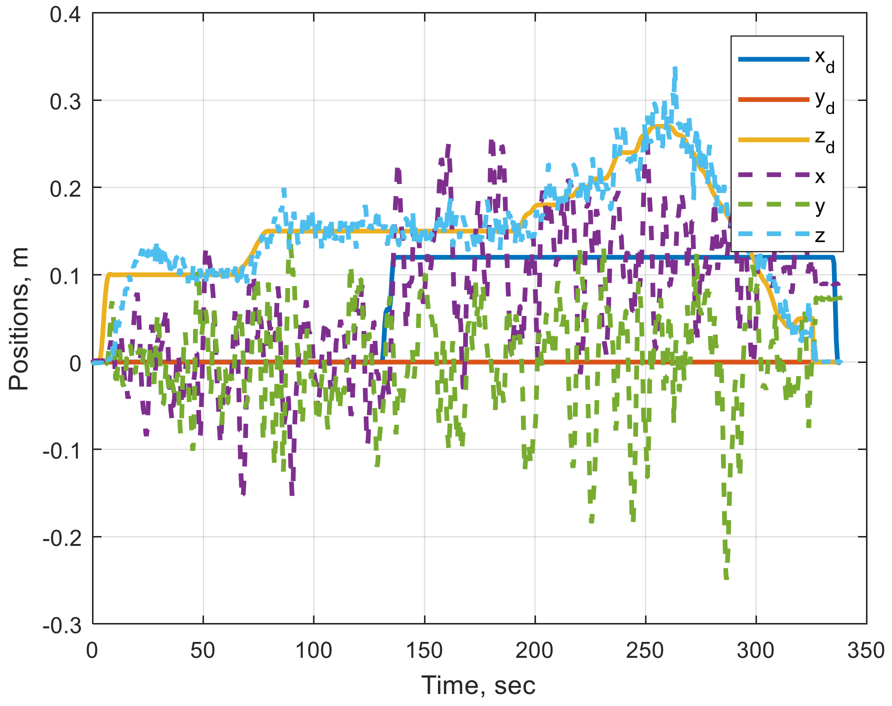


Figure 4.33 (a) Experimental Results for Case 6 Using ANN Controller: Translational Response

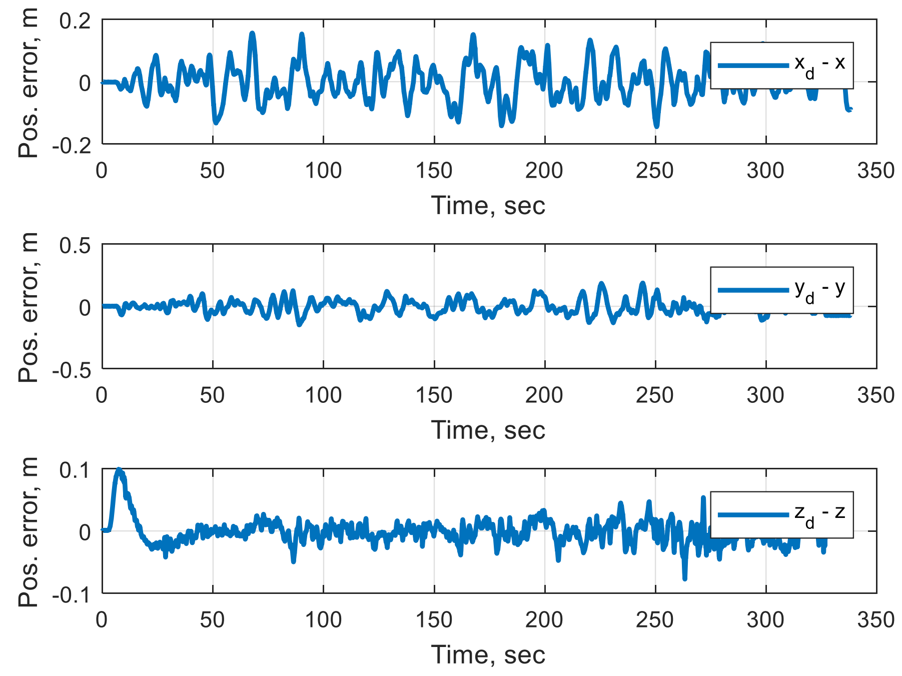


Figure 4.33 (b) Experimental Results for Case 6 Using ANN Controller: Position Errors

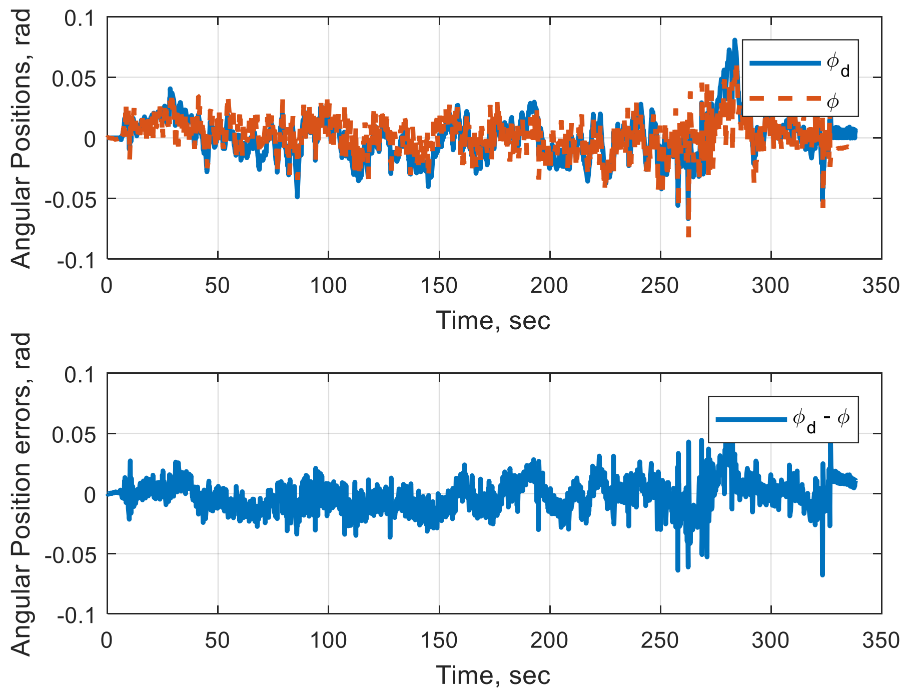


Figure 4.33 (c) Experimental Results for Case 6 Using ANN Controller: Roll Angle Response

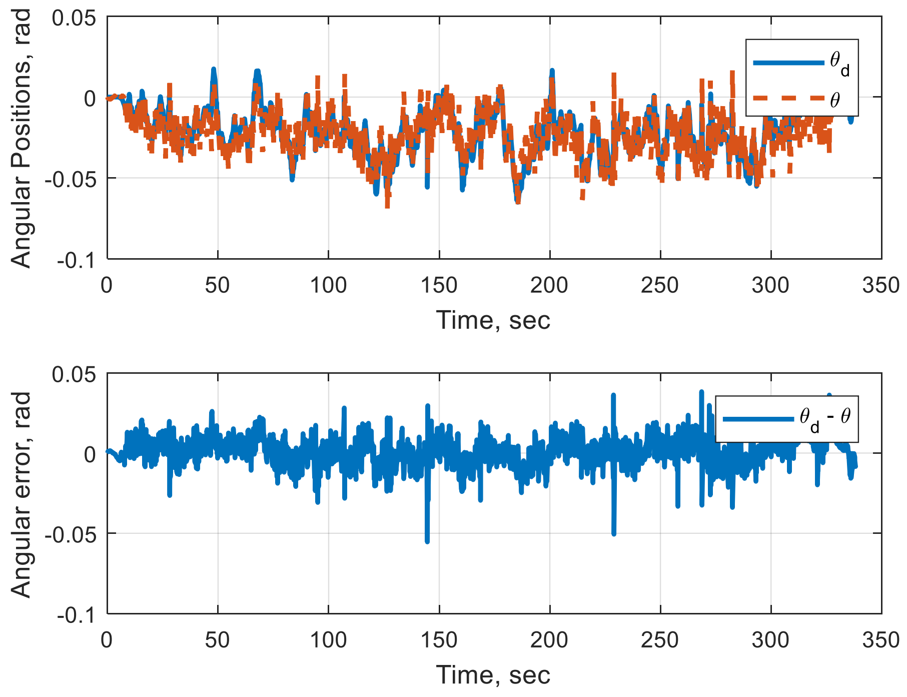


Figure 4.33 (d) Experimental Results for Case 6 Using ANN Controller: Pitch Angle Response

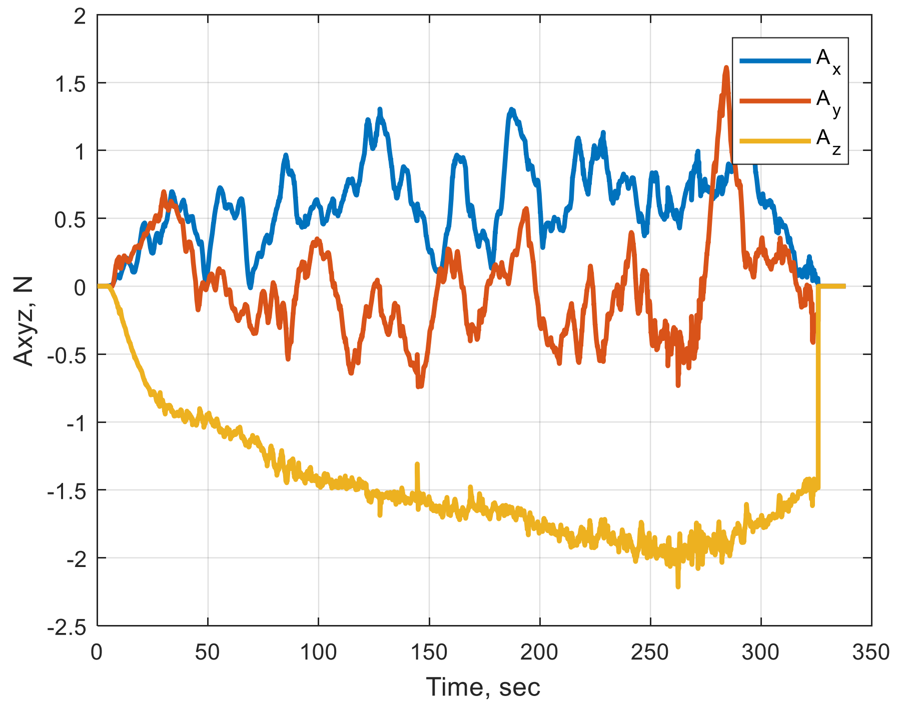


Figure 4.33 (e) Experimental Results for Case 6 Using ANN Controller: ANN Realized Forces

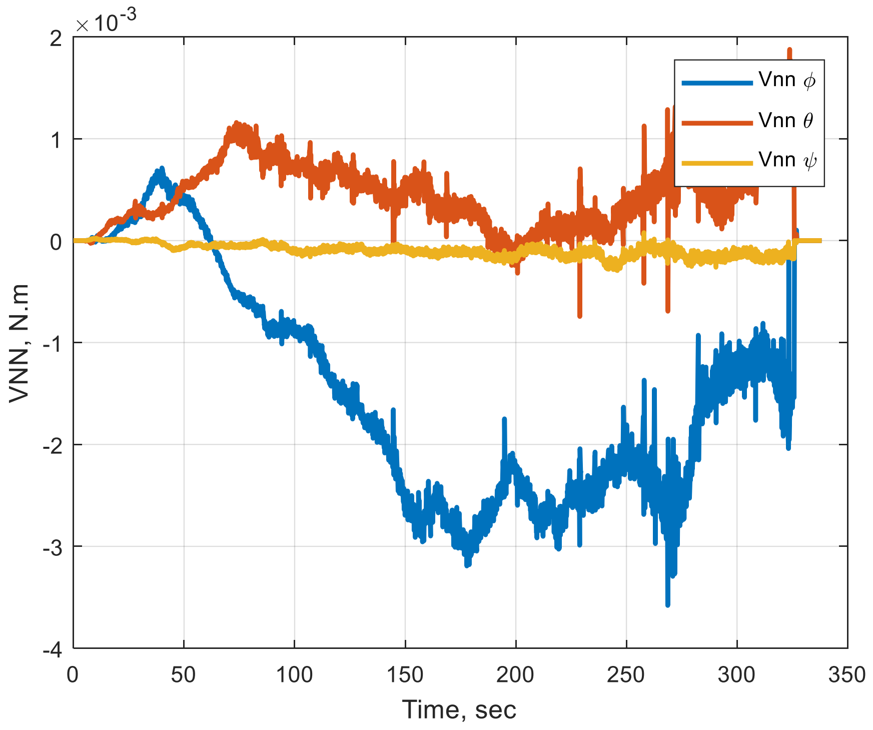


Figure 4.33 (f) Experimental Results for Case 6 Using ANN Controller: ANN Realized Moments

4.5.7 One-Wall, Redone (With Damaged Propeller)

The one wall case presented previously is redone with some adjustments to the setup and the inclusion of actuator dynamics for the ANN controller. A longer wall is made and the Qball is commanded to hover at a low altitude close to the wall, Figure 4.34.

Figures 4.36, 4.37 and 4.38 show the response for this case with the ANN, PID (200 Hz), and PID (80 Hz) respectively.

Notice that for all previous runs the Qball controller is set to 200 Hz sampling rate when the PID controller is used, while it is set to 80 Hz when the ANN controller is used. This case is intended to show the effect of sampling rate on performance, in addition to performance evaluation for the specific set up.

Notice that the X force has more positive magnitude compared with that of case 2, with a mean of $0.867\,N$ compared with $0.5276\,N$ for case 2, this looked skeptical, as both cases, roughly, share the same setup. Upon investigation, the front blade was found slightly damaged, as seen in Figure 4.35. This damage occurred at a crash that happened in the process of adding actuator dynamics to the ANN controller, before running case 6, however, it was not identified in the results of case 6 as the aerodynamic forces associated with that setup (*between the chairs*) dominated the effect of propeller damage. When the front propeller is damaged, it produces less lift and accordingly the quadrotor tends to pitch forward, and as a result it generates a positive X force.

Notice in the PID case that the Qball balances about a positive X value instead of zero. Which validates the conclusion drawn by the ANN. Also notice that the pitch command in this case saturates on the negative side but not on the positive side, which also shows that the Qball tends to pitch forward, as a result of the broken propeller.

The response in the ANN case shows a good performance, which indicates that the controller could successfully reject uncertainties associated with environment setup and propeller damage. However, in both cases a successful run was achieved.

Notice the significant deterioration in PID performance when the sampling rate is reduced to match that of the ANN. The low sampling rate has a great effect on attitude control, as attitude has much faster dynamics compared with trajectory dynamics which updates at 60 Hz at the SLAM dunk sensor.

The bang-bang nature of the roll control is a sign of instability, as the limits for the roll angles are adjusted carefully by Quanser to prevent instability. However, notice that the PID have produced better error RMS than ANN, as shown in Table 4-4. Therefore, the Qball performance in this case is a measure for the ban-bang controller rather than the PID's. And thus, if these attitude hard-limits are removed, the PID would not be able to maintain stability. It has been observed too, that quadrotor's performance with the ANN controller looks visually steadier than the PID.

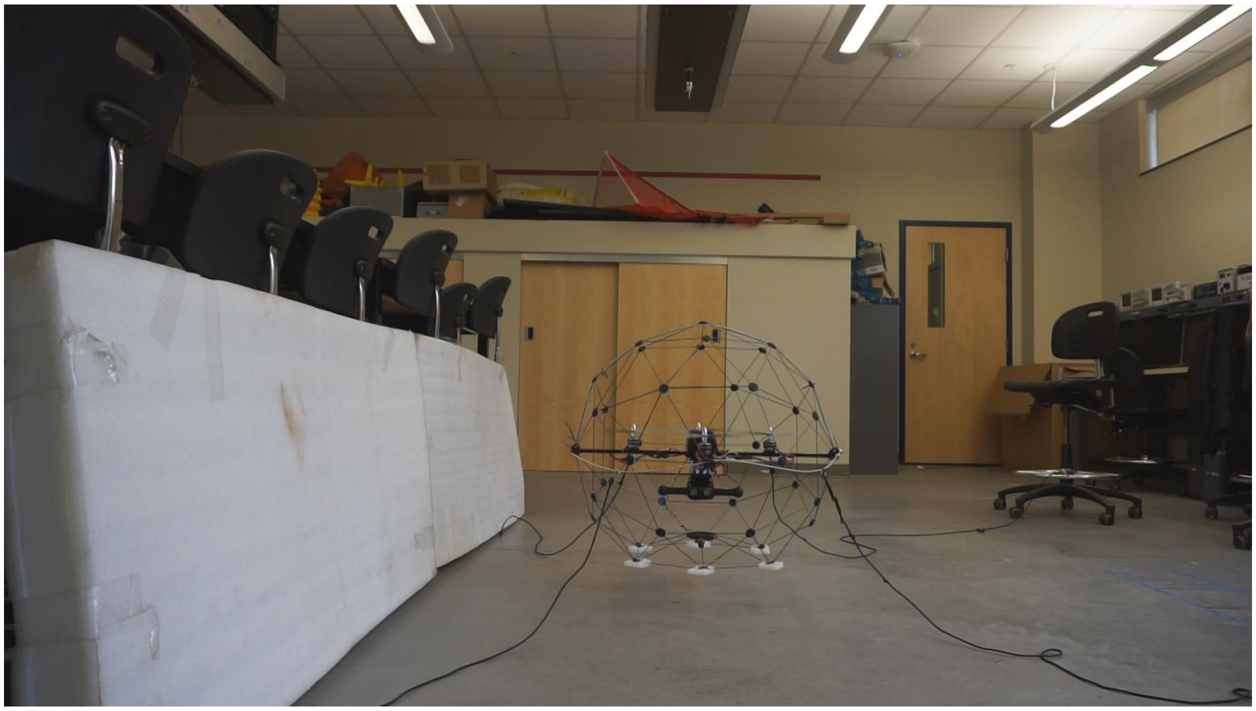


Figure 4.34 Environment Setup for Case 7: Modified One-Wall with Actuator Dynamics

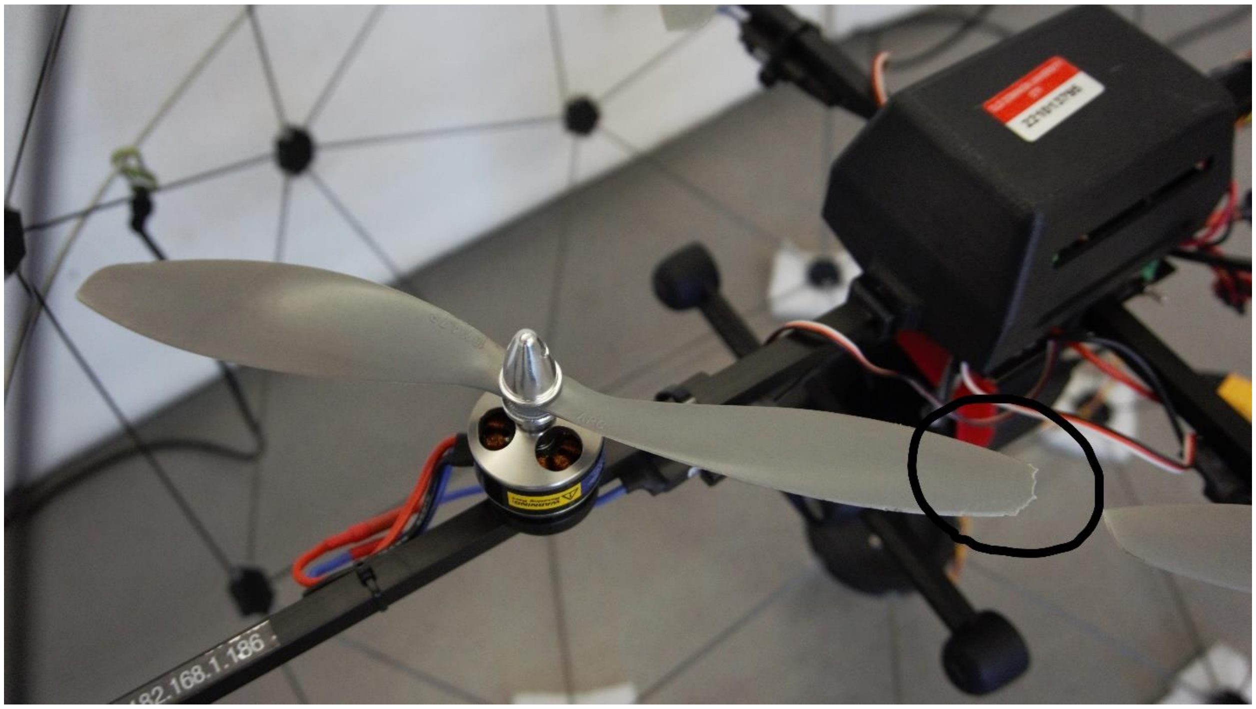


Figure 4.35 Damaged Propeller

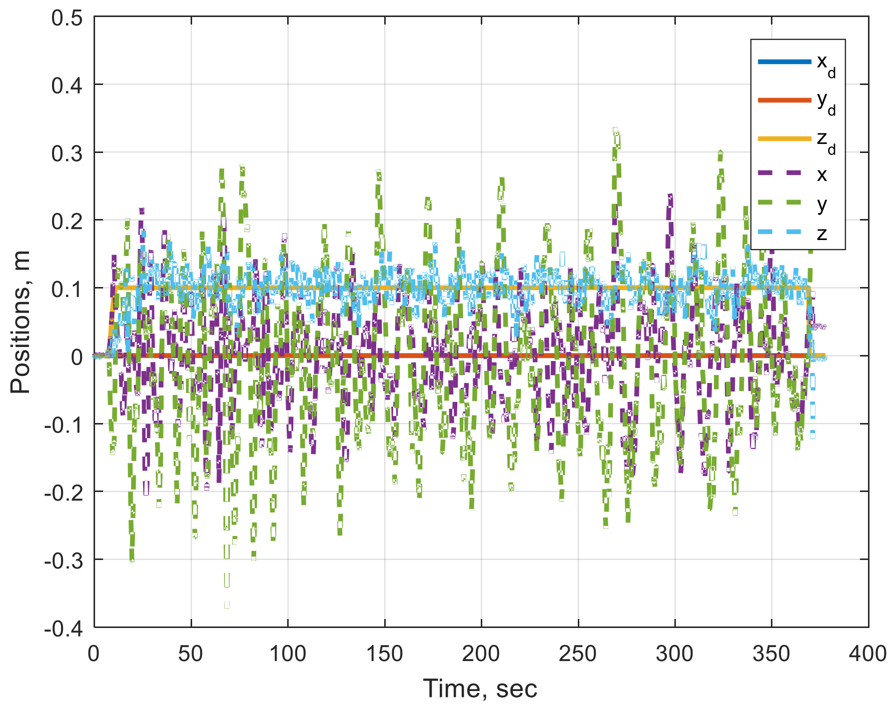


Figure 4.36 (a) Experimental Results for Case 7 Using ANN Controller: Translational Response

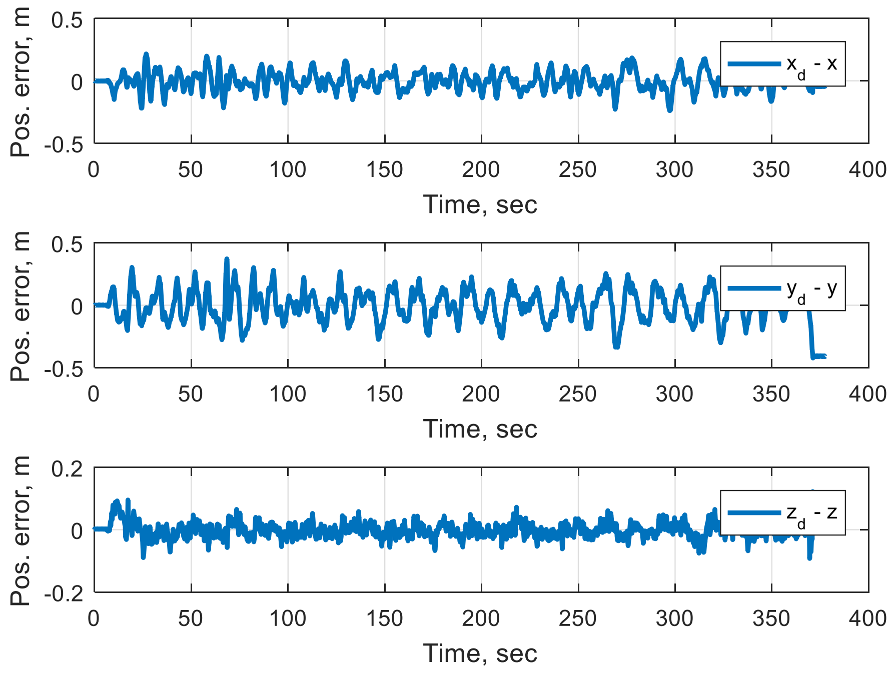


Figure 4.36 (b) Experimental Results for Case 7 Using ANN Controller: Position Errors

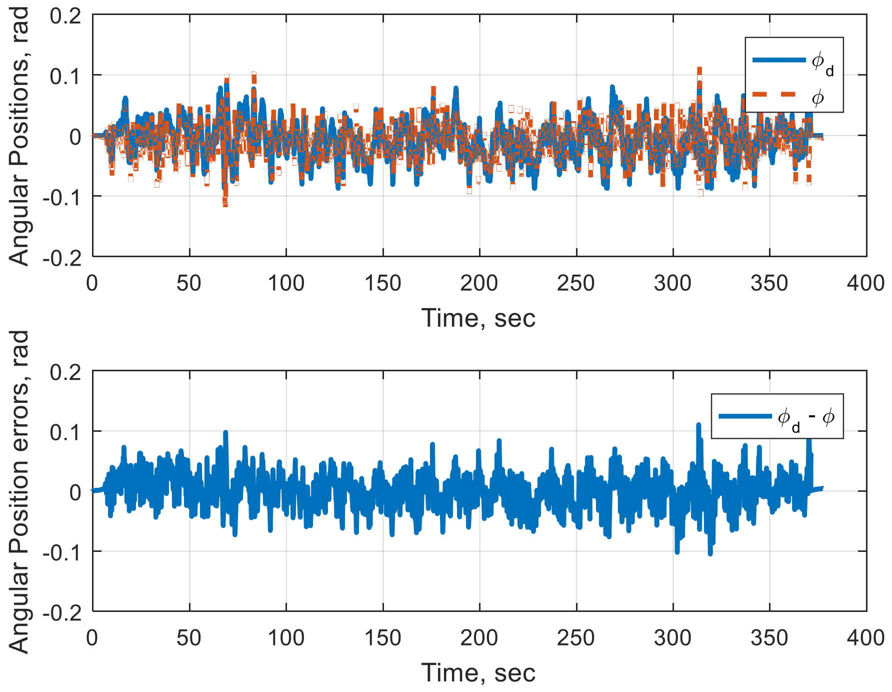


Figure 4.36 (c) Experimental Results for Case 7 Using ANN Controller: Roll Angle Response

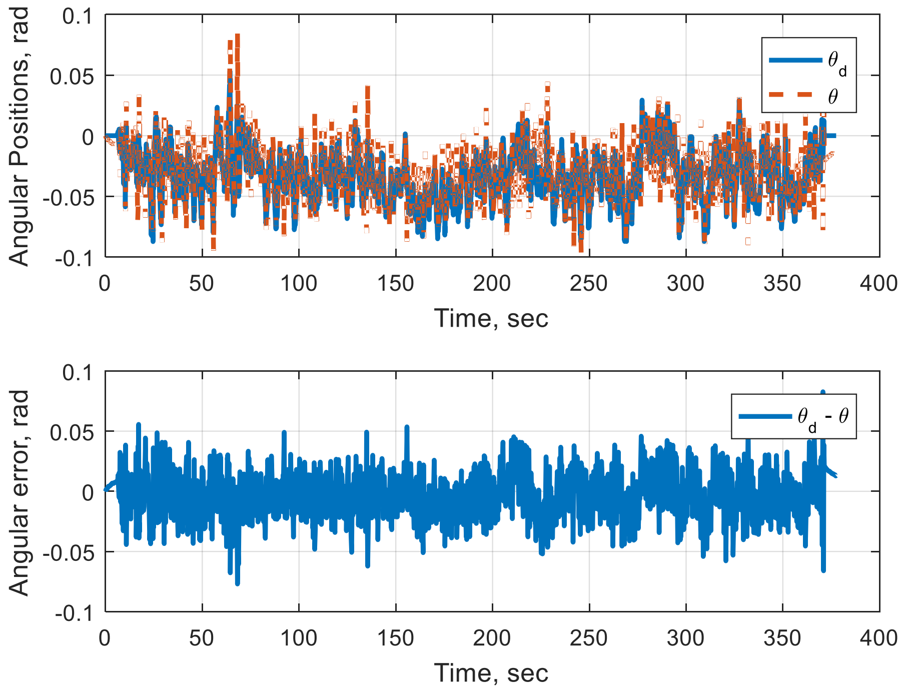


Figure 4.36 (d) Experimental Results for Case 7 Using ANN Controller: Pitch Angle Response

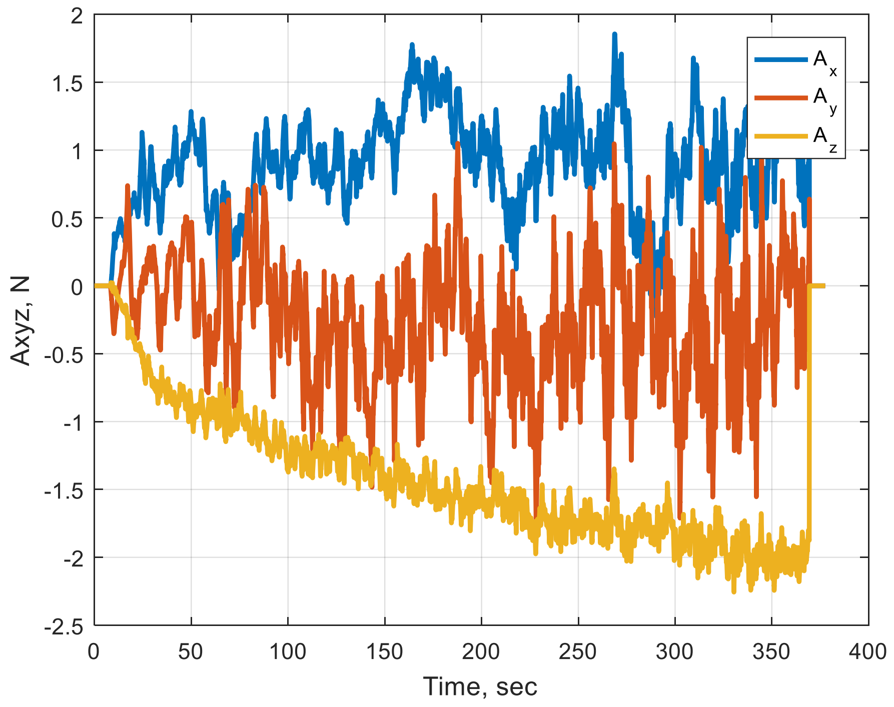


Figure 4.36 (e) Experimental Results for Case 7 Using ANN Controller: ANN Realized Forces

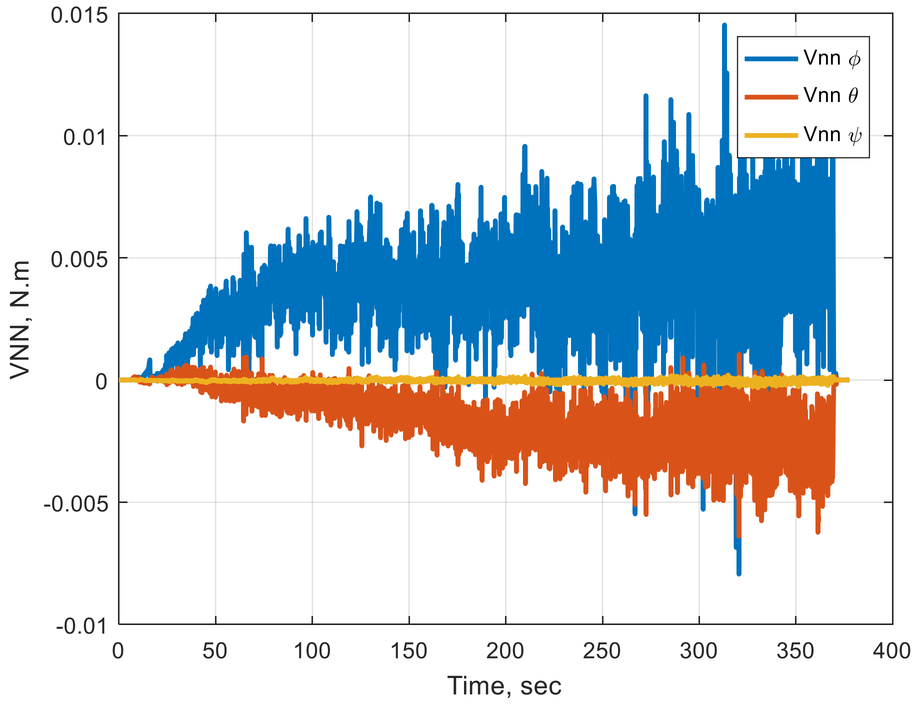


Figure 4.36 (f) Experimental Results for Case 7 Using ANN Controller: ANN Realized Moments

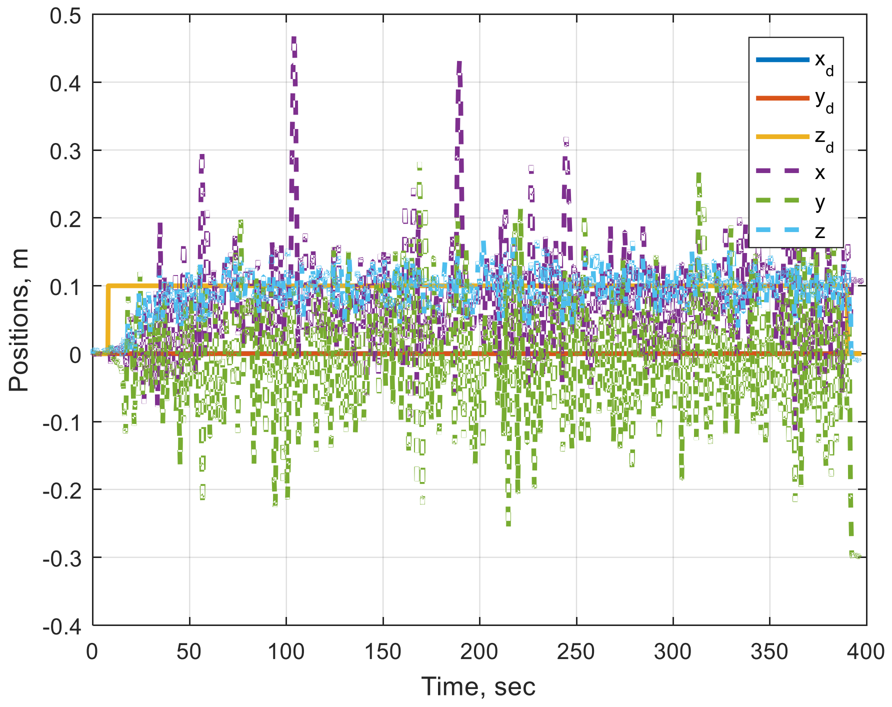


Figure 4.37 (a) Experimental Results for Case 7 Using PID Controller (200 Hz): Translational Response

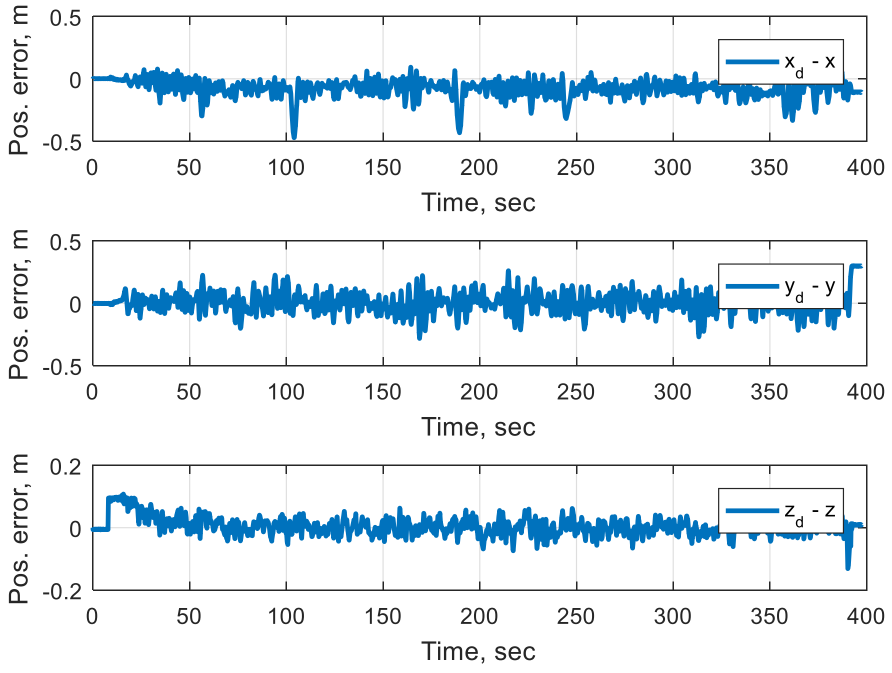


Figure 4.37 (b) Experimental Results for Case 7 Using PID Controller (200 Hz): Position Errors

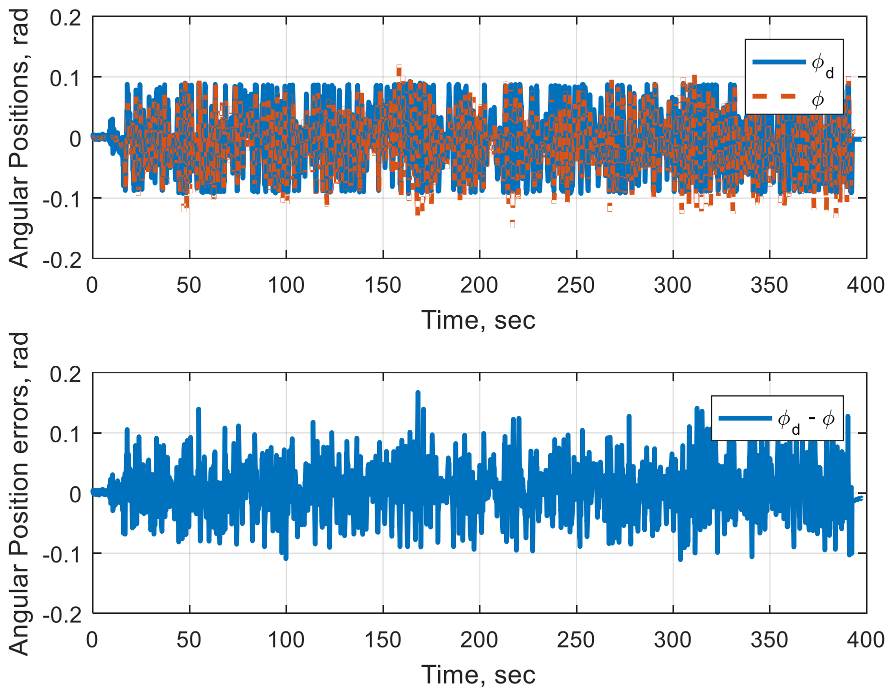


Figure 4.37 (c) Experimental Results for Case 7 Using PID Controller (200 Hz): Roll Angle Response

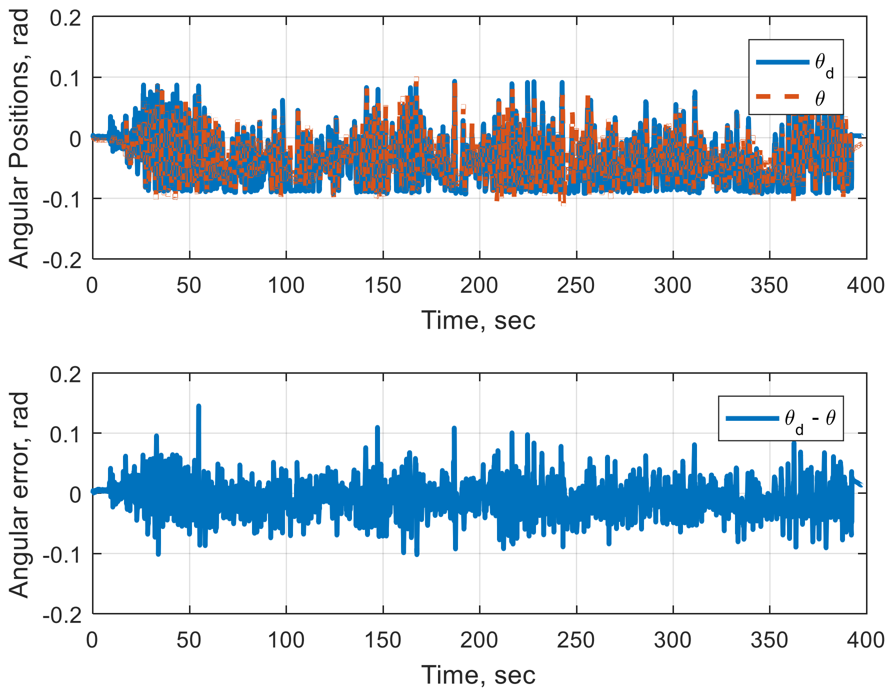


Figure 4.37 (d) Experimental Results for Case 7 Using PID Controller (200 Hz): Pitch Angle Response

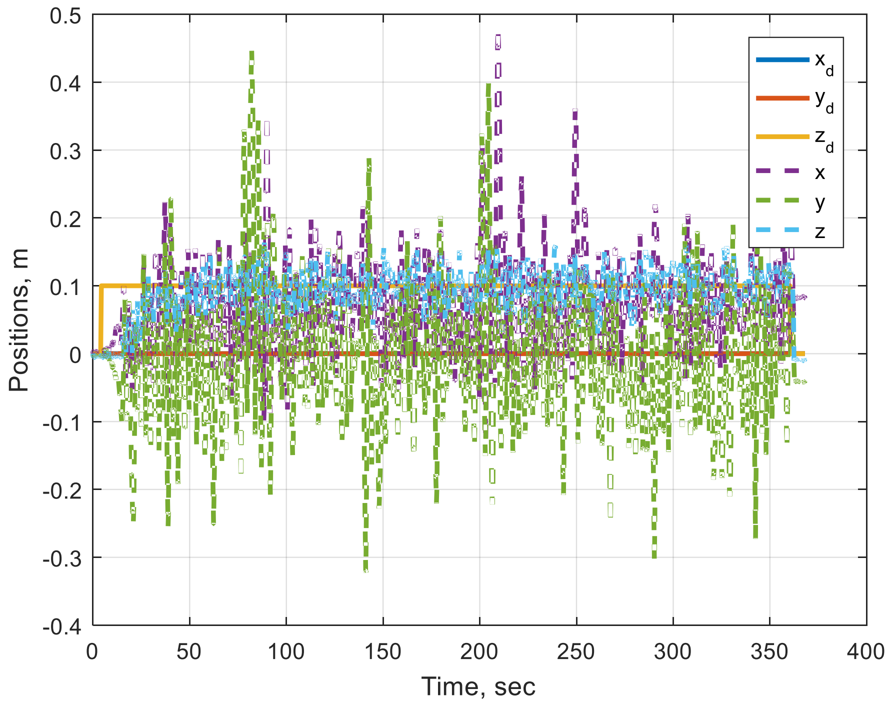


Figure 4.38 (a) Experimental Results for Case 7 Using PID Controller (80 Hz): Translational Response

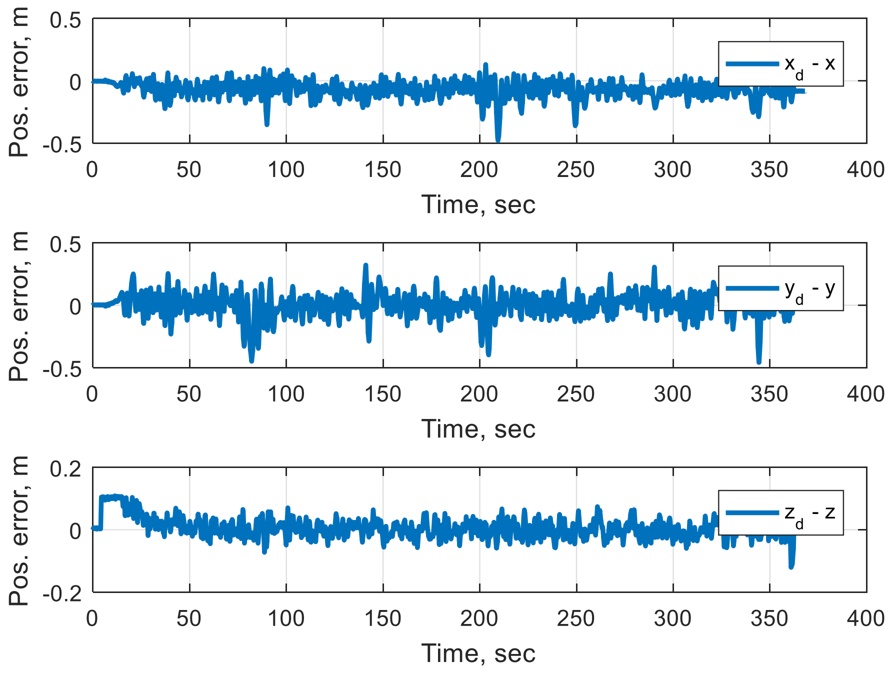


Figure 4.38 (b) Experimental Results for Case 7 Using PID Controller (80 Hz): Position Errors

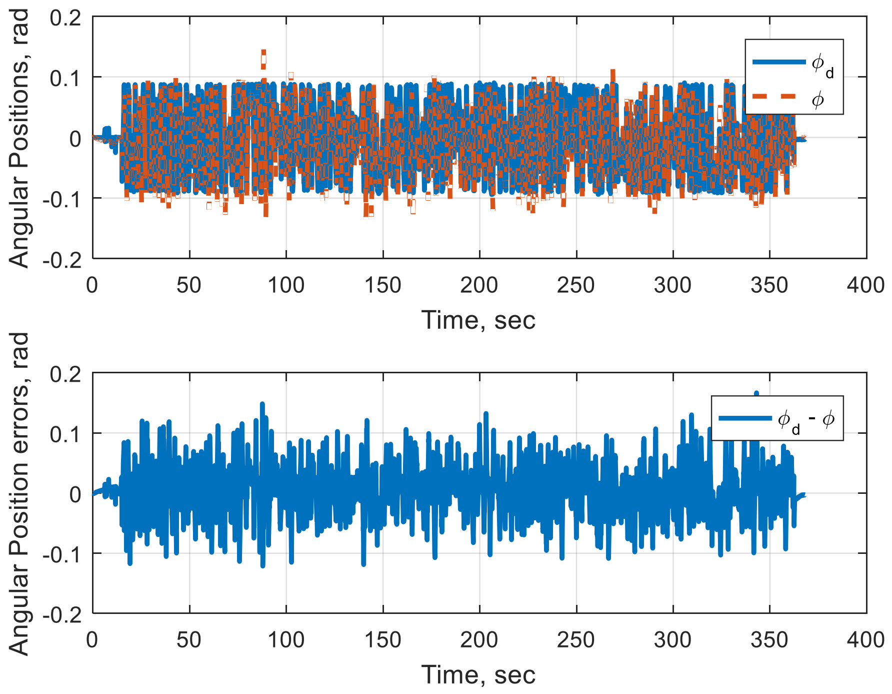


Figure 4.38 (c) Experimental Results for Case 7 Using PID Controller (80 Hz): Roll Angle Response

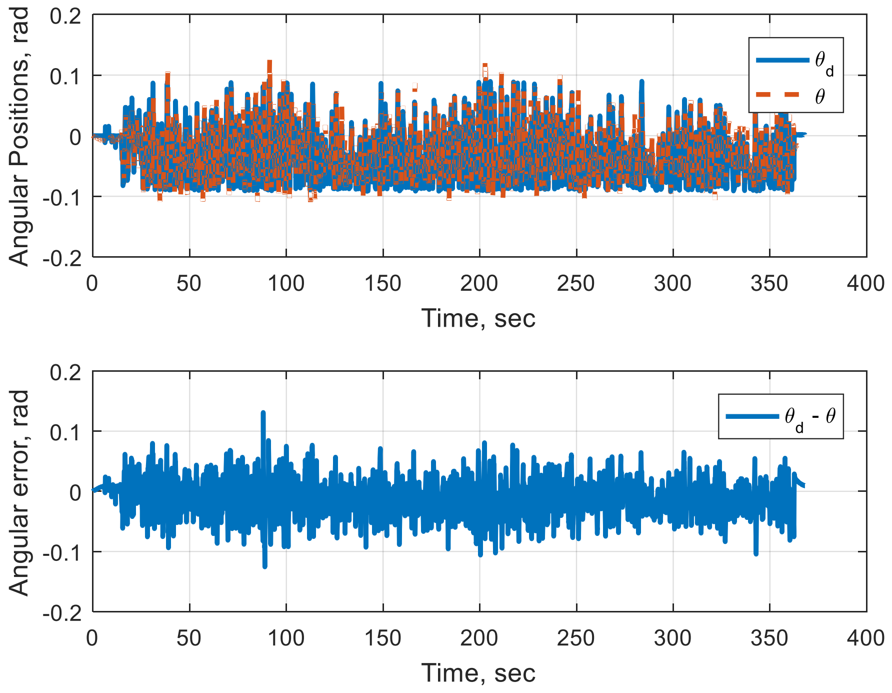


Figure 4.38 (d) Experimental Results for Case 7 Using PID Controller (80 Hz): Pitch Angle Response

4.5.8 Trajectory Following, with Fixed Propellers, and Added Actuator Dynamics

In this case the damaged propeller is fixed, and actuator dynamics are included, and the Qball is operated in a random position in the laboratory. Qball is given commands in X, Y and Z in this case to evaluate the general performance. The response of the Qball with the ANN controller is shown in Figure 4.40. At the maximum X command, the right back tether was pulling the Qball back, and it can be seen, in Figure 4.40 (e), that the ANN was reacting to the force applied by the tether. Comparing this response with PID's, Figure 4.39, it is clear that the ANN performance is satisfactory. A very important benefit of the ANN controller is the consistency in performance and robustness, a property the PID controller lacked.

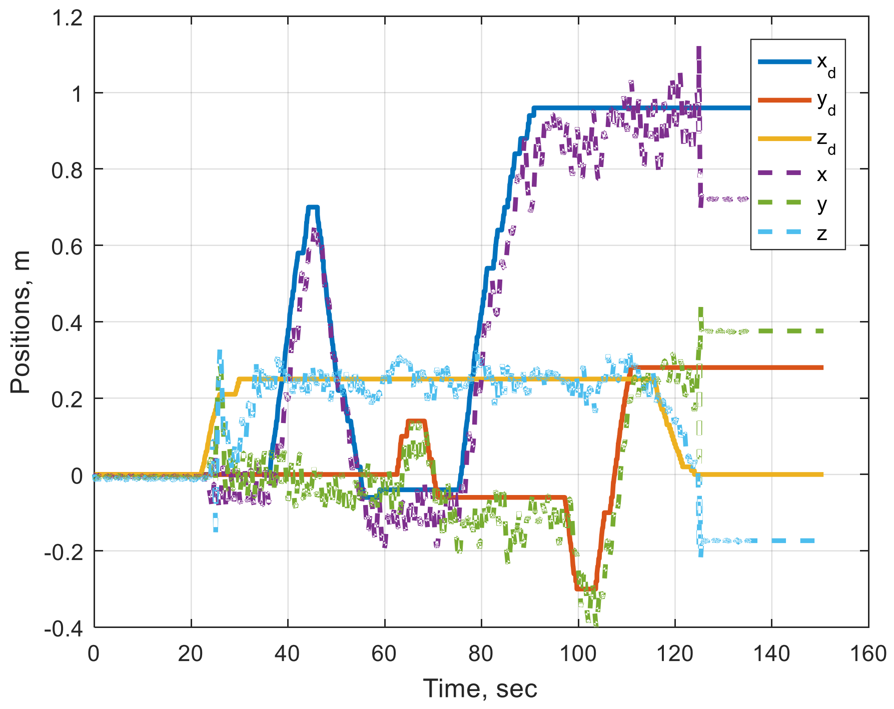


Figure 4.39 (a) Experimental Results for Case 8 Using PID Controller: Translational Response

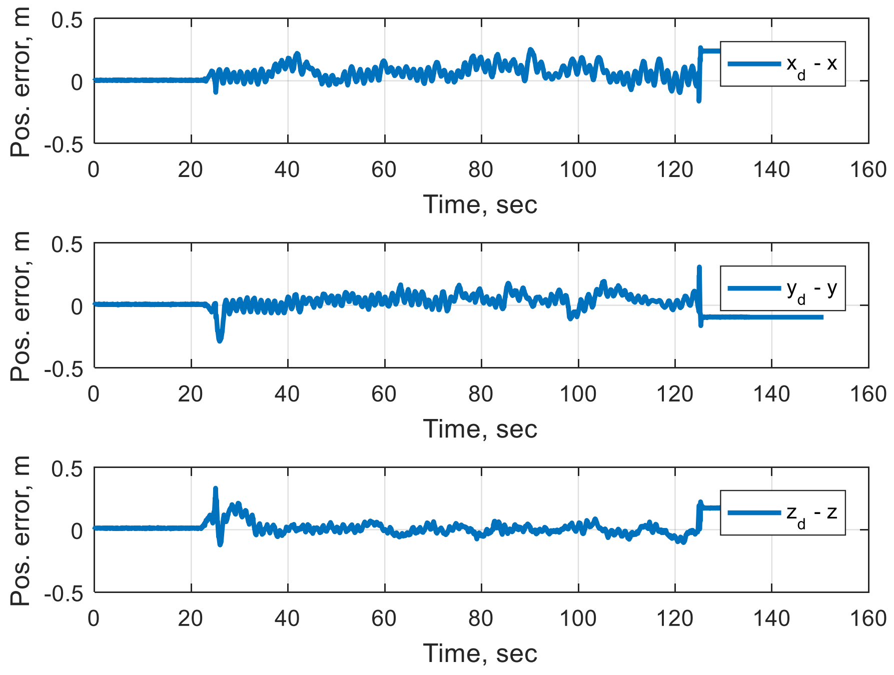


Figure 4.39 (b) Experimental Results for Case 8 Using PID Controller: Position Errors

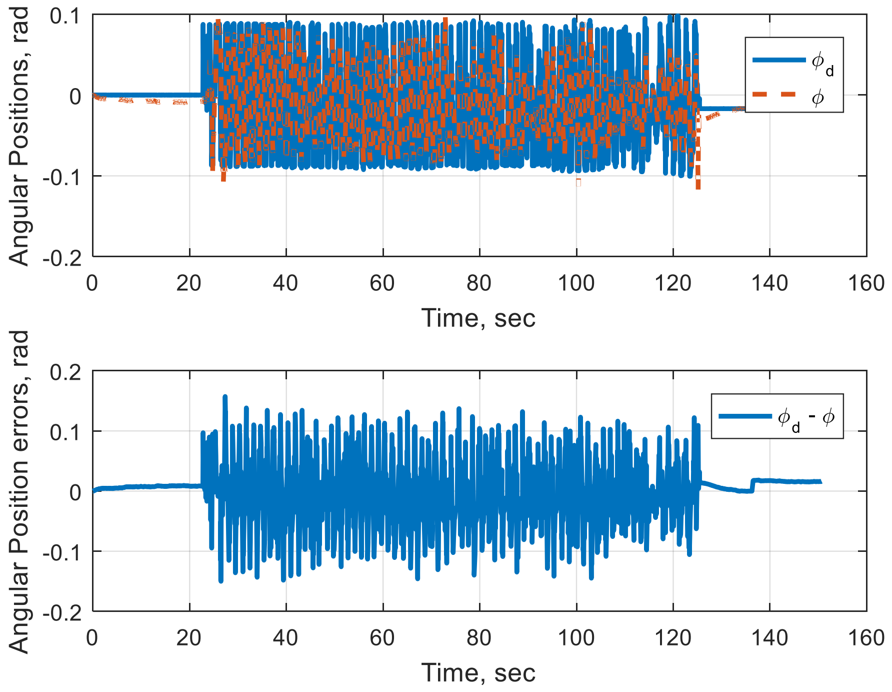


Figure 4.39 (c) Experimental Results for Case 8 Using PID Controller: Roll Angle Response

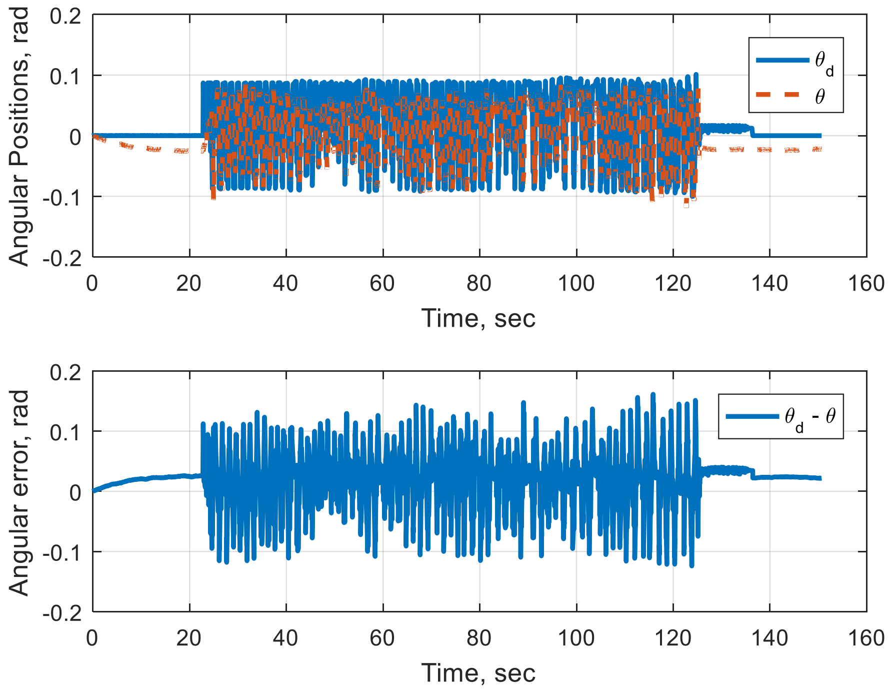


Figure 4.39 (d) Experimental Results for Case 8 Using PID Controller: Pitch Angle Response

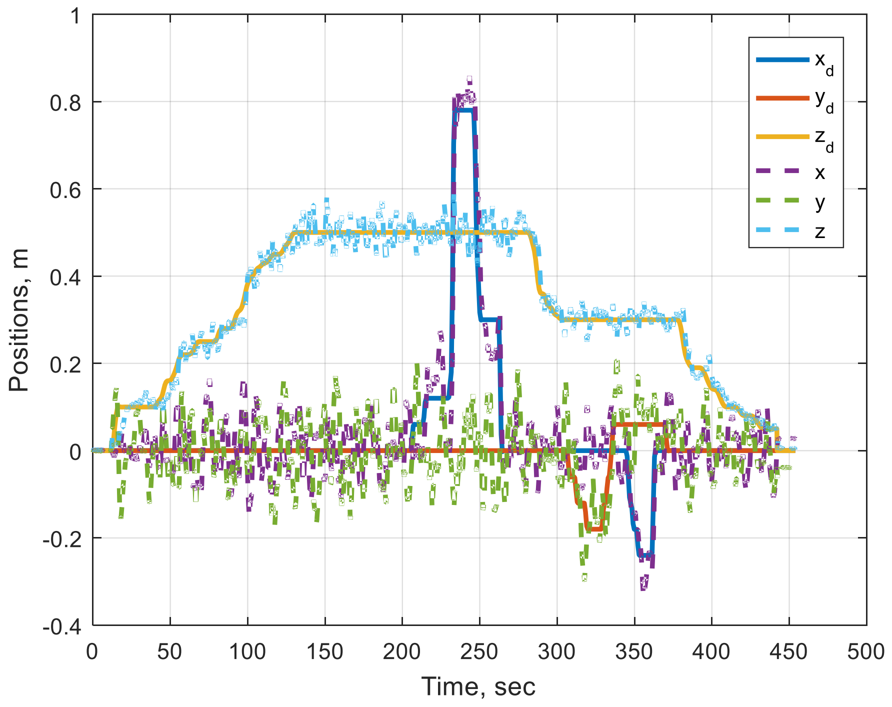


Figure 4.40 (a) Experimental Results for Case 8 Using ANN Controller: Translational Response

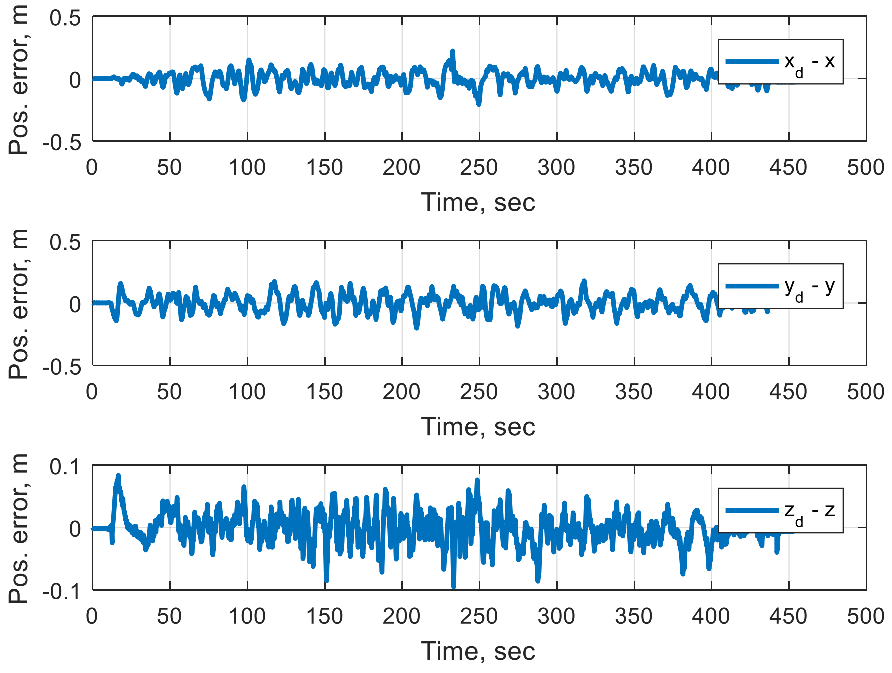


Figure 4.40 (b) Experimental Results for Case 8 Using ANN Controller: Position Errors

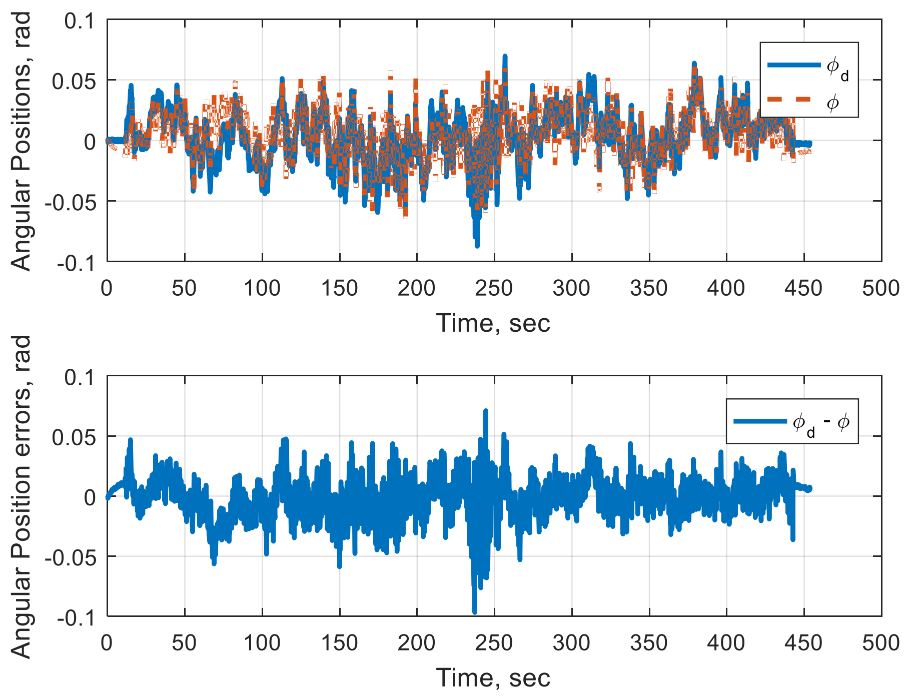


Figure 4.40 (c) Experimental Results for Case 8 Using ANN Controller: Roll Angle Response

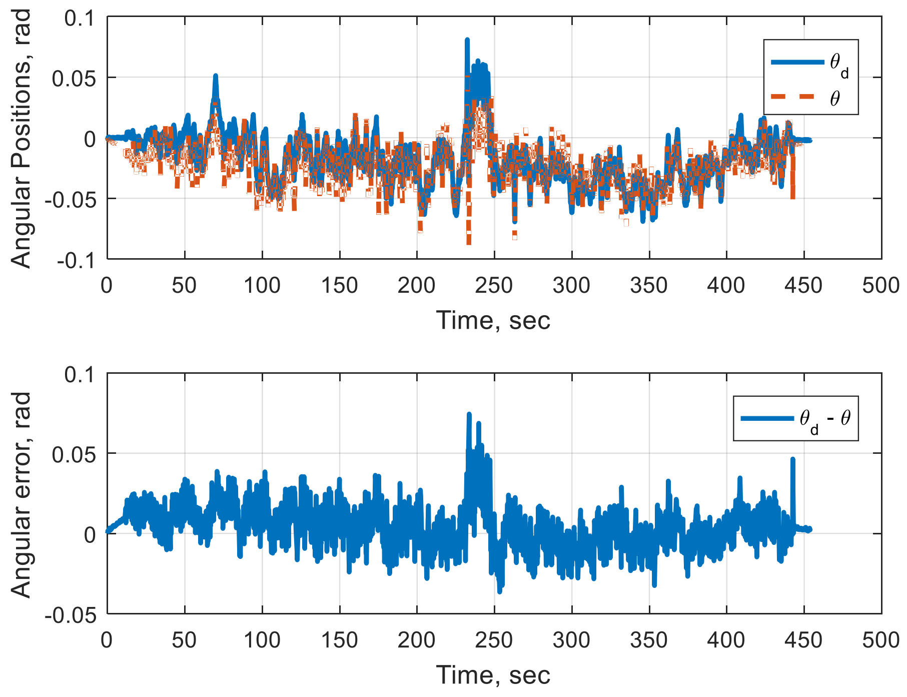


Figure 4.40 (d) Experimental Results for Case 8 Using ANN Controller: Pitch Angle Response

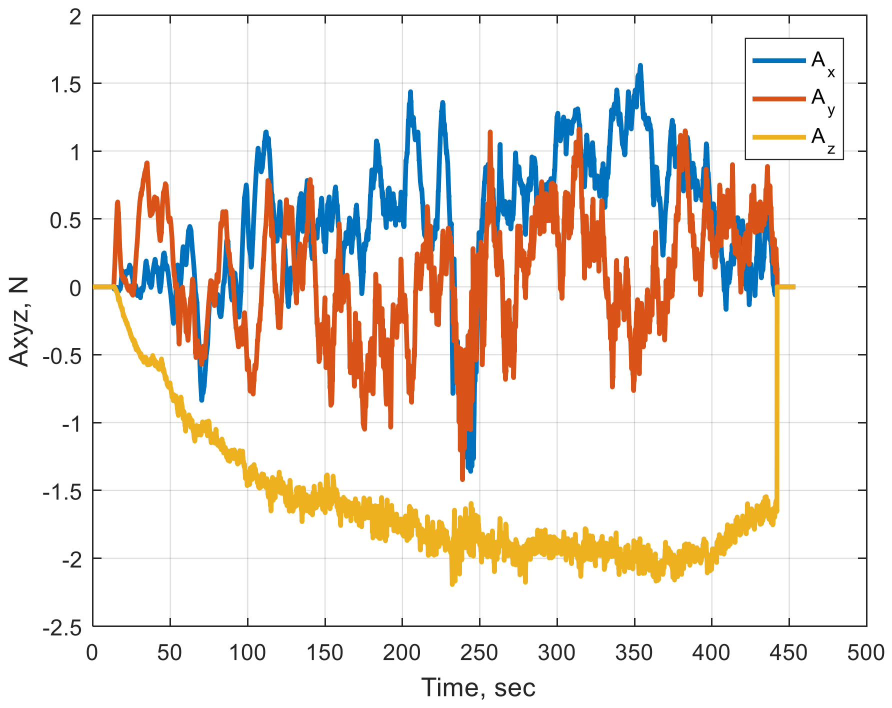


Figure 4.40 (e) Experimental Results for Case 8 Using ANN Controller: ANN Realized Forces

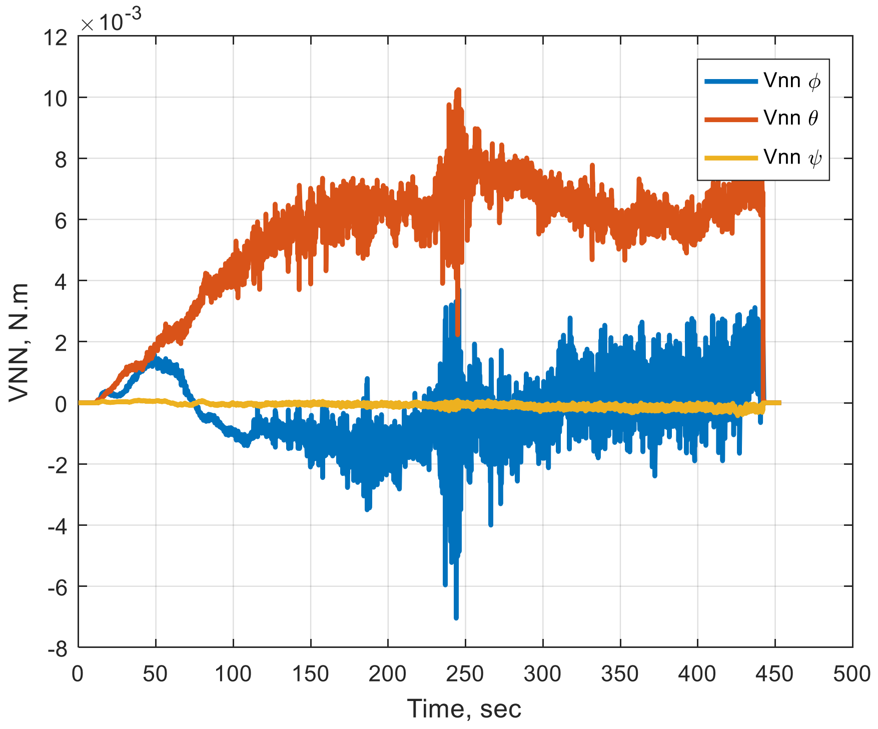


Figure 4.40 (f) Experimental Results for Case 8 Using ANN Controller: ANN Realized Moments

Table 4-4 Summery of Experimental Error RMS

Case	ANN Controller					PID Controller				
	$X\ (m)$	$Y(m)$	$Z(m)$	$\phi(rad)$	$\theta(rad)$	$X\ (m)$	$Y(m)$	$Z(m)$	$\phi(rad)$	$\theta(rad)$
No Wall	0.058	0.070	0.022	0.013	0.013	0.075	0.082	0.029	0.033	0.039
One Wall	0.054	0.129	0.019	0.023	0.013	0.085	0.101	0.027	0.034	0.028
Corner of Two Walls	0.068	0.078	0.017	0.013	0.013	0.065	0.059	0.02	0.027	0.035
Between Chairs	0.065	0.063	0.024	0.013	0.013	0.085	0.083	0.031	0.034	0.032
Tunnel-Like	0.061	0.122	0.036	0.025	0.023	0.251	0.169	0.073	0.040	0.058
Added Actuator Dynamics	0.055	0.060	0.019	0.012	0.008	NA	NA	NA	NA	NA
One Wall (redone)	0.076	0.133	0.023	0.025	0.017	0.106 0.101	0.087 0.102	0.029 0.032	0.035 0.036	0.026 0.028
Trajectory Tracking	0.057	0.068	0.023	0.015	0.013	0.124	0.070	0.083	0.038	0.0425

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

In this chapter, conclusions from this dissertation are discussed, and some recommendations are presented in terms of what research work could be done as a continuation of this research.

5.1 CONCLUSIONS

Quadrotors are proven to provide practical solutions for many applications, as a result, a significant amount of research is drawn towards them in both industrial and academic establishments. One important application is to make quadrotors execute indoor missions in hazardous conditions. While quadrotors could operate in unreachable places, they are difficult to manage in confined environments, because of the associated complex aerodynamic forces and moments. Researches approached the issue of quadrotor maneuverability in confined environment following three schools of thought; 1) Development of a high-fidelity model, 2) Sensory integration, and 3) Development of a robust and adaptive controller.

In this research, an ANN controller is designed to improve quadrotors flight control in confined environment. The proposed controller makes use of the modeling capability of ANNs to compensate for the aerodynamic uncertainties associated with confined environments. In addition, two nonlinear control methodologies are followed to design the controller, and to allow for shaping desired closed loop performance based on control theories.

In an effort to make this dissertation inclusive, required background in the fields of AI and control systems is provided. Hence, an introduction about the use of AI in the field of control systems is provided, with a focus on ANNs. Advantages and challenges of using such techniques in control systems are discussed. Furthermore, a brief theoretical background for the utilized control methodologies, sliding mode and backstepping, is presented. In addition, the Lyapunov stability theorem, used for stability analysis, and relevant stability definitions are provided.

A systematic design procedure for the developed controller when applied to quadrotors is presented, with the aim of improving stability, robustness, and performance when flying in confined environments. The step-by-step procedure could be followed to design controllers for nonlinear systems with similar form.

A novel ANN structure is proposed that improves learning speed and accuracy of the Linear-In-the-Parameters ANNs. This structure takes into account the linear relationship between system's inputs and outputs, in addition, to the nonlinear relationship. A novel ANN learning algorithm is developed to insure boundedness of the weights and accuracy of the ANNs output. The learning problem is tackled as an optimization problem with an objective function chosen based on sliding mode methodology. Lyapunov stability theorem is used to define the limits of the design parameters for the guaranteed stability of the closed loop system. The effectiveness of the proposed ANN structure when combined with the developed learning algorithm is presented using numerical simulation when applied to quadrotors. It is shown that the developed learning mechanism works efficiently in estimating different types of model and aerodynamic uncertainties. Finally, experimental results of the proposed controller when applied to quadrotor flight control in confined environment are presented. Challenges that limit the practical application of the developed ANN controller for quadrotors are discussed. In the process of controller implementation on the Qball quadrotor, a SLAM sensor is integrated with the Qball to allow for trajectory measurement, and necessary software modifications are made for the Qball modules to accommodate for the additional hardware. This modification increases the level of autonomy of the Qball, expands its range of operation, and replaces the expensive OptiTrack system. Experimental results showed the robustness, and effectiveness of the designed controller. The novelty of this research is in the proposed ANN structure, the developed learning algorithm, and the successful real-time implementation of the online ANN controller in the application of quadrotor trajectory control. Upon finishing this research, the following conclusions are drawn:

The developed ANN controller required minimal knowledge of the system's dynamics. However, because of its intuitive structure, the desired closed loop performance of the system could be naturally shaped by the choice of controller gains. The closed loop system, herein, consists of three first-order subsystems; the ANN learning dynamics, sliding mode hyperplane switching dynamics, and the sliding mode dynamics. Each one of these subsystems has its own gains that determine how fast it reaches steady state, while enforcing the relationships defined in theorem 3.1 for stability. In general, the learning dynamics should be faster than the hyperplane switching dynamics, which in turn, should be faster than the sliding mode dynamics.

It is important to note that the transient response of the closed loop system is influenced by the three dynamics at the time, until the learning and hyperplane switching dynamics settle, then the dynamical behavior of the system is solely defined by the sliding mode.

The proposed ANN structure combines the merits of one-layer-ANNs and multilayer-ANNs of learning speed and learning performance, with the choice of the general stochastic basis function. This structure is found to be very effective when combined with the developed learning algorithm. One important feature of this combination is that it only requires knowledge of explanatory variables of the sought unknown functions to be implemented. These explanatory variables are application dependent, and they are well documented for each application. In the case of unknown explanatory variable, all available measurements could be fed to the ANNs as inputs, and accordingly less influential variables would automatically achieve near zero weights when approximation is complete.

The designed ANN controller was successfully tested, in real-time experiment, on the flight control of quadrotors in confined environment. Experimental results showed that the benefits of the ANN controller go beyond closed loop performance, robustness, and stability in the presence of unknown uncertainties, to providing a model, in real-time, for the uncertainties experienced during flight, in the specific setup. Realized models could be studied later to better understand the influence of unknown aerodynamics associated with the flight environment. And furthermore, they could be implemented with numerical simulations to perform realistic tests for developed controllers when applied to quadrotors. This benefit provides an important solution to the first research school of thought in the field of quadrotor control, which focuses on advancing quadrotors control by seeking better models for them.

Limitations, such as, onboard computational power, and sensor measurement noise are found to be very influential when applying ANN controllers to real-time systems. Computational power, for example, provides a restriction to the choice of sampling rate at which the ANN learns, which in turn leads to slow learning. This limitation is found especially important in the application of quadrotor because of its fast attitude dynamics, which requires fast learning. Measurement noise on the other hand, affects learning in the sense that it appears to the ANN as real disturbances that should be accounted for. In addition, noise affects the performance of the nonlinear controller, especially when the time derivative of these measurements is required. These two factors are believed to be the reason for the lack of experimental results in the literature for the application of ANN controllers in quadrotors trajectory control, as they are very challenging to be successfully implemented in real time experimentation.

5.2 RECOMMENDATIONS

The developed controller is found to provide a very promising solution for the field of control of nonlinear systems, as it could achieve high levels of robustness and performance, however, more research is needed to analyze its performance and test its limits, especially in practical applications. Therefore, it is recommended that this controller be applied to different nonlinear systems, and to develop a solid closed form procedure for the use of such controller with nonlinear systems in different forms.

It is recommended to test the designed controller on different quadrotors to confirm its globality. As it provides the important advantage of requiring minimal adjustments when applied for different quadrotors. Upon confirmation, this controller could be installed in a black-box with few parameters to adjust and it would work with any quadrotor.

In addition to robustness, the ANN controller produced mathematical models for the aerodynamic forces and moments associated with confined environment. Accordingly, it is recommended to operate the Qball with the ANN controller and collect the produced ANN models for a wide spectrum of environment setups and publish these functions to be used for quadrotor controller testing. With the aid of these models, quadrotors could be tested using numerical simulations with the presence of realistic disturbances; which is lacking in the literature.

More research is still needed in the field of ANNs in control systems, especially in the aspect of practical application. ANNs are shown to be very effective in compensating for a wide spectrum of unknown uncertainties, as they represent global estimators. However, the practical application of ANNs in control systems is proven to be very challenging. The research in this field is expected to produce rapid and innovative results because of the availability of newly developed powerful, and yet small, onboard PCs, and the current advancement in sensory technologies.

The SLAM algorithm, used in SLAM dunk, needs some improvement. As some performance issues were observed during experimental implementation that need to be addressed, such as, localization deficiency due to low lighting condition, moving objects, repetition of features in the field of view, and the limited sampling rate. The reliability of SLAM dunk algorithm is crucial, as the performance of the Qball depends heavily on Parrot's SLAM dunk performance in providing consistent trajectory measurement.

The Parrot SLAM dunk kit comes with a powerful onboard PC, which is integrated with an IMU and additional sensors. Accordingly, it is recommended to code Qball's modules on SLAM dunk's PC to make use of its computational power, on one hand, and to make use of its IMU sensors, on the other hand. This would allow for the implementation of very complex controller algorithms with very high sampling rates, in contrast to Qball's onboard PC which has a limited processing power.

The Qball with the Parrot SLAM dunk sensor represents a potential apparatus for research in the field of autonomy and robotics. Accordingly, it is recommended to conduct extra research towards developing a fully autonomous UAV robot using the Qball, to deliver indoors missions. There are very interesting research subjects that could be conducted using this device, such as, the development and implementation of obstacle avoidance algorithms, navigation and control algorithms, and acrobatic maneuvers ...etc.

REFERENCES

[1] D. Floreano and R. J. Wood, "Science, technology and the future of small autonomous drones," *Nature,* vol. 521, p. 460–466, 2015.

[2] W. Guo and J. F. Horn, "Modeling and Simulation for the Development of a Quad-Rotor UAV Capable of Indoor Flight," in *AIAA Modeling and Simulation Technologies Conference and Exhibit*, Keystone, 2006.

[3] Y. Chen, Y. He and M. Zhou, "Modeling and Control of a Quadrotor Helicopter System Under the Impact of Wind Field," *Research Journal of Applied Sciences: Engineering and Technology,* vol. 6, no. 17, pp. 3214-3221, 2013.

[4] C. Powers, D. Mellinger, A. Kushleyev, B. Kothmann and V. Kumar, "Influence of Aerodynamics and Proximity Effects in Quadrotor Flight," in *Experimental Robotics: The 13th International Symposium on Experimental Robotics*, Springer, 2013, pp. 289-302.

[5] S. Bansal, A. K. Akametalu, F. J. Jiang, F. Laine and C. J. Tomlin, "Learning Quadrotor Dynamics Using Neural Network for Flight Control," in *IEEE 55th Conference on Decision and Control* , Las Vegas, 2016.

[6] G. M. Hoffmann, H. Huang, S. L. Waslander and C. J. Tomlin, "Quadrotor Helicopter Flight Dynamics and Control: Theory and Experiment," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, Hilton Head, 2007.

[7] S. Bouabdallah and R. Siegwart, "Full Control of a Quadrotor," in *IEEE International Conference on Intelligent Robots and Systems*, San Diego, 2007.

[8] K. Alexis, G. Nikolakopoulos and A. Tzes, "Model Predictive Quadrotor Control: Attitude, Altitude and Position Experimental Studies," *IET Control Theory and Applications,* vol. 6, no. 12, pp. 1812-1827, 2012.

[9] S.-E. Tsai and S.-H. Zhuang, "Optical Flow Sensor Integrated Navigation System for Quadrotor in GPS-Denied Environment," in *International Conference on Robotics and Automation Engineering*, Jeju, 2016.

[10] V. Grabe, H. H. Bulthoff and P. R. Giordano, "Robust Optical-Flow Based Self-Motion Estimation for a Quadrotor UAV," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vilamoura, 2012.

[11] M. W. Mueller, M. Hamer and R. D'Andrea, "Fusing Ultra-Wideband Range Measurements with Accelerometers and Rate Gyroscopes for Quadrocopter

State Estimation," in *IEEE International Conference on Robotics and Automation*, Seattle, 2015.

[12] B. Demir, R. Bayir and F. Duran, "Real-Time Trajectory Tracking of Unmanned Arial Vehicle Using a Selt-Tuning Fuzzy Proportional Integral Derivative Controller," *International Journal of Micro Air Vehicles,* vol. 8, no. 4, pp. 252-268, 2016.

[13] R. Lopez-Gutierrez, A. E. Rodriguez-Mata, S. Salazar, I. Gonzalez-Hernandez and R. Lozano, "Robust Quadrotor Control: Attitude and Altitude Real-Time Results," *Journal of Intelligent Robot Systems,* vol. 88, pp. 299-312, 2017.

[14] C. Nicol, C. J. B. Macnab and A. Ramirez-Serrano, "Robust Neural Network Control of a Quadrotor Helicopter," in *Canadian Conference on Electrical and Computer Engineering*, Niagara Falls, 2008.

[15] T. Dierks and S. Jagannathan, "Output Feedback Control of a Quadrotor UAV Using Neural Networks," *IEEE Transactions on Neural Networks,* vol. 21, no. 1, pp. 50-66, 2010.

[16] H. Boudjedir, F. Yacef, O. Bouhali and N. Rizoug, "Dual Neural Network for Adaptive Sliding Mode Control of Quadrotor Helicopter Stabilization," *International Journal of Information Sciences and Techniques,* vol. 2, no. 4, pp. 1-14, 2012.

[17] P. J. Antsaklis, "Intelligent Control," in *Wiley Encyclopedia of Electrical and Electronics Engineering*, Wiley, pp. 493-503.

[18] D. Xun and W. Chang-shan, "An Efficient Sequential Learning Algorithm for Growing and Pruning Direct-Link RBF (DRBF) Networks," in *International Conference on Neural Network and Brain*, Beijing, 2005.

[19] C. Wen, J. Zhou, Z. Lui and H. Su, "Robust Adaptive Control of Uncertain Nonlinear Systems in the presence of Input Saturation and External Disturbance," *IEEE TRANSACTIONS ON AUTOMATIC CONTROL,* pp. 1672-1678, 2011.

[20] C.-F. Hsu, C.-M. Lin and T.-T. Lee, "Wavelet Adaptive Backstepping Control for a Class of Nonlinear Systems," *IEEE TRANSACTIONS ON NEURAL NETWORKS,* vol. 17, no. 5, pp. 1175-1183, 2006.

[21] F. L. Lewis and S. S. Ge, "Neural Networks in Feedback Control Systems," in *Mechanical Engineer's Handbook*, New York, Wiley, 2005.

[22] L. R. Medsker and L. C. Jain, Recurrent Neural Networks: Design and Applications, CRC Press, 2000.

[23] K. J. Hunt, G. R. Irwin and K. Warwick, Neural Network Engineering in Dynamic Control Systems, Springer, 1995.

[24] C.-F. Hsu and C.-M. Lin, "Design of Self-Constructing Recurrent-Neural-Network-Based Adaptive Control," in *Recurrent Neural Networks*, Vienna, Austria, INTECH, 2008.

[25] A. M. Meystel and J. S. Albus, Intelligent Systems: Architecture, Design, and Control, John Wiley and Sons, Inc., 2002.

[26] P. J. Antsaklis, "Intelligent Learning Control," *IEEE Control Systems Magazine,* pp. 5-80, June 1995.

[27] P. J. Antsaklis and K. M. Passino, An Introduction to Intelligent and Autonomous Control, KLUWER ACADEMIC PUBLISHERS, 1993.

[28] M. M. Gupta, L. Jin and N. Homma, Static and Dynamic Neural Networks: From Fundamentals to Advanced Theory, Wiley-Interscience, 2003.

[29] R. E. King, Computational Intelligence in Control Engineering, Marcel Dekker, Inc, 1999.

[30] B. S. Kim and A. J. Calise, "Nonlinear Flight Control Using Neural Networks," *AIAA Journal of Guidance, Control, and Dynamics,* vol. 20, no. 1, pp. 26-33, 1997.

[31] C. Kwan and F. Lewis, "Robust Backstepping Control of Nonlinear Systems Using Neural Networks," *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS,* vol. 30, no. 6, pp. 753-766, 2000.

[32] D. H. Nguyen and B. Widrow, "Neural Networks for Self-Learning Control Syatems," *IEEE Control Systems Magazine,* pp. 18-23, April 1990.

[33] D. Schroder, Intelligent Observer and Control Design for Nonlinear Systems, Springer, 2000.

[34] A. Yesildirek and F. Lewis, "Feedback Linearization Using Neural Networks," *Automatica,* vol. 31, no. 11, pp. 1659-1664, 1995.

[35] W. T. Miller, R. S. Sutton and P. J. Werbos, Neural Networks for Control, Cambridge: The MIT Press, 1990.

[36] K. Warwick, G. Irwin and K. Hunt, Neural Networks for Control and Systems, London: Peter Peregrinus Ltd., 1992.

[37] M. Norgraad, O. Ravn, N. K. Poulsen and L. K. Hansen, Neural Networks for Modelling and Control of Dynamic Systems, London: Springer, 2000.

[38] K. Hornik, M. Stinchcombe and H. White, "Multilayer Feedforward Networks are Universal Approximators," *Neural Networks,* vol. 2, pp. 359-366, 1989.

[39] B. Igelnik and . Y.-H. Pao, "Stochastic Choice of Basis Functions in Adaptive Function Approximation and the Functional-Link Net," *Transactions on Neural Networks,* vol. 6, no. 6, pp. 1320-1329, 1995.

[40] M. Mohammadian, R. Amin and X. Yao, Computational Intelligence in Control.

[41] A. U. Levin and K. S. Narenda, "Control of Nonlinear Dynamical Systems Using Neural Networks: Controllability and Stabilization," *IEEE TRANSACTIONS ON NEURAL NETWORKS,* pp. 192-206, 1993.

[42] T. Kohonen and T. Honkela, "Kohonen Network," *Scholarpedia,* vol. 2, no. 1, p. 1568, 2007.

[43] A. I. Galushkin, Neural Networks Theory, Springer, 2007.

[44] P. J. Werbos, "Backpropagation Through Time: What It Does and How to Do It," in *IEEE*, 1990.

[45] H. Zargarzadeh, T. Dierks and S. Jagannathan, "Optimal Control of Nonlinear Continuos-Time Systems in Strict-Feedback Form," *IEEE TRANSACTINS ON NEURAL NETWORKS AND LEARNING SYSTEMS,* vol. 26, no. 10, pp. 2535-2549, 2015.

[46] D. Vrabie and F. Lewis, "Neural Network Approach to Continous-Time Direct Adaptive Optimal Control for Partially Unknown Nonlinear Systems," *Neural Networks,* vol. 22, pp. 237-246, 2009.

[47] K. G. Vamvoudakis and F. Lewis, "Online Actor-Critic Algorithm to Solve the Continuous-Time Infinite Horizon Optimal Control Problem," *Automatica,* vol. 46, pp. 878-888, 2010.

[48] D. Vrabie, O. Pastravanu, M. Abu-Khalaf and F. Lewis, "Adaptive Optimal Control for Continuous-Time Linear Systems Based on Policy Iteration," *Automatica,* vol. 45, pp. 477-484, 2009.

[49] D. Liu, H. Li and D. Wang, "Online Synchronous Approximate Optimal Learning Algorithm for Multiplayer Nonzero-Sum Games With Unknown Dynamics," *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* vol. 44, no. 8, pp. 1015-1027, 2014.

[50] D. Vrabie, K. Vamvoudakis and F. Lewis, "Adaptive Optimal Controllers Based on Generalized Policy Iteration in a Continuous-Time Framework," in *17th Mediterranean Conference on Control and Automation*, Thessaloniki, 2009.

[51] D. Nodland, A. Ghosh, H. Zargarzadeh and S. Jagannathan, "Neuro-Optimal Control of Helicopter UAVs," in *SPIE: Unmanned Systems Technology XIII*, 2011.

[52] J. Na and G. Herrmann, "Online Adaptive Approximate Optimal Tracking Control with Simplified Dual Approximation Strucure for Contiuous-Time Unknown Nonlinear Systems," *IEEE/CAA Journal of Automatica Sinica,* vol. 1, no. 2, pp. 412-422, 2014.

[53] A. Das, F. Lewis and K. Subbarao, "Backstepping Approach for Controlling a Quadrotor Using Lagrange Form Dynamics," *Journal of Intelligent Robot Systems,* vol. 56, pp. 127-151, 2009.

[54] A. Das, K. Subbarao and F. Lewis, "Dynamic Inversion with Zero-Dynamics Stabilisation for Quadrotor Control," *IET Control Theory and Application,* vol. 3, no. 3, pp. 303-314, 2009.

[55] F. L. Lewis, S. Jagannathan and A. Yesildirek, Neural Network Control of Robot Manipulators and Nonlinear Systems, London: Taylor and Francis, 1999.

[56] L. A. Zadeh, "FUZZY LOGIC = Computing with Words," *IEEE TRANSACTIONS ON FUZZY SYSTEMS,* vol. 4, no. 2, pp. 103-111, 1996.

[57] N. K. Sinha and M. M. Gupta, Soft Computing and Intelligent Systems, ACADEMIC PRESS, 2000.

[58] L.-X. Wang, "Stable Adaptive Fuzzy Controllers with Application to Inverted Pendulum Tracking," *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS-PART B,* pp. 677-691, 1996.

[59] R. F. Stengel, "Toward Intelligent Flight Control," *IEEE Transactions on Systems, Man, and Cybernetics,* vol. 23, no. 6, pp. 1699-1717, 1993.

[60] Y. Li and S. Song, "A Survey of Control Algorithms for Quadrotor Unmanned Helicopter," in *IEEE International Conference on Advanced Computational Intelligence*, Nanjing, 2012.

[61] M. Cutler and J. P. How, "Actuator Constrained Trajectory Generation and Control for Variable-Pitch Quadrotors," in *AIAA Guidance, Navigation, and Control Conference*, Minneapolis, 2012.

[62] A. Mekky and O. R. Gonzalez, "LQ control for the NASA learn-to-fly free-to-roll project," in *Aerospace and Electronics Conference (NAECON) and Ohio Innovation Summit (OIS), 2016 IEEE National*, Dayton, 2016.

[63] J.-J. Slotine and W. Li, Applied Nonlinear Control, Prentice Hall, 1991.

[64] M. Krstic, P. Kokotovic and I. Kanellakopoulos, Nonlinear and Adaptive Control Design, New York: Wiley, 1995.

[65] G. M. Hoffmann, H. Huang, S. L. Waslander and C. J. Tomlin, "Precision Flight Control for a Multi-Vehicle Quadrotor Helicopter Testbed," *Elsevier: Control Engineering Practice,* vol. 19, no. 9, pp. 1023-1036, 2011.

[66] S. Bouabdallah, P. Murrieri and R. Siegwart, "Design and Control of an Indoor Micro Quadrotor," in *IEEE International Conference on Robotics and Automation*, Lausanne, 2004.

[67] G. Chowdhary, T. Wu, M. Cutler, N. K. Ure and J. P. How, "Experiment Results of Concurrent Learning Adaptive Controllers," in *AIAA Guidance, Navigation, and Control Conference*, Minneapolis, 2012.

[68] A. K. Shastry, A. Pattanaik and M. Kothari, "Neuro-Adaptive Augmented Dynamic Inversion Controller for Quadrotors," in *4th IFAC Conference on Advances in Control and Optimization of Dynamical Systems* , Tiruchirappalli, 2016.

[69] P. Castillo, R. Lozano and A. Dzul, "Stabilization of a mini-rotorcraft having four rotors," in *IEEE International Conference on Intelligent Robots and Systems*, Sendai, Japan, 2004.

[70] T. Madani and A. Benallegue, "Backstepping Control for a Quadrotor Helicopter," in *IEEE International Conference on Intelligent Robots and Systems*, Beijing, 2006.

[71] S. Bouabdallah and R. Siegwart, "Backstepping and Sliding-Mode Techniques Applied to an Indoor Micro Quadrotor," in *IEEE International Conference on Robotics and Automation*, Barcelona, 2005.

[72] G. Joshi and R. Padhi, "Robust Control of Quadrotors using Neuro-Adaptive Control Augmented with State Estimation," in *AIAA Guidance, Navigation, and Control Conference* , Grapevine, 2017.

[73] A. P. Engelbrecht, Computational Intelligence: An Introduction, Chichester: John Wiley & Sons, Ltd, 2007.

[74] A. K. Jain, J. Mao and K. M. Mohiuddin, "Artificial Neural Networks: A Tutorial," *Computer,* vol. 29, no. 3, pp. 31-44, 1996.

[75] A. S. Zinober, Deterministic Control of Uncertain Systems, London: Peter Peregrinus Ltd., 1990.

[76] I. Kanellakopoulos, P. V. Kokotovic and A. S. Morse, "Systematic Design of Adaptive Controllers for Feedback Linearizable Systems," *IEEE Transactions on Automatic Control,* vol. 36, no. 11, pp. 1241-1253, 1991.

[77] E. Lavretsky and K. A. Wise, Robust and Adaptive Control: with Aerospace Applications, New York: Springer, 2013.

[78] F. Lewis, D. M. Dawson and C. T. Abdallah, Robot Manipulator Control: Theory and Practice, New York: Marcel Dekker, 2004.

[79] F. Lewis, D. M. Dawson and C. T. Abdallah, Robot Manipulator Control: Theory and Practice, New York: Marcel Dekker, 2004.

[80] R. Mahony, T. Hamel and A. Dzul, "Hover Control via Lyapunov Control for an Autonomous Model Helicopter," in *The 38th Conference on Decision & Control*, Phoenix, Arizona USA, 1999.

[81] K. Hornik, M. Stinchcombe and H. White, "Multilayer Feedforward Networks are Universal Approximators," *Neural Networks,* vol. 2, pp. 359-366, 1989.

[82] Quanser, "Quanser Qball-X4: User Manual," Doc. 888, Rev. 2.

[83] B. R. Turnbaugh, "Extending Quad-Rotor UAV Autonomy with Onboard Image Processing," Naval Postgraduate School, Monterey, 2015.

[84] S. Ahrens, D. Levine, G. Andrews and J. P. How, "Vision-based guidance and control of a hovering vehicle in unknown, GPS-denied environments," in *IEEE International Conference on Robotics and Automation*, Kobe, 2009.

[85] H. Durrant-Whyte and T. Bailey, "Simultaneous Localisation and Mapping (SLAM): Part I," *IEEE Robotics & Automation Magazine,* vol. 13, no. 2, pp. 99-110, 2006.

[86] T. Bailey, J. Nieto and E. Nebot, "Consistency of the FastSLAM Algorithm," in *IEEE International Conference on Robotics and Automation*, Orlando, 2006.

[87] "Parrot SA- Wikipedia," Wikipedia, 7 September 2017. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Parrot_SA&oldid=799462511. [Accessed 7 January 2018].

[88] "Parrot S.L.A.M.dunk | Official Store," Parrot SA, 2017. [Online]. Available: https://www.parrot.com/hu/en/business-solutions/parrot-slamdunk#parrot-slamdunk. [Accessed 7 January 2018].

[89] T. Flash and N. Hogan, "The Coordination of Arm Movements: An Experimentally Confirmed Mathematical Model," *The Journal of Neuroscience,* vol. 5, no. 7, pp. 1688-1703, 1985.

[90] H. K. Khalil, Nonlinear Systems, New Jersy: Prentice Hall, 2002.

VITA

Ahmed Elhussein Eltayeb Mekky

+1(757) 701-5806

ahmed.e.e.mekky@gmail.com

EDUCATION

Old Dominion University, VA, USA (2012-2018)
Doctor of Philosophy in Aerospace Engineering,
Major: Dynamic Systems and Control, **GPA:** 3.92

Funded as a research and teaching assistant. Dissertation Title 'Design and Implementation of an Artificial Neural Network Controller for Quadrotor Flight in Confined Environment'.

Old Dominion University, VA, USA (2010-2012)
Master of Science in Aerospace Engineering,
Major: Dynamic Systems and Control, **GPA:** 3.88

Funded as a research and teaching assistant. Thesis Title 'Modeling, Identification, Simulation and Control of a Hybrid MAGLEV Ball System'.

University of Khartoum, Khartoum, Sudan, (2003-2007)
Bachelor of Science in Mechanical Engineering,
Major: Mechanical Engineering, First Class

Ranked second in a batch of 100+ students at the department of Mechanical Engineering, University of Khartoum, based on Accumulative GPA, 2007. Graduation project Title 'Gas Turbine Power Augmentation Using Inlet Air Cooling'.

Khartoum North Governmental High School, Khartoum, Sudan, (2000-2002)

Ranked the first schoolwide, based on the academic achievement in the national exam for high schools (2002).

EXPERIENCE

Old Dominion University, Mechanical and Aerospace Engineering Dept., Norfolk, VA, USA
Graduate Research and Teaching Assistant, Aug. 2010 to May 2018

University of Khartoum, Mechanical Engineering Dept., Khartoum, Sudan
Teaching Assistant, July 2008 to Aug. 2010

National Electricity Corporation, Khartoum, Sudan
Mechanical Engineer, 2009 to 2010

AREAS OF EXPERTISE:

- Dynamic Systems modeling, identification, controller design and closed loop performance analysis
- Atmospheric and Space flights Dynamics modeling, Controller design and Performance analysis
- Control systems stability and robustness analysis, and Real-Time implementation
- Linear controller design techniques and performance analysis; in time and frequency domains
- Nonlinear, Optimal, Robust, and Neural Network controller design methodologies

PUBLICATIONS & PRESENTATIONS

Presented and Published: Mekky A., Alberts T.," **Modeling, Identification, Validation and Control of a Hybrid Maglev Ball**", Dynamic Systems and Control Conf., ASME, 2012.

- Invited for an oral presentation at 2nd International Symposium on Energy Challenges & Mechanics, Aberdeen, Scotland, 2014.

Presented and Published: Mekky A., Gonzalez O., "**LQ Control for the NASA Learn-to-Fly Free-to-Roll Project**", National Aerospace & Electronics Conference & Ohio Innovation Summit, IEEE, 2016.

Future Publication: Mekky A., Alberts T., "**Design of an Artificial Neural Network Controller for Quadrotor Flight in the presence of Aerodynamic uncertainties**", Ready for submission.

Future Publication: Mekky A., Alberts T., Gonzalez O., "**Experimental Implementation of an Artificial Neural Network Controller for Quadrotor Flight in the Presence of Aerodynamic Uncertainties**", Journal of Navigation Guidance and Control, AIAA.

HONERS & AWARDS

- SHELL Co. Prize for the best academic performance in the academic year 2006-2007, Mechanical Engineering Department, University of Khartoum, Second place.
- Golden Key International Honor Society, since 2011, Member.
- The Dr. Robert A. and Ronnie Slocum Magoon Scholarship for Aerospace Engineering. ODU 2014-2017
- Prabhav Mniyar Endowed Scholarship. ODU 2016-2018
- The Oktay Baysal Endowed Graduate Scholarship in Computational Engineering for Aerospace. ODU 2017-2018

REFERENCES

Thomas Alberts, PhD
Professor
Old Dominion University
Mechanical and Aerospace Department
1313 ENGR. & COMP. SCI. BLDG.
NORFOLK, VA 23529
757-683-3736, talberts@odu.edu
Relationship: MSc, PhD advisory committee director

Brett Newman, PhD
Professor
Mechanical and Aerospace Department
1317 ENGR. & COMP. SCI. BLDG.
Norfolk, VA 23529
757-683-5860, bnewman@odu.edu
Relationship: MSc, PhD advisory committee member

Oscar R. Gonzalez, PhD
Professor
Electrical and Computer Engineering Department
231H KAUFMAN HALL
NORFOLK, VA 23529
757-683-4966, ogonzale@odu.edu
Relationship: PhD advisory committee member