Ministry of Higher Education and Scientific Research
University of Mohamed Khider - Biskra
Faculty of Science and Technology
Department of Electrical Engineering

MODELING AND INTELLIGENT CONTROL OF A DRONE

by

Oussama Bouaiss

Submitted to the Department of Electrical Engineering
in partial fulfillment of the requirements for the degree of

PHD in Automatics

Jury

Pr.	Mohamed Toufik Benchouia	University of Mohamed Khider	Jury president
Pr.	Raihane Mechgoug	University of Mohamed Khider	Thesis supervisor
Pr.	Mohamed Yacine Hammoudi	University of Mohamed Khider	Examiner
Pr.	BenGuesmia Hani	University of Mohamed Boudiaf	Examiner

2022/2023

Ministry of Higher Education and Scientific Research

University of Mohamed Khider - Biskra

Faculty of Science and Technology

Department of Electrical Engineering

MODELING AND INTELLIGENT CONTROL OF A DRONE

by

Oussama Bouaiss

Submitted to the Department of Electrical Engineering
on July, 2023, in partial fulfillment of the
requirements for the degree of
PHD in Automatics

Abstract

This thesis tackles the modeling, design, and control of a Quadrotor unmanned aerial vehicle, with a focus on intelligent control and smart applications such as obstacle avoidance, robust trajectory tracking, visual soft landing, and disturbance compensation. It details the mathematical modeling opted for the simulation and the control. Furthermore, It describes the classic control methodology for both linear and nonlinear control techniques with interpreted simulations; The methodology is subsequently applied to develop an open-source autonomous quadrotor miniature model. In addition, advanced control theory has been applied using Adaptive Linear Quadratic Gaussian, Model predictive control, and intelligent Radial basis functions neural network for the robust tracking of generated trajectory for either obstacle avoidance or bio-inspired soft landing on a specially designed landing pad. The thesis depicts as well the adaptive optimal observation by an enhanced Kalman filter combined with Madgwick sensor's data fuse. Control laws were mainly either mathematically derived or adaptively generated based on stability analysis using Lyapunov theory, The simulation incorporated several analytical comparisons to prove efficiency and compare the performance.

Keywords: Quadrotor, Adaptive control, Extended Kalman Filter, Soft landing, Obstacle avoidance, Model predictive control, Linear Gaussian control, Radial basis function neural network.

الملخص

تتناول هذه الأطروحة النمذجة والتصميم والتحكم في مركبة جوية بدون طيار رباعية المحرك ، مع التركيز على التحكم الذكي والتطبيقات الذكية مثل تجنب العوائق والتتبع المتين للمسار والهبوط البصري السلس وتعويض الاضطرابات. يعرض تفاصيل النمذجة الرياضية المختارة للمحاكاة والتحكم. علاوة على ذلك ، يصف منهجية التحكم الكلاسيكية لكل من تقنيات التحكم الخطية وغير الخطية مع المحاكاة المفسرة ؛ يتم تطبيق المنهجية لاحقًا لتطوير نموذج مصغر مستقل مفتوح المصدر للطائرات الرباعية المحرك. بالإضافة إلى ذلك ، تم تطبيق نظرية التحكم الذكية باستخدام التحكم الغاوسي الخطي التكيفي ، والتحكم التنبئي النموذجي ، والشبكة العصبية الذكية للوظائف الأساسية الشعاعية من أجل التتبع المتين للمسار المتولد من أجل تجنب العوائق أو الهبوط الناعم المستوحى من الطبيعة على منصة هبوط مصممة خصيصًا. يصور أيضا التقدير الأمثل التكيفي بواسطة مرشح كالمان المحسن. جنبا إلى جنب مع دمج بيانات مستشعر مادجويك. تم اشتقاق قوانين التحكم بشكل رئيسي رياضيًا أو تم إنشاؤها بشكل تكيفي بناءً على تحليل الثبات باستخدام نظرية ليابونوف للإستقرار ، وقد تضمنت المحاكاة العديد من المقارنات التحليلية لإثبات الكفاءة ومقارنة الأداء.

الكلمات المفتاحية رباعية المحرك ، التحكم التكيفي ، مستشعر كالمان الممتد ، الهبوط السلس ، تجنب العوائق ، التحكم التنبئي للنموذج ، التحكم الغاوسي الخطي ، الشبكة العصبية لوظيفة الأساس الشعاعي.

Thesis Supervisor: Mechgoug Raihane
Title: Associate Professor at the Department of Electrical Engineering

Acknowledgments

The University of Mohamed Khider is acknowledged for providing Laboratory and facilities for the successful completion of this Ph.D. project. Special thanks to all who were involved in this research. The author declares that there are no conflicts of interest as regards this thesis.

Dedication

To my beloved family, For your unwavering love, encouragement, and sacrifices, This thesis is dedicated to you.

Thank you for being my source of strength, For believing in my dreams and supporting my journey.

Your presence has been my inspiration, And your support has fueled my determination.

To my professors Pr.Mechgoug, Pr. Taleb-Ahmed, and Pr. Terki, you have been always there during my journey.

With heartfelt gratitude.

In loving memory of my dear uncle Said, I wish you were here.

OUSSAMA Bouaiss

This doctoral thesis has been examined by a Committee of the Department of Electrical Engineering as follows:

Contents

List of Figures

List of Tables

List of articles and publications

Table 1: Published articles

Articles	Journal	DOI
Modeling, Control and Simulation of Quadrotor UAV	2020 1st International Conference on Communications, Control Systems and Signal Processing (CCSSP) **(IEEE)**	10.1109/CCSSP49278.2020.9151687
Visual soft landing of an autonomous quadrotor on a moving pad using a combined fuzzy velocity control with model predictive control	Signal, Image and Video Processing 23 April 2022 **(Springer Nature - A Class)**	https://doi.org/10.1007/s11760-022-02199-y
Optimal Control of Quadrotor with a Novel Madgwick/Extended Kalman Observer to Track a Spline Trajectory for Obstacle Avoidance	Iranian Journal of Science and Technology, Transactions of Electrical Engineering 12 October 2022 **(Springer Nature - A Class)**	https://doi.org/10.1007/s40998-022-00552-y
Adaptive Neural Network-Based Compensation Control Of Quadrotor For Robust Trajectory Tracking	International Journal of Adaptive Control and Signal Processing 13 July 2023 **(Wiley - A Class)**	https://doi/abs/10.1002/acs.3659

Table 2: Under-processing articles

Articles	Journal	DOI
Robust Trajectory Tracking Using Adaptive RBFNN Control	Franklin Institute Journal **(Elsevier : A+ Class)**	Not yet attributed

Chapter 1

Introduction

"Helicopters don't fly, they vibrate so badly the ground rejects them."

Tom Clancy

1.1 Motivations and Objectives

In recent years, the focus has been directed to Unmanned Aerial vehicles (UAV)s, its domain has received countless research and academic works. Their use has become a sort of indispensable task in many sectors, which led to a massive investment of time, money, and technology to develop algorithms and strategies for sophisticated systems of control. Essentially, UAVs had been made for Reconnaissance, Intelligence, Surveillance, and Target Acquisition (RISTA), however, recent years had shown more modalities of use especially in military and commercial sectors as in delivery applications. Quadrotors as one of the important UAV models show a big advantage by stationary ability during the hovering mode although it is considered an inefficient mechanism in matter of energy consumption relative to fixed wings platforms. The techno-scientific challenge to design and control a quadrotor robustly by applying intelligent approaches to achieve beyond-imagination applications was very motivating. Modeling techniques besides controlling approaches of multirotor with hardware development had all contributed to gaining better performance, endurance, and even to executing acrobatic maneuvers and sophisticated trajectories [19], which opened up more doors for new challenges and designs.

Advanced Microelectroeechanical Systems (MEMS) sensors, micro-ships, & Global positioning system (GPS) receivers allowed an essential step toward mainstream applications [39][65]. Moreover, commercial and civilian use represent an interesting domain where many applications were developed such as in agriculture, geology, climate & weather studies, mapping of open and confined environments, delivery, and even for entertainment and tourism [93][32]. Yet, fixed wings UAVs are mechanically more efficient mechanisms energy-wise, owning the heritage of classic aircraft design, Quadrotors among the family of multirotor had received a huge contribution and attention due to hovering mode, and the ability of Vertical Taking Off and Landing (VTOL) [16]. Furthermore, indoor capabilities and agility in confined spaces are more than to be matched by other types. Maneuverability and acrobatic abilities were tackled also as a performance objective [12], it's significant to state that quadrotors commercial business topped a budget of 22 billion dollars [93].

A Quadrotor is a maneuverable highly nonlinear model of UAVs that falls into the class of underactuated systems, with Six Degrees Of Freedom (DOF) and is energized by four rotors with fixed-pitch propellers that rotate at different controlled speeds [76]. In addition, the quadrotor presents tremendous uncertainties in modeling, identification, and design parameterization, despite neglecting the external disturbance of environment and noise of actuators and sensors.

Here comes the challenge, where intentions were made to address a performant control, robust tracking, and smart application and abilities applying intelligent methods. Diverse approaches have been considered in the literature and research about the control philosophy of quadrotors [76]. Advanced control theory contributed significantly to overcoming the difficulties in a variety of designs of autonomous vehicles [65]. Adaptive designs and platforms were more developed for the aspects of specific tasks and applications. Early works about quadrotor development focused just on a reliable control [65][19]. Conversely, recent papers present a deeper sight of tasks and

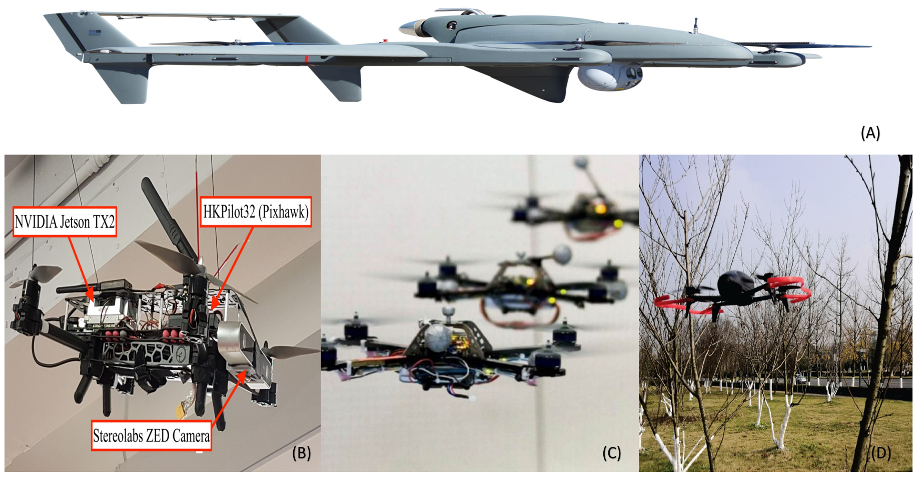


Figure 1-1: Applications on quadrotors in different projects (A).[5], (B).[100] (C).[95] and (D).[26]

control performance such as 3D mapping [48], Simultaneously Localization And Mapping (SLAM) [30] [26], trajectory generation, and tracking [113][81][107] [97], sparse and swarm control [67][15], augmented and Hybrid systems, and autonomous landing [10][41][21], etc. Fig.1-1 illustrates some interesting projects about quadrotors.

The essence of this work is to present an assessment of essential strategies of quadrotors design and intelligent control, rising up from the mathematical modeling of the quadrotor, by the validation of the nominal model, then clarifying the classic techniques of control with detailed schemes, Algorithms, and simulations with building an open source model, to the derivation of intelligent and modern architecture of control and tracking. Trajectory generation was well investigated, as the sophisticated application such as soft landing and obstacle avoidance in addition to robust compensation of disturbance.

1.2 State of the art

Quadrotors are classified as rotorcrafts typology, A widely known structure of a Quadcopter is shown as crossed bars of the body, with several motor/propeller installed at the arms ends, The four rotors are configured into clockwise and counter-clockwise rotating pairs; thereby canceling the horizontal spinning (Yawing). The rotation of propellers generates enough thrust to lift and rotate the drone around the main three axes of rotation.

Quadrotor UAVs are defined as aircraft with the approbation of communication and control with no man on board [65], this latter opens the possibility of either an autonomous control system or being remotely controlled, which can be managed by a ground pilot with a communication post. Yet, all UAVs are equipped with a Flight Control Unit (FCU) to ease the exploitation of the Inertial Measurement Unit (IMU) [51] and other sensors.

Historically, quadrotors were attracting people since the earlier of 20th century, named at that time by Breguet-Richet Gyroplane as in Fig.1-2 [106], In 1922, George de Bothezat and Ivan Jerome made another attempt to develop the copter philosophy for more applicability than what has been done previously [25]. In 1956, an approved flight of quadrotor test showed the practicality of such a design even though it did not get a successful commercialization. Later on, Curtiss Wright company constructed the VZ-7 model for the US Army in 1958, which was considered the real ancestor of nowadays design. Researches had a big push by the boom of electronics of control and technology. Far from this ancient design, the quadrotor recently opted for a miniature scale, Vijay proved that a small-scale UAV shows more stability and ease of control than a larger one [93]. Resources have been set to develop and designs applications of quadrotors industrialization, which exploded the budget of investment [31], especially, with the commercial perspective that boosts Numerous universities and companies to develop research centers to improve this sector and look for particular applicability, especially after the Russian-Ukrainian war.

In literature, Control of quadrotors tackled many essential axes as detailed in Fig.1-3, at the same time, from a control theory point of view, it was dispatched between linear, nonlinear, and intelligent methods, all applied to the quadrotor model to achieve either high tracking performance, obstacle avoidance, navigation, and autonomous exploration and tasks.

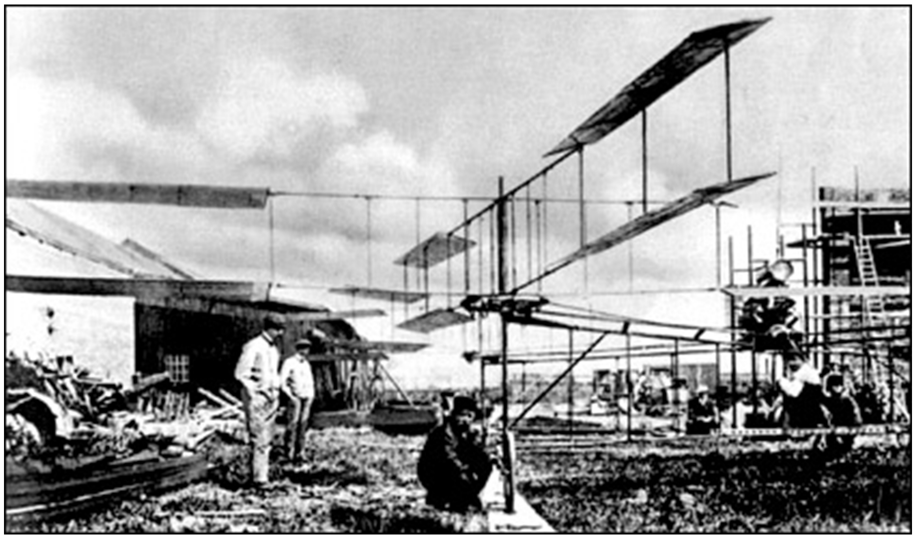


Figure 1-2: Breguet-Richet Gyroplane [106]

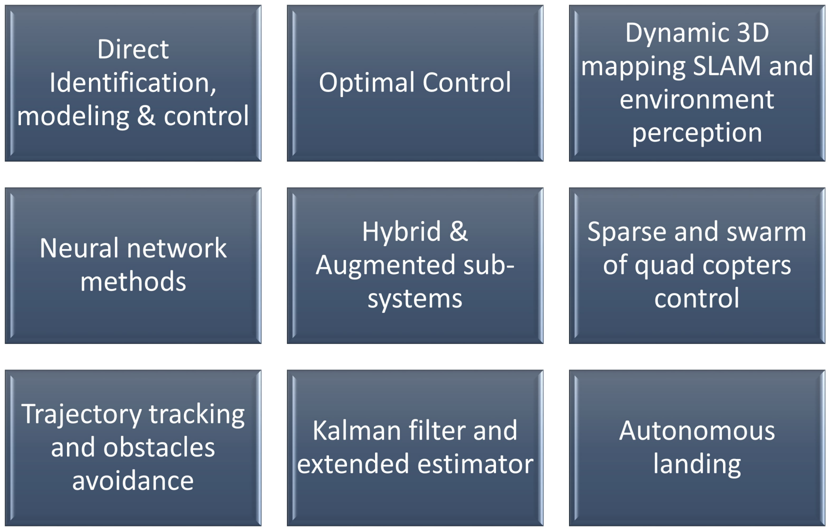

Direct Identification, modeling & control
Optimal Control
Dynamic 3D mapping SLAM and environment perception
Neural network methods
Hybrid & Augmented sub-systems
Sparse and swarm of quad copters control
Trajectory tracking and obstacles avoidance
Kalman filter and extended estimator
Autonomous landing

Figure 1-3: Main axis of researches and applications

For the linear procedure, Proportional Integral Derivative (PID) Controller was always the easiest & robust way to attribute quadrotor control [19][78] [22] [89], Roger et al. presented a novel robust PID economic control [71]. Linear Quadratic Regulator (LQR) for stability and tracking was proven robust in [68]. Linear H infinite opted for robustness in [87]. Nonlinear techniques were approached in numerous typologies, model inversion strategy was applied for quadrotor control in [6], and with a suspended load in [29], feedback linearization was opted for by many researchers as its simplicity, however, it is prone to instability in many cases due to non-invertibility. Backstepping method was highly investigated using a variety of architecture as it's not that complicated in implementation on hardware, a flip maneuver of a quadrotor using Backstepping approach with a gain update was opted in [12], it was used too to guarantee UAV's trajectory tracking [55]. The Sliding mode was subjected by many researchers and showed robust control and tracking as shown in [47][40][71]. Linear techniques can not ensure neither convergence not performance once we move away from the operating envelope, nonlinear techniques rely extremely on dynamics modeling, which opens the doors for instability due to unmodeled effects and external forces, in addition, both of them present a limited ability to adapt to uncertainties in matter of performance.

Intelligent control methods were considerably investigated as well, Fuzzy Logic Controller (FLC) was designed for quadrotors hovering & tracking modes in [53]. Model Predictive Control (MPC) has been applied to ensure nontrivial maneuvers and obstacle avoidance for quadrotor UAV under disturbances [62] and for autonomous landing [20]. Neural Network (NN) methods were also contributing to quadrotor development, as we may talk about direct nonlinear model identification, obstacle avoidance via Convolution Neural Network (CNN)-based learning, Dai et al. performed experimentation of SLAM & SFM algorithms on PARROT BEBOP 2 using vision camera based on CNN prediction for depthwise [26]. Huan et al. used NN Control based on Reinforcement learning optimized by Gradient algorithm, Implementation later of the trained model was directly implemented in Pixhawk flight controller model [82]. Another approach was based on using Diagonal Recurrent Neural Network (DRNN) for updating PID gains in [78]. Magdwick et all developed a faster data fusion of sensor for attitude feed back based on mathematical optimization of quaternion equations [66], in this thesis we will develop an enhance Extended Kalman Filter (EKF) combined with Madgwick method.

The majority of quadrotors designs take advantage of EKF for optimal full state vector estimation from measurements, which was concluded to be an effective method in autonomous navigation systems that showed robustness under disturbance [102] [50], or in the presence of white Gaussian noise as shown in [102], SLAM algorithm uses the advantage of the data driven by the IMU estimator for variety of applications, such as bio-inspired autonomous landing [30]. Guo et al. experimented EKF observer to enhance PID based H infinity control of multiple quadrotors [41], an evolutionary algorithm with principles of KF to tune unknown noises was optimized in [50], other researchers have used virtual sensing based on EKF observer to develop a quadrotor controller [78].

A vast domain of applications is carried out about Quadrotors, far away from projecting control theory over attitude behavior, trajectory generation & path following were engaged in many papers, Alcocer et al. considered model-based mathematical optimization of trajectory with high performance compared with respect to three different classical known trajectory tracking controllers [81]. Tracking of trajectory subject to uncertain inertial & mass parameters was considered in [113]. It has been proven that there are plenty of procedures to generate optimal trajectories far away from the complexity and expensive time of optimal control model [20].

Obstacles avoidance was a major criterion in trajectory design & tracking, hence, many philosophies were approached, FLC was proven to be a good tracking strategy for quadrotors, with Lyapunov analysis to investigate the exponential stability [53], Chang et al. designed tracking based on PID with Active Disturbance Rejection Control (ARDC). Obstacle detection was achieved by sensors whilst a proactive generation of the ellipsoid potential field was avoided. Other Researchers opted for a trajectory generation first using quadratic programming with Linear constraints, then, The control of tracking was decided in a closed loop scheme with double integrator Lyapunov control, following minimal SNAP (4th derivative) and minimal of JERK (3rd derivative) [107], results of the flip experiment are shown in Fig. 1-4 .

The Backstepping technique also opted for trajectory tracking with avoidance of singularities by emphasizing the quaternion model [97]. It is mandatory to reveal that path following problem is defined when implementing a controller that tracks trajectory without time referencing. Those controllers discuss more the feasibility as

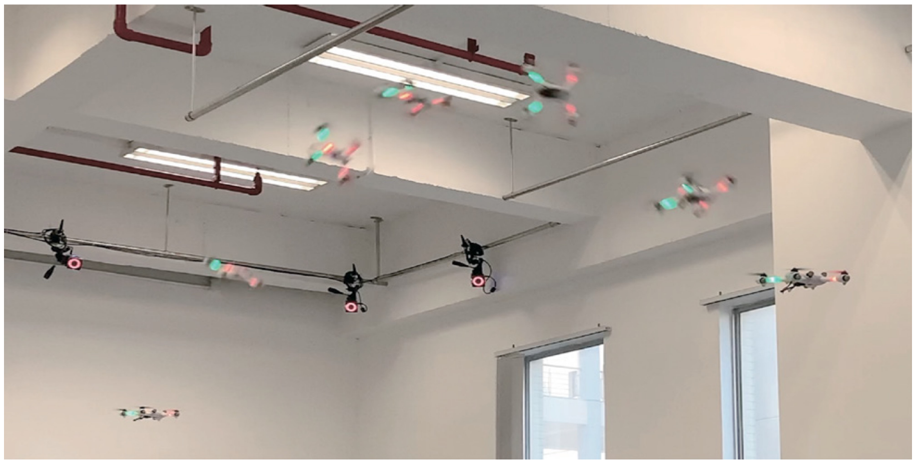


Figure 1-4: Sequence of attitude snapshots of the quadrotor performing a 360 flip [107]

the realization of stable following along the generated path, Löber, demonstrated the realisability of Optimal Trajectory Tracking of Nonlinear Dynamical Systems [63].

Trajectory optimization had another research area where researchers worked on the control of multiple and swarm of quadrotors, distributed finite time formation tracking control problem with collision avoidance was investigated for multiple quadrotor UAVs subject to external disturbances by Huang et al. focus was on mounting a challenge of Disturbance estimation with a Sliding Mode Control (SMC) to follow the desired trajectory generated by Optimal Control Problem (OCP) as they proved Lyapunov stability [47]. Borkar et al. exploited Lissajous curves for the pre-planning of trajectories for Re-configurable formations of quadrotors for surveillance applications [15], another group of researchers tackled the zone search optimization of a quadrotors flock by adhering to a bio-inspired methodology & considering the interaction between Quadcopters to avoid obstacles and collision using PID control applied on swarm algorithm, the authors proved the advantage of Multi Object Navigation Optimization of the dynamic model with other techniques [67], while others regarded the multiple quadrotor cooperative control subject to aerodynamics drag & position deviations, the paper written by Yue et al. suggested a leader drone to be followed based on adaptive SMC [110]. Same leader-followers philosophy was maintained by Liu et al. in [60] where they opted for visually servoing Formation Tracking Control for Quadrotor UAVs Team, the authors studied the following patterns, cooperative work, a sudden change of leader process.

Navigation in an unknown environment for quadrotors was studied as well referred as Guidance, Navigation and Control (GNC) algorithms, GNC got some researches contributions, vision-based localization was used as well as an option for 3D positioning [92] [80], building a 3D maps using SLAM was also simulated by Jiang et al. for a quadrotor in Gazebo data set combined by Robot Operating System (ROS) [48], SLAM combined with state estimation through INS was researched also, for bio-inspired autonomous landing using a Kinect camera [30]. Another study was about the Stereo vision for a quadrotor to avoid single and multiple obstacles defined in the ellipsoidal bounding box [80].

Recently, NN methods have been used in quadrotors for identification and obstacle avoidance. Structure From Motion (SFM) with SLAM enabled by Reinforcement learning-based approaches considering a CNN scheme for obstacle avoidance was studied [26]. DRNN for updating PID gains strategy was implemented successfully in [78].

Augmented and hybrid mechanical systems based on the quadrotor model were also investigated in the literature. Nonlinear Lyapunov approach for hybrid dynamics (Quadrotor + load) focused on slung load attached to quadrotor trajectory tracking issue was performed reliably [21]. Dynamic inversion of the quadrotor with a Suspended Load was modeled by applying Newton's laws, then the author opted for control via dynamic inversion [29]. Adding robot arms to the quadrotor model was one of the interesting subjects, as a 2 DOF arm was a PID methodology controlled for manipulative quadrotor [51]. Swing attenuation for quadrotor transporting a cable-suspended payload problem was modeled and controlled via two approaches depending on swing angle; the authors used Interconnection and Damping Assignment-Passivity Based Control (IDA-PBC). OCP has opted in many papers to track generated trajectories for quadrotors subject to time minimization or avoidance of contact [68] [47]. Divers open source models of quadrotors were developed, either using Pixhawk FCU, a developed model based on Arduino Mega with data fusion of IMU sensors was investigated and explained.

1.3 Contribution of this Work

The thesis focuses essentially on the modeling, and intelligent control of an autonomous quadrotor with smart applications based on optimal sensors data fusion and robust trajectory tracking. The contribution of this work lies in numerous areas:

1. **In a matter of modeling:** mathematical modeling of quadrotors dynamics: derivation of equations by Newton and Euler formalism to obtain a valid mathematical representation of flying dynamics. In addition to demonstrating step by step of the development of classic methods of control with all schemes, Algorithms, and simulations, where we have Carried out and provided a real depth for the simulation by considering the wind disturbance that affects the general position and high-frequency noise of actuators and sensors thru Gaussian white noise. A 3D simulation environment to visualize the quadrotor flight results to ease the perception of control effectiveness was built in Matlab Sim-Mechanic environment.

2. **In a matter of Conception and design:** Implementation and realization of an open source quadrotor model with detailed algorithms based on Arduino Mega development card. Algorithms, material, and sensors were all detailed. Codes, Simulink models, and programs are publicly available for researchers to be exploited as a benchmark for their studies.[1]

3. **In a matter of Optimal observer:** A Novel faster robust Attitude and Heading Reference system (AHRS) algorithm of asynchronous data fusion filter to estimate orientation based on merging EKF and Madgwick approach enhanced by compensation of bias and calibration of distortion with the integration of GPS localization data. The use of the Adaptive EKF to mitigate sensing noise destabilizing effect, with the demonstration of stability analysis by Lyapunov theory.

[1]GitHub codes.

4. **In a matter of adaptive control:** Design of an adaptive LQG tracker for the UAV that exploits AHRS-EKF algorithm to track a 3D Spline trajectory with disturbance rejection and cancellation of the Gaussian random noise, both represented by the wind gust disturbance that affects the general position and the high-frequency noise of actuators and sensors using various Gaussian white noises. with the Comparison of adaptive LQG control with nested loop design PID, Integral Backstepping, and FLC control performances.

5. **In a matter of visual detection and control:** Design a robust application of the pose estimation based on a special landing pad by multiple ArUco markers with different patterns at different sizes to ease the detection.

6. **In a matter of optimal control:** Derive the MPC for position tracking with attenuation of external disturbance and performance comparison to PID, MPC is Combined by an EKF & Magdwick Data fusion of asynchronous sensors for state observation.

7. **In a matter of Soft landing:** Planning a mimic of natural decision making as expert human processing for landing by incorporating the FLC for velocity control.

8. **In a matter of Intelligent Control:** The Conception of a nested NN design based on adaptive RBFNN to control position in the outer loop and a supervised NN to control the attitude that carries out superior performance such as a faster convergence and capable of disturbance rejection and stabilize attitude at the initial phase, with higher precision in comparison to some nonlinear and intelligent control approaches which are limited to robustness against the assessed type of noise and perturbation.

9. **In a matter of performance and robustness:** Comparison of different algorithms of control with the opted RBF strategy relative to the tracking error and robustness to disturbance and unmodeled effects. a PID, an IBS based on nonlinear dynamics, and a decentralized offline trained Multi Layer Perceptron

Neural Network (MLPNN) have been all of them evaluated in comparison. In Matlab simulation, motors dynamics and mechanical noise have been all added to the model.

1.4 Developed Applications

Despite the realization of an open source model, Two main intelligent applications have been built during the research period.

First the Visual Soft Landing Of An Autonomous Quadrotor On A Moving Pad Using A Combined Fuzzy Velocity Control with MPC [20], where we demonstrated a sophisticated solution for the soft landing application of a fully autonomous quadrotor on a moving pad considering external disturbance, model uncertainties & actuators noise. The specially designed landing pad was initially detected by an onboard vision system with a robust Algorithm to estimate its coordinates precisely by a fusion of the camera pose estimation with INS filters. The desired trajectory for waypoint and landing is dynamically generated based on jerk optimization, integrating a bio-inspired velocity profile by FLC with a position loop estimator to smoothen the landing. MPC was chosen for tracking.

A second project was about the Optimal Control Of Quadrotor With a Novel Madgwick/Extended Kalman Observer To Track A Spline Trajectory For Obstacle Avoidance, where a demonstration of the adaptive Linear Quadratic Gaussian (LQG)) control of quadrotor using a novel faster full state observer based on an enhanced EKF subjected to track a remotely generated Spline trajectory for obstacle avoidance. Quaternion Orientations and Attitude and Heading Reference system (AHRS) were validated by experimental tests at less than one degree of precision error, whereas the proposed adaptive LQG control of the quadrotor was simulated for tracking control and validated in Simulink environment in the presence of disturbance.

Furthermore, Intelligent proactive compensation of disturbance has been well studied theoretically using Adaptive Radial Basis Function (RBF).

Chapter 2

Modeling of dynamics

In this thesis, the mathematical model of quadrotor UAV is derived using Newton-Euler formalism, no considerations will be taken for Hubb forces or ground effects, due to the validation of the attributed model by many researchers. Most of the used symbols are defined in Table 2.1. Moreover, most of the ignored effects are considered disturbances and model uncertainties that we treat in the control aspect.

Table 2.1: Symbols definition

Symb	Defintions	Symb	Defintions
ϕ	Roll angle	θ	Pitch angle
ψ	Yaw angle	Ω	Rotor speed
J_{TP}	Total rotors inertia	l	Lever
b	Thrust coefficient	g	Gravity
d	Drag coefficient	Ω	Rotor speed
m	Quadrotor mass	I	Inertia matrix
I_{3X3}	3 by 3 Identity matrix	0_{3X3}	3x3 Zeros matrix
T_i	Torque of motor "i"	l	lever
$R\Theta$	rotation matrix	$T\Theta$	transformation matrix
I_{xx}, I_{yy}, I_{zz}	inertia matrix diagonal in B-frame	$\ddot{\Gamma}^E$	vector of linear acceleration in E-frame
F^E	Generalized forces in E-frame	$\dot{V}^B$	linear acceleration in B-frame

2.1 Design concept

A common structure of the quadrotor is based on mounting four throttle motors on the four extremities of crossed barres frame, where every two adjacent propellers rotate in opposite directions at controlled speeds to stabilize the 6 DOF of the UAV as illustrated in Fig.2-1 . This propellers' arrangement tends to give a general change in the lift for the quadrotor when all propellers increase or decrease simultaneously their rotation speed, this global thrust is known by U_1. complementary change of two opposite propellers speeds creates a torque around one of its X_b or Y_b body frame axes, this input is known by U_2 or U_3, a total summation of all rotation creates a relative torque around the Z_b axe, referred by U_4. Roll, pitch & yaw are the three defined possible rotations around X_b, Y_b ,& Z_b respectively, where "b" notation is referring to the drone's body frame as shown in Fig. 2-1. Those rotations generate two translations along X_e & Y_e in the earth reference frame, vertical linear motion is a result of lift produced by the four propellers. This ability to perform 6 DOF motion by four actuators classifies the quadrotor into an under-actuated dynamic unstable system, with four inputs and six outputs [54] -[18]. Thus, the linear translations in X_e & Y_e directions are coupled. This fact would be trivial when observing the dynamics of quadrotor where the angles and their time derivatives do not depend on translation components. However, on the other hand, the translations do depend on the angles [40].

To develop the six DOF spatial free body equations of motion, it's common to create two frames references where orientations are well demonstrated [19]. Earth inertial frame (E frame), is defined by ground with gravity orientation to the direction of negative Z, 3D quadrotor translations would be adequate in this reference frame.

E frame: earth related reference: $E(O_e, X_e, Y_e, Z_e)$.

The body fixed frame (B frame), is defined by the orientation of the quadrotor with rotors axes parallel to the Z_b direction. X_b & Y_b axes are directed within the body arms, this frame is suitable for attitude study.

B frame: Body related reference: $B(O_b, X_b, Y_b, Z_b)$.

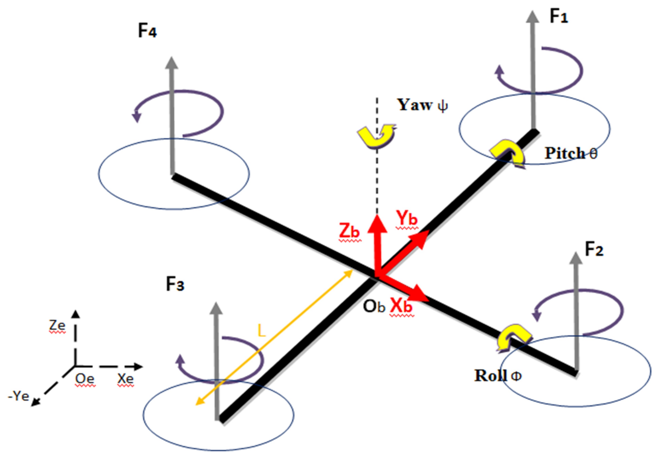


Figure 2-1: Configuration of quadrotor frames

F_1, F_2, F_3 & F_4 are the forces of thrusts generated by each motor respectively, it is crucial to coincide the origin of the body frame on the body center of gravity to simplify the equations of motions and inertial moments. In order to ease the modeling work, within Newton Euler formalism, many assumptions have been taken:

- The structure is considered perfectly rigid with symmetrical properties to ease the diagonal inertial matrix calculation.
- Propellers are considered identical and rigid without flapping.
- Motors are almost similar to match close electric modeling and behaviors.
- Thrust and drag are maintained in perfect gas conditions with laminar stream velocity. In fact, some propellers are designed in ducted fans to optimize the thrust, It is known that thrust is proportional to the square of the spinning speed of the propeller.

2.2 Newton Euler formalism

Equations of motions are developed in the body frame to ensure ease of the study.

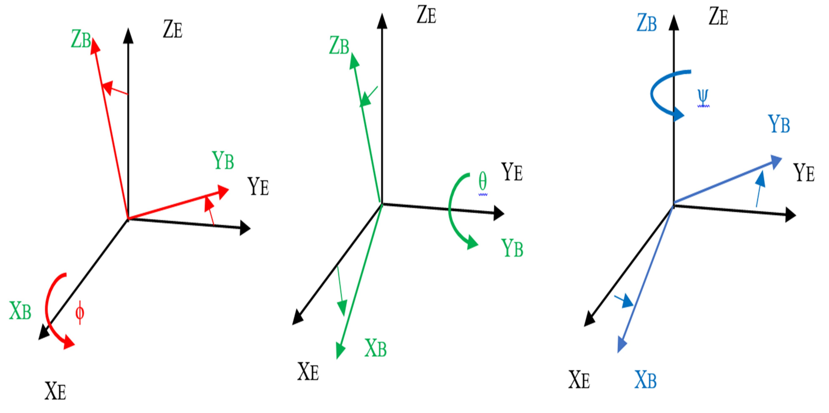


Figure 2-2: Euler's angles of rotations ($\theta, \phi, and \psi$ respectively)

2.2.1 Kinematics

For body kinematics, we don't take into consideration forces or torques applied on the quadrotor.

The linear and angular position in the E frame, and the linear and angular velocities in the B frame can be noted as:

$$\Sigma = [\Gamma^E, \Theta^E]^T = [X\ Y\ Z\ \phi\ \theta\ \psi]^T \tag{2.1}$$

$$\nu = [V^B, W^B]^T = [u\ v\ w\ p\ q\ r]^T \tag{2.2}$$

Using the rotation matrices of Euler between the two orthogonal references as in Fig.2-2:

$$R\Theta = R(\Psi, z)\ R(\theta, y)\ R(\phi, x) =$$
$$R\Theta = \begin{bmatrix} c\psi c\theta & c\psi c\phi + c\psi s\theta s\psi & s\psi s\phi + c\psi s\theta c\phi \\ s\psi c\phi & c\psi c\phi + s\psi s\theta s\phi & -c\psi s\theta + s\psi s\theta c\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \tag{2.3}$$

$R\Theta$ is a result of three consecutive rotations around linear independent axis. ψ, ϕ and θ represent the Yaw, Roll and Pitch angles respectively. Relation between linear velocities in B frame V^B and in E frame $\dot{\Gamma}^E$:

$$V^E = \dot{\Gamma}^E = R\Theta \; V^B \tag{2.4}$$

By the same analogy, we got the angular velocity in E frame $\dot{\Theta}^E$ by the following relation derived from Euler Rates:

$$\dot{\Theta}^E = T\Theta \; W^B \tag{2.5}$$

where

$$T\Theta = \begin{bmatrix} 1 & s\phi + t\theta & c\phi t\theta \\ 0 & c\theta & s\phi \\ 0 & s\phi/c\theta & c\theta c\phi \end{bmatrix} \tag{2.6}$$

$T\Theta$ is the inverse if $T\Theta^{-1}$ that is determined by resolving the Euler rates and satisfies:

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + R(\phi, x)^{-1} \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + R(\phi, x)^{-1} R(\theta, y)^{-1} \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} = T\Theta^{-1} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{2.7}$$

$T\Theta$ is usually linearized to the I_{3*3} matrix because of instantaneous small angles of attitude, So, derivation of generalized position in E frame could be written as :

$$\dot{\Sigma} = J\Theta \; \nu \tag{2.8}$$

$$J\Theta = \begin{bmatrix} R\Theta & 0_{3*3} \\ 0_{3*3} & T\Theta \end{bmatrix} \tag{2.9}$$

2.2.2 Dynamics

From a dynamics perspective, the effect of forces and torques on the motions should be studied in the B frame, the reason behind that is due to the invariance of the inertial matrix in time, as using the symmetry of the body to ease the equations, despite the explicit nature of applied forces in the B frame. The origin O_B of B-frame has been precisely chosen as the Center Of Mass (COM) to simplify the calculation of the inertial moment's matrix. In such a way, the quadrotor is symmetric with respect to the X_b, Y_b& Z_b axes. Then, the center of gravity coincides with the geometric center of the quadrotor. The dynamics equations of the quadrotor model without consideration of air drag can be derived as [52].

From Euler's First axiom of Newton's Second Law: deriving the linear velocity vector in the E-frame:

$$m\ \ddot{\Gamma}^E = F^E \tag{2.10}$$

$$m\ (\dot{V}^B + W^B * V^B) = F^B \tag{2.11}$$

With $W^B = R\Theta^{-1}\dot{R}\Theta$.

Similarly, deriving the angular component of motions of Euler axiom of Newton's second law

$$I\ \ddot{\Theta}^E = \tau^E \tag{2.12}$$

$$T\Theta^{-1} * I\ (T\Theta\ \dot{W}^B) = T\Theta^{-1}\tau^E \tag{2.13}$$

$$I\ \dot{W}^B + W^B * (I\ W^B) = \tau^B \tag{2.14}$$

From both Eqs. (2.11) & (2.14), we can write:

$$\begin{bmatrix} m\ I_{3*3} & 0_{3*3} \\ 0_{3*3} & I \end{bmatrix} \begin{bmatrix} \dot{V}^B \\ \dot{W}^B \end{bmatrix} + \begin{bmatrix} W^B * (m\ V^B) \\ W^B * (I\ W^B) \end{bmatrix} = \begin{bmatrix} F^B \\ \tau^B \end{bmatrix} \tag{2.15}$$

A generalized vector of forces and torques Λ could be represented by:

$$\Lambda = [F^B\ \tau^B]^T = [F_x\ F_y\ F_z\ \tau_x\ \tau_y\ \tau_z]^T \tag{2.16}$$

So , Eq. (2.15) can be written as:

$$M_B\ \dot{\nu} + C_B\ \nu = \Lambda \tag{2.17}$$

$$\dot{\nu} = -M_B^{-1}\ C_B\ \nu + M_B^{-1}\ \Lambda \tag{2.18}$$

Where M_B is diagonal, because of the considered assumptions.

$$M_B = \begin{bmatrix} m\ I_3{*}3 & 0_{3*3} \\ I_{3*3} & I \end{bmatrix} = \begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{xx} & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{yy} & 0 \\ 0 & 0 & 0 & 0 & 0 & I_{zz} \end{bmatrix} \tag{2.19}$$

C_B called the Coriolis centripetal matrix and its represented as follows:

$$C_B = \begin{bmatrix} 0 & 0 & 0 & 0 & m\ w & -m\ v \\ 0 & 0 & 0 & -m\ w & 0 & m\ u \\ 0 & 0 & 0 & m\ v & -m\ u & 0 \\ 0 & 0 & 0 & 0 & I_{zz}\ r & -I_{yy}\ q \\ 0 & 0 & 0 & -I_{zz}\ r & 0 & I_{xx}\ p \\ 0 & 0 & 0 & I_{yy}\ q & -I_{xx}\ p & 0 \end{bmatrix} \tag{2.20}$$

Eq.(2.18) is generic and it is valid for any rigid body motion with the considered assumptions that have been made previously. The dynamics of the quadrotor are under three contributions, Gravitational forces in the B-frame:

$$G_B = R\Theta^{-1}\ F_E = R\Theta^{-1}\ G_E \tag{2.21}$$

Using the orthogonal normalized property

$$R\Theta^{-1} * R\Theta^{T} = I \tag{2.22}$$

$$G_B = \begin{bmatrix} m\ g\ s\theta \\ -m\ g\ c\theta\ s\phi \\ -m\ g\ c\theta\ s\phi \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.23}$$

The second effect is due to the rotations of propellers that create thrusts along all the motors' axis, the thrust of each motor is proportional to the square of propeller speed. From the understanding of the nature of motions of quadcopter, as previously explained, we can define the vector U_B:

$$U_B = \begin{bmatrix} 0 \\ 0 \\ U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ b\ \sum_{i=1}^{4} \Omega_i^2 \\ b\ l\ (\Omega_4^2 - \Omega_2^2) \\ b\ l\ (\Omega_3^2 - \Omega_1^2) \\ d\ (\Omega_4^2 + \Omega_2^2 - \Omega_3^2 - \Omega_1^2) \end{bmatrix} \tag{2.24}$$

Where we define U_1 as the resultant thrust generated by all the motors and it is oriented within Z_b axes. U_2 is the moment of torque generated around the X_b axes involving the motor number (4) & (2). U_3 is the moment of torque generated around the Y_b axes involving the motor number (3) & (1). U_4 is the moment of torque generated around the Z_b axes involving all motors.

The third effect to be considered is the Gyroscopic effect, which happens because the velocity unbalances of the four propellers relative to body motions:

$$N_B = J_{TP} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ q & -q & q & -q \\ -p & p & -p & p \\ 0 & 0 & 0 & 0 \end{bmatrix} \Omega \tag{2.25}$$

Where $\Omega = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4$. By applying the forces and torques model to the generic equation of motion, Eq.(2.18), we find:

$$\dot{\nu} = M_B^{-1} \left[-C_B\, \nu + U_B + N_B + G_B \right] = \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} \tag{2.26}$$

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} 1/m & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/m & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/m & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/I_{xx} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/I_{yy} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/I_{zz} \end{bmatrix} \Bigg[\begin{bmatrix} 0 & 0 & 0 & 0 & -mv & mv \\ 0 & 0 & 0 & -mw & 0 & -mu \\ 0 & 0 & 0 & -mv & mu & 0 \\ 0 & 0 & 0 & 0 & -I_{zz}r & I_{yy}q \\ 0 & 0 & 0 & I_{zz}r & 0 & -I_{xx}q \\ 0 & 0 & 0 & -I_{yy}q & I_{xx}p & 0 \end{bmatrix}$$

$$\begin{bmatrix} u \\ v \\ w \\ p \\ q \\ r \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} + J_{TP} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ q & -q & q & -q \\ -p & p & -p & p \\ 0 & 0 & 0 & 0 \end{bmatrix} \Omega + \begin{bmatrix} m\ g\ s\theta \\ -m\ g\ c\theta\ s\phi \\ -m\ g\ c\theta\ s\phi \\ 0 \\ 0 \\ 0 \end{bmatrix} \Bigg] \tag{2.27}$$

By development of last expression, we find:

$$\begin{cases} \dot{u} = (vr - wq) + gs\theta \\ \dot{v} = (wp - ur) - gc\theta s\phi \\ \dot{w} = (uq - vp) - gc\theta s\phi + U_1/m \\ \dot{p} = 1/I_{xx}\ ((I_{yy} - I_{zz})qr + J_{TP}q\Omega + U_2) \\ \dot{q} = 1/I_{yy}\ ((I_{zz} - I_{xx})pr + J_{TP}p\Omega + U_3) \\ \dot{r} = 1/I_{zz}\ ((I_{xx} - I_{yy})pq + U_4) \end{cases} \tag{2.28}$$

The dynamics of the quadrotor are represented by the system of differential equations in (2.28)which is referenced in the B-frame, however considering the hovering mode, where minimal angle variation in B-frame, a hybrid frame reference could be defined as follows:

$$\Sigma = [\Gamma^E, W^B]^T = [X\ Y\ Z\ p\ q\ r]^T \tag{2.29}$$

Dynamics of the system in the new hybrid reference will be written as:

$$\begin{cases} \ddot{X} = (s\psi s\phi + c\psi s\theta c\phi)\ U_1/m \\ \ddot{Y} = (-c\psi s\phi + s\psi s\theta c\phi)\ U_1/m \\ \ddot{Z} = -g + c\theta c\phi\ U_1/m \\ \ddot{\phi} = 1/I_{xx}\ ((I_{yy} - I_{zz})\ \dot{\phi}\dot{\psi} + J_{TP}\dot{\phi}\Omega + U_2) \\ \ddot{\theta} = 1/I_{yy}\ ((I_{zz} - I_{xx})\dot{\theta}\dot{\psi} + J_{TP}\dot{\theta}\Omega + U_3) \\ \ddot{\psi} = 1/I_{zz}\ ((I_{xx} - I_{yy})\dot{\theta}\dot{\phi} + U_4) \end{cases} \tag{2.30}$$

2.3 State representation

The dynamic of quadrotor might be writing in state space representation:

$$\dot{X} = F(X, U) \tag{2.31}$$

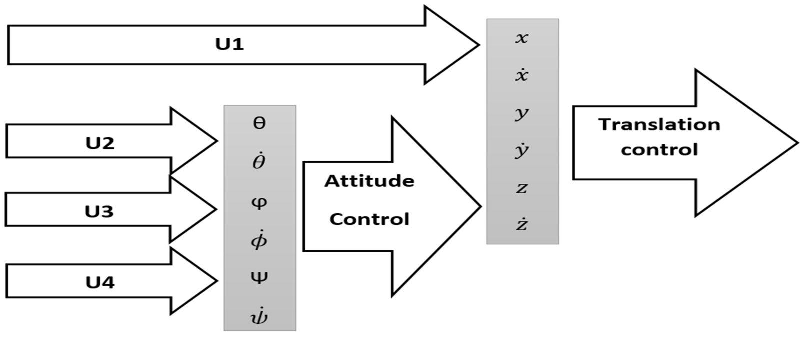


Figure 2-3: states correlation

By introducing $X = (x_1\ x_2\ ...\ x_{12})^T$ in R^{12} space of state vector of the system:

$$\begin{bmatrix} x_1 = x \\ x_2 = \dot{x}1 \\ x_3 = y \\ x_4 = \dot{x}_3 \\ x_5 = z \\ x_6 = \dot{x}_5 \\ x_7 = \phi \\ x_8 = \dot{x}_7 \\ x_9 = \theta \\ x_{10} = \dot{x}_9 \\ x_{11} = \psi \\ x_{12} = \dot{x}_{11} \end{bmatrix} F(X,U) = \begin{pmatrix} x_2 \\ (\cos x_7 \sin x_9 \cos x_{11} + \sin x_7 \sin x_{11})\ U_1/m \\ x_4 \\ (\cos x_7 \sin x_9 \sin x_{11} - \sin x_7 \cos x_{11})\ U_1/m \\ x_6 \\ -g + (\cos x_7 \cos x_9)\ U_1/m \\ x_8 \\ 1/I_{xx}\ ((I_{yy} - I_{zz})\ x_{12}x_{10} + J_{TP}x_{10}\Omega + U_2) \\ x_{10} \\ 1/I_{yy}\ ((I_{zz} - I_{xx})x_{12}x_8 + J_{TP}x_8\Omega + U_3) \\ x_{12} \\ 1/I_{zz}\ ((I_{xx} - I_{yy})x_{10}x_8 + U_4) \end{pmatrix} \tag{2.32}$$

The dynamics representation in the state-space gives a special characteristic to ease the quadrotor control. Two loops can be formed as shown in Fig.2-4, where one is completely independent of the other one, Cartesian position vector depends on attitude states, however, the attitude of the quadrotor is independent of the translations. Nested looping by outer/inner loops will facilitate adaptive control [102][11].

This philosophy of subsystems dependency is generally famous in under actuated systems and widely used to manipulate states in an easier format, moreover, many applications use the philosophy of nested loops when subsystems are ideally isolated [1], for closed loops

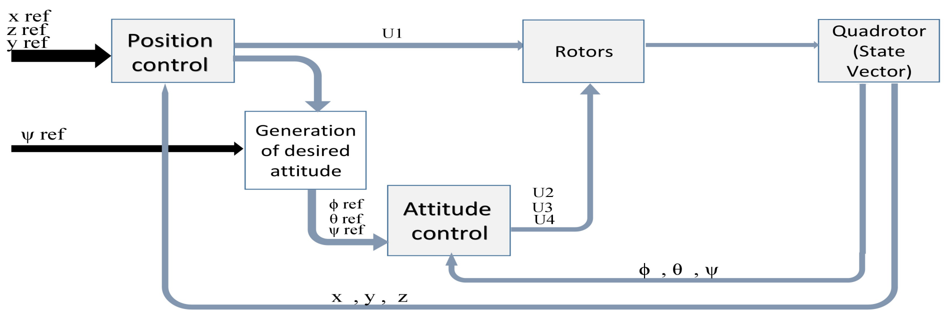


Figure 2-4: The nested control subsystems diagram. *The inner loop for attitude and the translation position outer control loop are shown.*

control based on processing units, we can run those nested loops at different calculation rates. For this reason, a well-known strategy of feedback control is based on this design.

2.4 Rotor Dynamics

The quadrotor is outfitted with four fixed pitch propellers, brushless motors are commonly used for their easiness of control using Electronic Speed Controller (ESC), and for their quick response, ESCs are controlled via Pulse Width Modulation (PWM) at 250 hz. The ESC is operated at a frequency of 250 hertz to enable precise control. The dynamics of the rotor speed can be satisfactorily represented using a first-order model.

2.5 Conclusion

This chapter presented dynamics modeling using Newton-Euler formalism. Euler-Lagrange optimization would result in the same representation being derived. The nominal model would be used in simulation and in the Kalman filter linearization. Thrust forces were considered proportional to the square of the propellers' speed, and Hub horizontal forces acting on all the blade elements were neglected and considered only as a bounded perturbation, furthermore, the mass of the miniature quadrotor, and the size of its propellers limit the Hub forces effect. Drag moments and translational drag are both found so small compared to other forces and moments' contributions. The attributed model in Eq.2.32 is a faithful model, moreover, it has been considered and validated by numerous papers.[1]

[1]GitHub codes - The Simulink Model.

Chapter 3

Classic Control Techniques

Concept of Simulation

This chapter focuses on simulating several classic control methods from different classes of control theory on a quadrotor model in Simulink. The goal is to achieve trajectory tracking while accounting for noise and disturbances. Different control architectures were analyzed, including both nestled loops and decentralized strategies. The primary focus was on evaluating tracking performance, robustness, response time, and attitude stabilization.

3.1 Design of the challenge

A 3D-developed quadrotor model on PTC Creo Parametric CAD software was used for the real-time visualization as shown in Fig.3-1. The known validated mathematical model was exploited. Disturbance and noise were added as Gaussian white noises to the model. Therefore, a low feeding rate Gaussian signal was added as a perturbation to reproduce gust wind effects on Cartesian positions, and from the other side, to emulate the noise of sensors readings and to simulate the un-modeled dynamics, a very high feeding rate Gaussian noise was added.

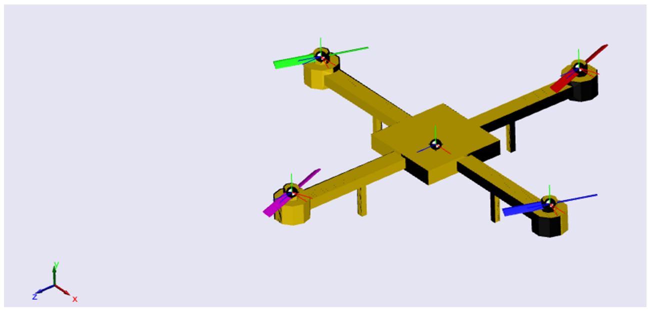


Figure 3-1: 3D Simulation of quadrotor in Matlab simmechanics environment

The 3D visualization model uses a multi-bodies configuration developed in Matlab Simmecanics, with all parts attached in the same environment with 6 DOF, the drawings are imported from Creo Parametric to constitute the complete quadrotor. The 3D visualization is controlled online by the control part of Simulink which executes a mathematical simulation of each control theory. A synthesis of the Simmechanics model is shown in Fig. (3-2).[1]

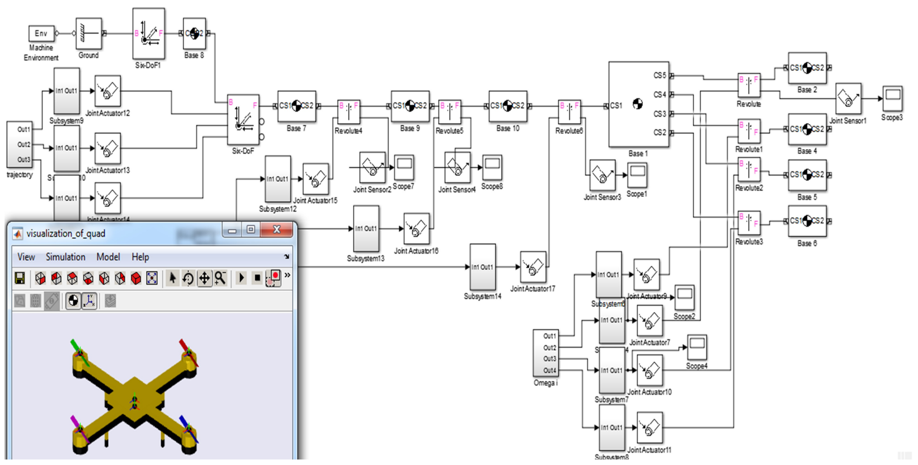


Figure 3-2: Environment of Simmechanic for 3D visualisation

To better understand the performance of each control method, a path-tracking challenge has been raised. Instead of a simple simulation on hovering mode control, Lissajous curves which describe s complex harmonic motion were chosen as a base trajectory to be followed with variable altitude, those quasi-periodic orbital trajectory curves have been used previously in several missions by many researchers. The aim was to propose a trajectory that can be used for surveillance by a single or several UAVs.

[1]GitHub codes-3D Model visualization.

Desired trajectory generation in Matlab

$t = 0 : 0.2 : 30 * pi;$

$A = 5; a = 5; B = 10; b = 4; C = 8; c = 5; coef = 0.07;$

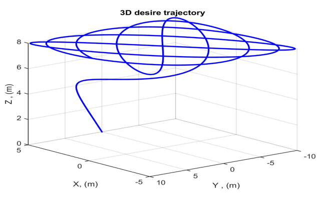
$x = A * cos(a * coef * t);$

$y = B * sin(b * coef * t);$

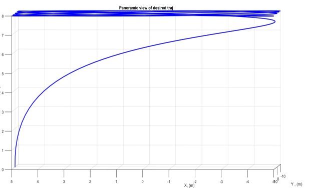
$z = C * (1 - exp(-5 * coef * t));$

$plot3(x, y, z, 'b')$

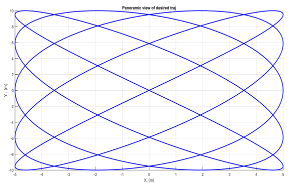
$grid\ on$



(a) 3D desired trajectory



(b) Altitude variation



(c) Trajectory top view

Figure 3-3: Desired trajectory to be tracked by quadrotor for surveillance aim

Fig.3-3 illustrates the 3D desired trajectory to be followed in time reference[2], the horizontal motion of the quadrotor was designed to cover a large surface of the surveillance mission, and the altitude was planned to follow an exponential convergence to the desired height.

Another depth of challenge to the trajectory tracking was given by assigning different initial positions for the quadrotor and the desired trajectory at time zero. Generally,the trajectory tracking is more complicated than the path following challenge, as the tracking is a time-referenced problem.

The system developed in the modeling section in Eq.(2.32) represents valid dynamics of the quadrotor UAV system. Un-modeled dynamics will be covered by generated noise and disturbance during the simulation process. The motors model will be considered as well as a fast-reacting first-order system based on the fact of brushless motors' quick response, brushless motors are widely used recently in projects of realization and experimentation.

Effectively, the nested loops architecture was found to be one of the best design strategies to execute control approaches. Translation positions are depending on angles that force us

[2]GitHub codes-Lissajou curves.

to put them in outer control, rotation angles, and their derivatives control known as attitude are free from translation positions and this subsystem can be controlled in the inner loop, refer to Fig.3-4.

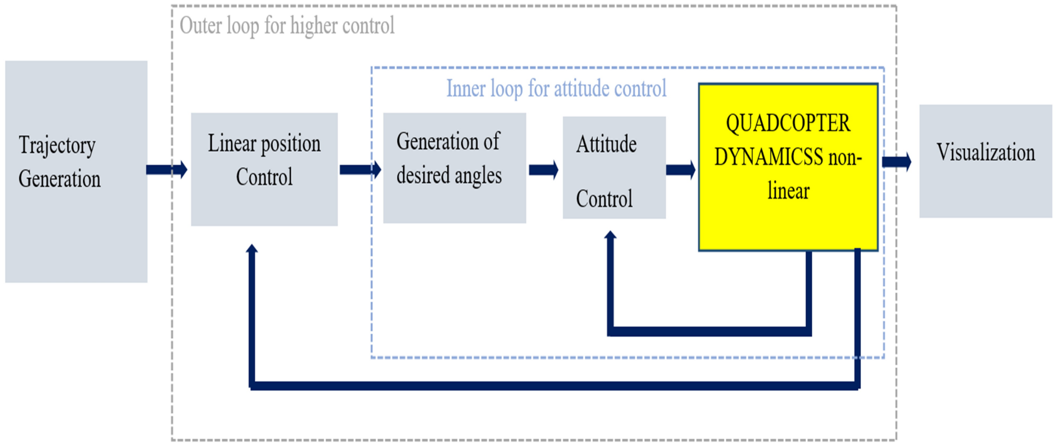

Outer loop for higher control
Inner loop for attitude control
Trajectory Generation
Linear position Control
Generation of desired angles
Attitude Control
QUADCOPTER DYNAMICSS non-linear
Visualization

Figure 3-4: Control loops for quadrotor

3.1.1 Calculation of desired roll and pitch angles

Generation of desired angles ϕ_d & θ_d during the simulation was done by one of three equivalent methods:

First Method

The first method is based on the calculation of those desired angles ϕ_d & θ_d from the desired position while considering instantaneous small ψ angle, the angles are resulting from simple resolution of the system of equations mentioned in Eq.(3.1). However, this small function generates desired angles with more discontinuities than tolerated for the needed derivative contributions, this latter can affect the calculation of angles derivatives even with the use of firm value saturation, which adds more non-linearities to the simulated system.

$$\begin{cases} u_x = \cos\phi \sin\theta \cos\psi + \sin\phi sin\psi \\ u_y = \cos\phi \sin\theta \sin\psi - \sin\phi \cos\psi \end{cases} \tag{3.1}$$

A simple first solution can be written as:

$$\begin{cases} \psi << 1 \\ \phi = \arcsin(-u_y) \\ \theta = \arcsin\left(u_x / \cos(\phi)\right) \end{cases} \tag{3.2}$$

Second Method

A second solution which is widely used came by solving the equations analytically:

$$\begin{cases} \phi_d(t) = \arcsin\Big(\Big(\sin\psi(t) u_x - \cos\psi_d(t))\Big) \Big) / (u_x^2(t) + u_y^2(t))\Big) \\ \theta_d(t) = \arcsin\Big(\Big(u_x(t) - \sin\psi_d(t)\sin\phi_d(t)\Big) \Big) / (\cos(\psi_d(t))\cos(\phi_d(t))\Big) \end{cases} \tag{3.3}$$

Nonetheless, this method is related just to the dynamics considered and limited toward more added consideration of unmodeled dynamics.

Third Method

A third smoother method that comes with a more expensive calculation cost is based on direct nonlinear solving of the angles system, smoother profiles are more likely to be generated using this method based on results obtained from simulation.

The benefits of this third method can be hidden behind the simplification and assumptions made while modeling, in complicated cases where we consider more accurate modeling with no neglect of minimal nonlinear contribution, this method is more reliable and precise. A faster run of this non-linear solution can be resulted in limiting the maximum number of resolution iterations or minimizing the tolerated error. Matlab function $fsolve$ has been used in Simulink as interpreted function since $fsolve$ is not among the functions supported for code-generation in Embedded MATLAB Function blocks.

Another depth of performance of this method comes when we deal with other types of multirotor and other copters designs.

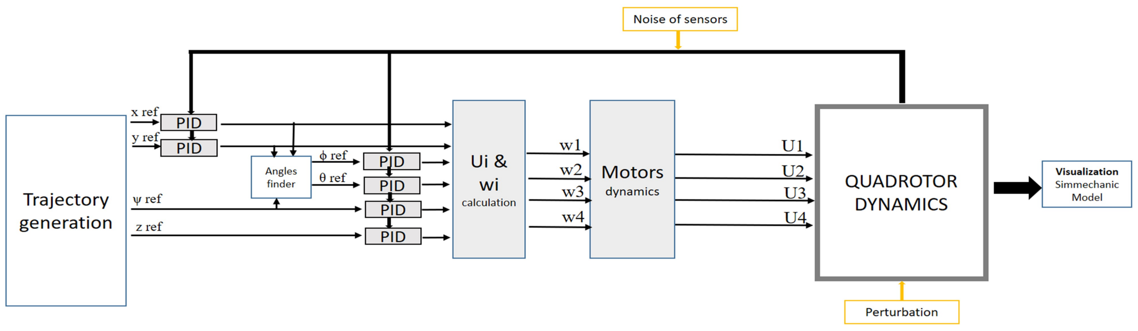


Figure 3-5: PID control of quadrotor design

3.2 Quadrotor PID Control

A design of multiple PID controllers was approached for each position's state. A tuning of a decentralized architecture of six PIDs' parameters was performed in the Matlab-Simulink environment.

3.2.1 Synthesis of the PID

The objective of this PIDs strategy is to stabilize the attitude based on inner looping that controls angles (orientations) and their derivatives and then to minimize linear position errors as second-order stable dynamics that are applied in the outer loop to ensure path following. From a trivial perspective, the PID correction made for attitude should be much quicker to the outer loops PID corrections attributed to translation positions As shown in Fig.(3-5), the simulation used several blocks, PIDs are used for each linear and rotational position, a manual tuning was proceeded to get desired responses, parameters for each PID can be done separately. To track a trajectory, a tracking error vector e of six components e_i is formulated with the objective to converge them exponentially to zero. each of these tracking errors is assigned for one of the six-position states. For each control, we want the state xi to follow the desired x_{id}, we define:

$$e_i = x_{id} - x_i \tag{3.4}$$

Derivation of this error e_i gives:

$$\dot{e}_i = \dot{x}_{id} - \dot{x} \tag{3.5}$$

In order to formulate an exponential stable error that converges to zero, we may consider this stable dynamic as follows:

$$\ddot{x}_{id} - \ddot{x}_i + K_d\dot{e}_i + K_p e_i + K_i \int_0^t e_i(\tau)d\tau = 0 \tag{3.6}$$

so

$$\ddot{x}_i = \ddot{x}_{id} + K_d\dot{e}_i + K_p e_i + K_i \int_0^t e_i(\tau)d\tau \tag{3.7}$$

This methodology will be followed to generate all control inputs for each PID loop with $K_p > 0$, $K_d > 0$, and $K_i > 0$.

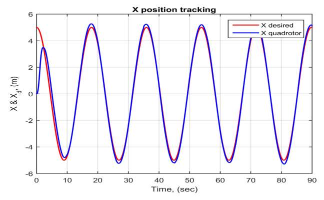
A simple Lyapunov candidate function such as $V = \frac{1}{2}X^T X$ can prove the stability of the PID control if we satisfy the error exponential convergence.

3.2.2 Results

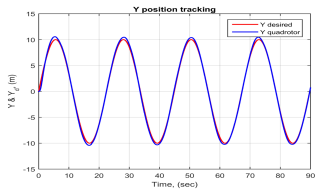
PID control has been applied for all linear and angular position states. The simulation demonstrated satisfying tracking dynamics as shown in Fig.3-6, the quadrotor trajectory in red, following the blue desired reference while starting initially from different spots. Fig.3-6 shows the result of the simulation using the validated quadrotor model with PID control strategy to follow the Lissajous desired trajectory with an acceptable error considering the different initial positions.

Linear position tracking showed minimal error, wind gust simulation effect via the low-rate Gaussian noise is clear on the quadrotor position, with a quick correction to track. The angles' tracking was also very efficient, observing the noise introduced on the sensor by the high rate of white Gaussian noise. Obtained results of the simulation were satisfactory regarding the simple synthesis approach of PID control. [3]

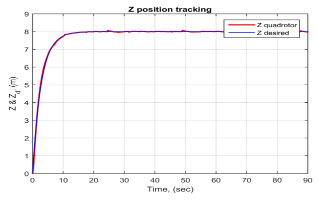
[3]GitHub codes-PID control.



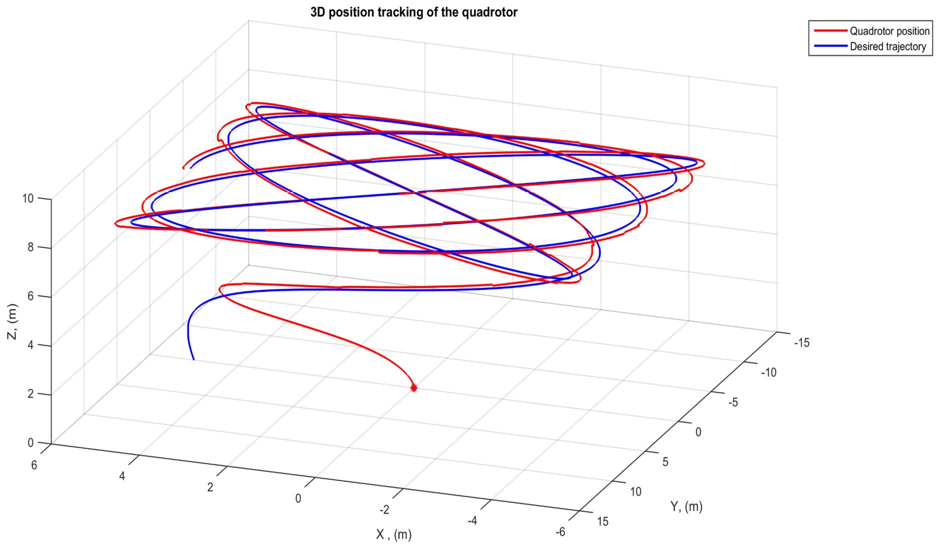
(a) X tracking



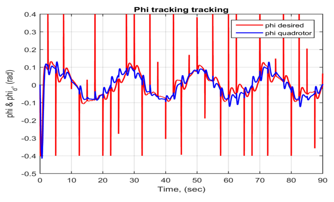
(b) Y tracking



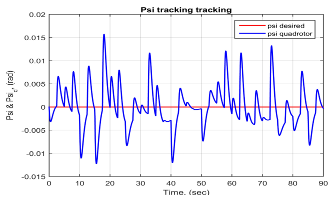
(c) Altitude tracking



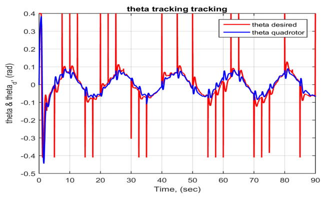
(d) 3D tracking



(e) Roll tracking



(f) Yaw tracking



(g) Pitch tracking

Figure 3-6: PID trajectory tracking

3.2.3 Practical depth of PID implementation

Implementation of such a method on microcontrollers is the easiest compared to other approaches, criteria of running speed of the control loop and input saturation should be considered, this latter justifies the use of saturation over the control "U_i" used for simulation. A control loop frequency at less than 50 Hz, may induce enough delay to destabilize the system and may create too much oscillation that deteriorates even IMU measurement quality.

Although the high rate frequency of IMU readings, an appropriate loop of execution should be wisely chosen to not waste data from motion measurement.

An integral action should be considered to kill the steady state error, nevertheless, a reset of integral contribution should be integrated with the process to not diverge the summation of errors. The fast response actuation of the ESC over the brushless motors is a key factor of stabilization, a delay from the execution loop or from the ESC response will tremendously break down the attitude performance. In case of implementation, accelerometer data should be used to stabilize the quadrotor horizontally and define the zero level (Auto-leveling), due to the electronic drift of the Gyro.

3.3 Quadrotor control by Integrator Backstepping

IBS is a nonlinear approach that contributes the integral benefits in the backstepping design, this approach can ensure asymptotic stability with attenuation of steady-state errors with the help of the integral effect. Its robustness was simulated as we added to the model an external disturbance and internal noise. The Control design is based on four IBS controllers. Four controllers are resulting from the underactuated nature of the quadrotor, which is based on just four controllers' capability. The derivation of those controllers is similar for both attitude angles & altitude. The linear positions controllers for X and Y will be done systematically as we ensure validated tracking of desired ϕ_d and θ_d angles. Thus, only one control for attitude will be derived in this thesis. Once we calculate U_i, we apply them to the next block to calculate the voltages and control rotation of each motor.

3.3.1 Attitude Control

Essentially, Quadrotor control is founded on Attitude control, so, care will be focused on the stabilization of angles at the first level and then on the general position correction at a deeper stage, the attitude will result in the regulation of the inner loop, which adopts the 3D orientation of the quad to the desired values.

The first step in the IBS control strategy is to define the tracking error for each rotation. Let's consider the error for ϕ angle around X_b axes:

$$e_1 = \phi_d - \phi \tag{3.8}$$

By the derivation of error e_1:

$$\dot{e}_1 = \dot{\phi}_d - p \tag{3.9}$$

p is the angular rate around X_b as mentioned in Fig.2-1. We need to control the dynamics of p to converge to $\dot{\phi}_d$. For that, we initiate the desired tracking dynamics by

$$p_d = c_1 e_1 + \dot{\phi}_d + \alpha_1 \int_0^t e_1(\tau) d\tau \tag{3.10}$$

with c_1 & $\alpha_1 > 0$.

Hence, $\alpha_1 \int_0^t e_1(\tau) d_\tau$ represents the integral contribution to cancel the steady error.

let's consider:

$$e_2 = p_d - p \tag{3.11}$$

By derivation of Eq.(3.11) & using of Eqs.(3.9)-(3.10)- and ϕ dynamics in Eq.(2.32), we can get:

$$\dot{e}_2 = c_1(\dot{\phi}_d - p) + \ddot{\phi}_d + \alpha_1 e_1 - \dot{\theta}\dot{\psi} a_1 - \dot{\phi} a_2 \Omega_r - b_1 U_2 \tag{3.12}$$

Where an appearance of a control input U_2 is shown in this latter. By assuming a stable dynamic for e_2:

$$\dot{e}_2 + c_2 e_2 + e_1 = 0 \tag{3.13}$$

with $c_2 > 0$

A control input that satisfies this assumption can be calculated as follows:

$$U_2 = 1/I_{xx}\left((1-c_1^2+\alpha_1)e_1+(c_1+c_2)e_2-c_1\alpha_1\int_0^t e_1(\tau)d\tau+\ddot{\phi}_d-\dot{\phi}\dot{\psi}(I_{yy}-I_{zz})/I_{xx}-\dot{\theta}\Omega_r J_{TP}/I_{xx}\right) \tag{3.14}$$

Systematically. The other control inputs derived from θ and ψ errors are given by:

$$U_3 = 1/I_{yy}\left((1-c_3^2+\alpha_2)e_3+(c_3+c_4)e_4-c_3\alpha_2\int_0^t e_3(\tau)d\tau+\ddot{\theta}_d-\dot{\phi}\dot{\psi}(I_{zz}-I_{xx})/I_{yy}-\dot{\phi}J_{TP}/I_{yy}\Omega_r\right) \tag{3.15}$$

$$U_4 = 1/I_{zz}\left((1-c_5^2+\alpha_3)e_5+(c_5+c_6)e_6-c_5\alpha_3\int_0^t e_5(\tau)d\tau\right) \tag{3.16}$$

With $(c_3, c_4, c_5, c_6, \alpha_2, \alpha_3) > 0$

3.3.2 Altitude and planar positions Control

Similarly, to the derivation technique applied for ϕ angle, an altitude tracking error is defined as:

$$e_7 = z_d - z \tag{3.17}$$

$$e_8 = c_7 e_7 + \dot{z}_d + \alpha_4 \int_0^t e_4(\tau)d\tau \tag{3.18}$$

Its errors dynamics can be tracked as>

$$e_8 = c_7 e_7 + \dot{z}_d + \alpha_4 \int_0^t e_4(\tau)d_\tau - \dot{z} \tag{3.19}$$

By derivation of this error, a control input is appearing, assuming a stable first-order dynamics for this error, we can find:

$$U_1 = (m/\cos\phi\cos\theta)\Big(g + (1-c_7^2+\alpha_4)e_7 + (c_7+c_8) - c_7\alpha_4\int_0^t e_4(\tau)d_\tau\Big) \tag{3.20}$$

For both X & Y control, simple stable second-order tracking dynamics were defined by a PID, then desired angles were generated by direct calculation continuously to energize the IBS block in order to calculate the U_i control inputs using Eqs.(3.14) to (3.16) & (3.20).

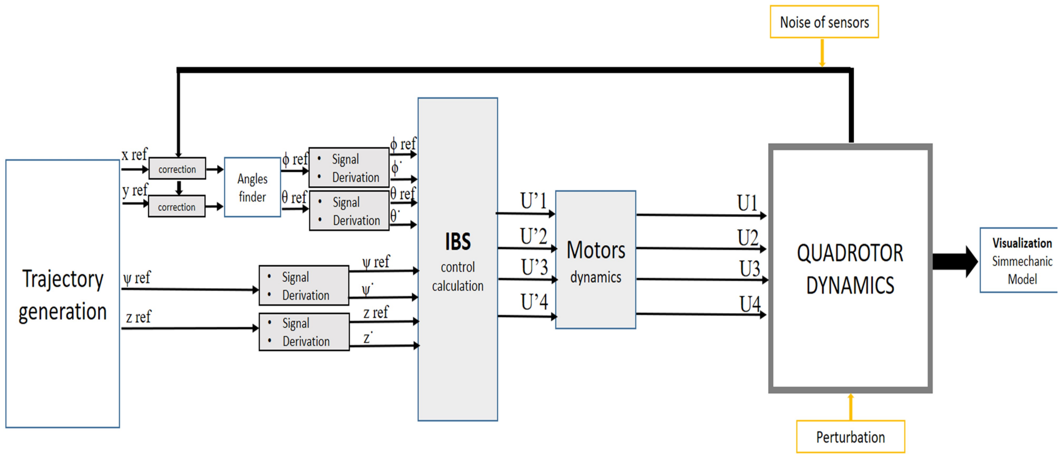
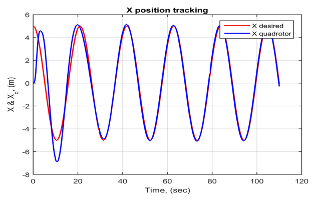


Figure 3-7: IBS control design of quadrotor

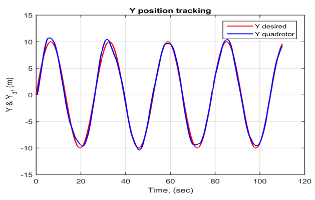
3.3.3 Integrator backstepping algorithm

During the simulation process, we defined twelve coefficients of IBS to the eight errors' dynamics and their integral actions $(\phi, \theta, \psi\ \&\ z\ )$. Errors should be calculated first at each step of the simulation based on desired states and the measured outputs and their derivatives. Then Ui control inputs should be calculated to excite the other blocks prior quadrotor nonlinear model block.

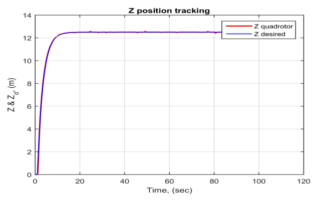
An algorithm that can summarize the IBS method is illustrated as:



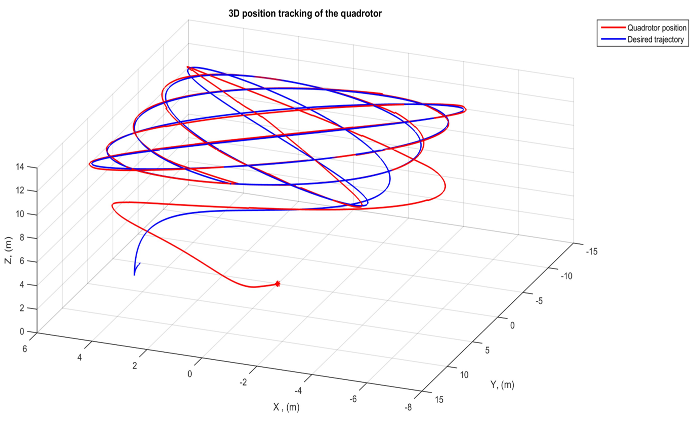
(a) X tracking



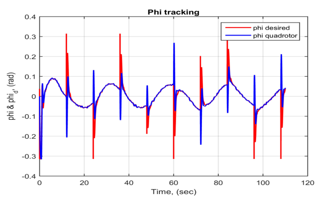
(b) Y tracking



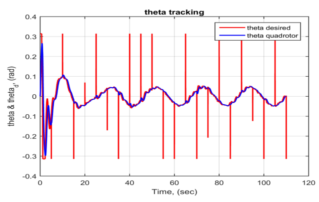
(c) Altitude tracking



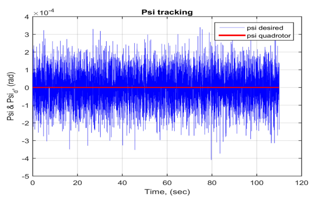
(d) 3D tracking



(e) Roll tracking



(f) Pitch tracking



(g) Yaw tracking

Figure 3-8: IBS trajectory tracking

```
Algorithm 1: Integrator Backstepping Algorithm
  Result: Apply the control $U_i$ on actuators
  Require coefficients of correction ;
  Read trajectory states;
  while $T \leq Looptime$ do
      Read /call for the sensors reading / state estimations;
      Calculate errors, their derivatives & their summations;
      Calculate $U_i$ by Eqs.(3.14) to (3.16) & Eq.(3.20). ;
      Calculate each rotor input;
      Generate the right control of each motor;
      Apply the control on actuators ;
      Wait for loop time to finish and execute the next loop;
  end
```

In case of adaptive parameterization, update correction parameters block and execute loop from step 1.

Linear position tracking showed minimal tracking error, the wind gust effect by low-rate Gaussian noise is observable on the quadrotor position, with the quick correction to track. angles tracking was also excellent and better than PID attitude results, all observing the noise effect introduced on the sensor by the high-frequency Gaussian noise. IBS control method simulation link[4].

3.4 Optimal Control using Hamiltonian optimization

The optimal Control is designed based on the validated mathematical model of the quadrotor previously derived, The simulated model is a continuous model, state vector values are either measured or obtained via a Kalman filter. The optimization and the linearized model of control as mentioned in Eq.2.32 around the hovering equilibrium state can be written as follow:

$$\begin{cases} Optimize \quad J = \frac{1}{2}\int_{t_0}^{t_f} X^T Q X dt + \frac{1}{2}\int_{t_0}^{t_f} U^T R U dt + \frac{1}{2} X_f^T S_f X_f \\ Subject\ to \quad \dot{X} = AX + BU \end{cases} \tag{3.21}$$

[4]GitHub codes.

The Pair $\{A, B\}$ is controllable, The system dynamics is known as the path constraint, J is the performance index or the cost function to be optimized, $Q \geq 0$, $R > 0$, and $S_f \geq 0$ are positive definite pondering matrices[14]. By taking $\Phi_f = \frac{1}{2} X_f^T S_f X_f$, and using the Hamiltonian on last performance index J, An optimal Control U can be designed as follows:

$$J_{ext} = \frac{1}{2} \int_{t_0}^{t_f} (X^T Q X + U^T R U + \frac{d}{dt}\Phi_f + \lambda^T (AX + BU - \dot{X}))dt \tag{3.22}$$

$$H = X^T Q X + U^T R U + \lambda^T (AX + BU) \tag{3.23}$$

- The state equation:
$$\dot{X} = \frac{\partial H}{\partial \lambda} = AX + BU \tag{3.24}$$

- The Co-state equation:
$$\dot{\lambda} = -\frac{\partial H}{\partial X} = -(QX + A^T \lambda) \tag{3.25}$$

- The Optimality:
$$\frac{\partial H}{\partial U} = 0 \quad => \quad U = -R^{-1} B^T \lambda \tag{3.26}$$

- Boundary condition:
$$\lambda_f = \frac{\partial \Phi}{\partial X_f} = S_f X_f \tag{3.27}$$

By taking $\lambda = PX$ in Eq 3.25, we can develop as follows:

$$\dot{P} + PA + A^T P - PBR^{-1} B^T P + Q = 0 \tag{3.28}$$

Which is the Riccati equation, with $P_f = S_f$ (*from the boundary condition*), the Control U can be defined by:

$$U = -R^{-1} B^T P X = -KX \tag{3.29}$$

A and B are the trivial linearized nonlinear model around the equilibrium. Applying the Kalman's theory for infinite time regulator (*Permanant regime*), $\dot{P} = 0$, Eq.3.28 is equivalent to an Algebraic Riccati equation (ARE) which is efficient numerically:

$$PA + A^T P - PBR^{-1} B^T P + Q = 0 \tag{3.30}$$

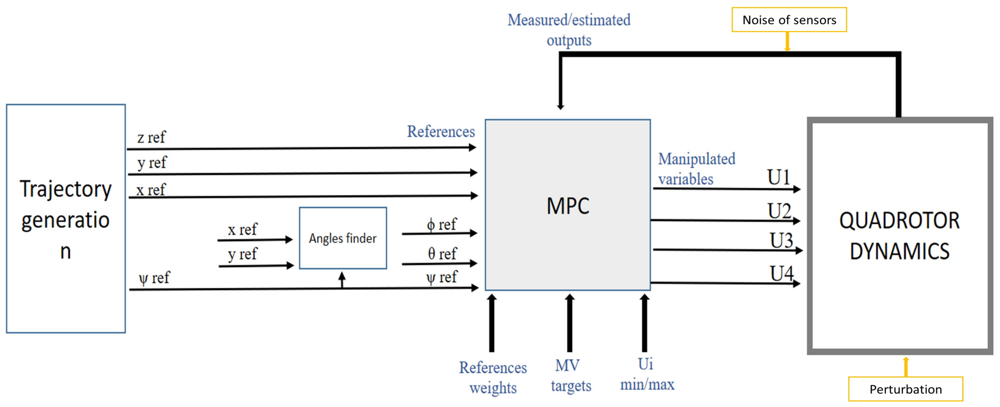


Figure 3-10: Design of optimal control architecture of Quadrotor

3.4.1 Optimal Control design

The Optimal control design shown in Fig.3-10 is structured as follows:

- Calculate A, B.
- Define the performance index Matrices Q, R
- Solve the Riccati equation such as in Eq.3.30.
- Apply the feedback control using Eq.3.29

3.4.2 Stability of closed loop using Optimal control

Choosing a Lyapunov positive definite candidate function $V = X^T PX$.

$$\dot{V} = \dot{X}^T PX + X^T P\dot{X}$$

Considering the feedback control

$$\dot{V} = ((A - BK)X)^T PX + X^T P((A - BK)X)$$

$$\dot{V} = X^T((A - BR^{-1}B^TP)^TP + P(A - BR^{-1}B^TP))X$$

$$\dot{V} = X^T((PA + A^TP - PBR^{-1}B^TP + Q - Q - P^TBR^{-T}BP)X$$

$$\dot{V} = X^T(-Q - P^TBR^{-T}BP)X \tag{3.31}$$

We have, $R > 0$, $R^{-1} > 0$, and $P > 0$), Hence,

$$-Q - P^TBR^{-T}BP < 0 \tag{3.32}$$

$\dot{V} < 0$, The optimal feedback control is asymptotically stable

3.4.3 Optimal tracking

The optimization process for tracking is based on minimizing the performance index J subject to system dynamics and states constraints.

$$J = (X_{Ref} - X)^TQ(X_{Ref} - X) + U^TRU \tag{3.33}$$

Where $X_{Ref}(t)$ is the reference vector in time.

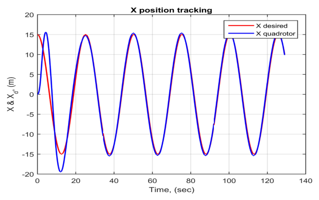
An adaptive sort of the optimal tracking can consider using Variable A, and B, such as A_i, and B_i are the linearization of of the model in Eq.2.32 around the estimated state.

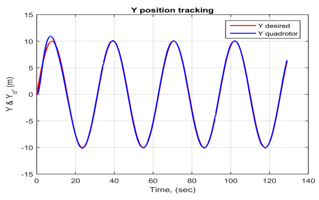
3.4.4 Results

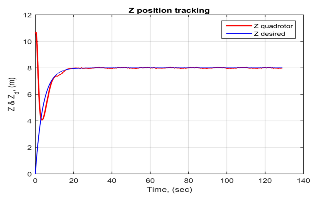
Despite the complexity and expensive computational time, the optimal controller remains an effective approach for autonomous quadrotor tracking control. Regardless of the noise and disturbance, this control approach handled excellent tracking performance as shown in Fig.3-11.

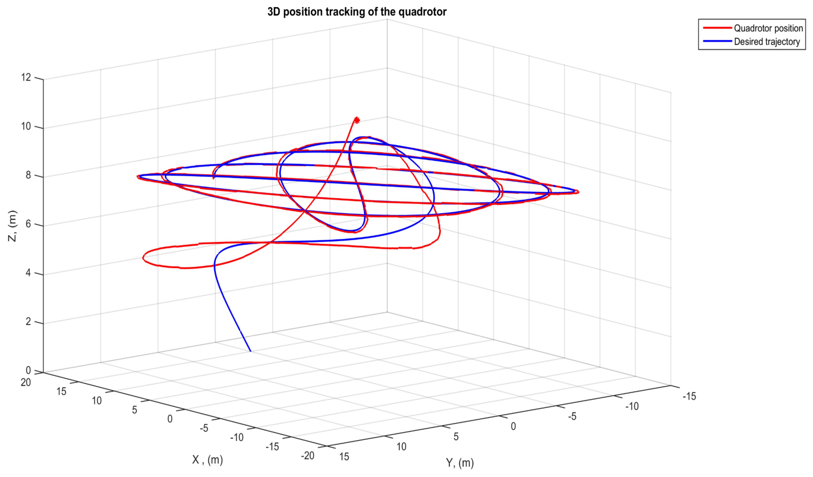
Linear position tracking was excellent with quick recovery from initial error, the wind gust effect by low-rate white Gaussian noise is barely observable but with quick correction. angles tracking was also excellent and better than PID and IBS attitude results, the high-frequency noise effect introduced on the sensor is observable. MPC method simulation file can be found on link[5] Although the shown stability by the optimal control, the performance

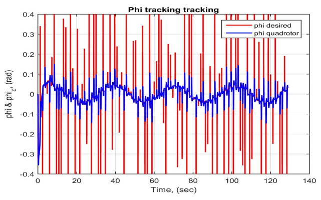
[5]GitHub codes-Matlab MPC.

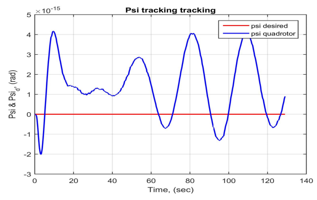

(a) X tracking


(b) Y tracking


(c) Altitude tracking


(d) 3D tracking


(e) Roll tracking


(f) Yaw tracking

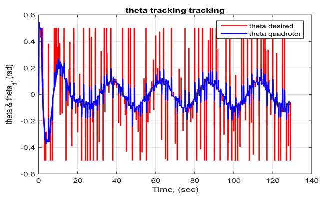

(g) Pitch tracking

Figure 3-11: Optimal control trajectory tracking

can be heavily affected and deteriorated far away from the equilibrium point. The linearization around the hovering point limits the adaptability of this control law far away from it, The classical optimal control should be adaptive to enlarge the optimality envelope, a solution based on considering the Jacobian linearization is detailed in the next chapters.

Chapter 4

Conception and realization of an open source quadrotor model

In this Chapter, the focus is made on building a quadrotor model. The hardware/software architecture for the quadrotor UAV will be detailed. The developed platform was oriented toward research and educational purposes.

4.1 Hardware listing

A basic design for a low-cost quadrotor solution is based on:

- Mechanical frame Q450. It is used to mount all components, A smaller and lighter body frame ensures better stability.
- Four Brush-less motors (1000-2000 KV).
- Electronic Speed Controller (ESC) 25-35 amp. They exploit the energy from the battery to supply the motors with variate voltages to control the motors' speed via PWM generated from the CPU/RC receiver (it is important to use fast reacting ESCs within the max current possible)(to energize the fast reacting ESC, it is recommended to look for batteries with high C factor)
- Propellers 8045: the chosen propellers should generate enough thrust at a relatively low RPM, to give more envelope of control. The "80" in "8045" represents the length

of the propeller. The "45" in "8045" indicates the pitch of the propeller which means 4.5-inch forward movement.

- Processing unit: an Arduino mega 2560 was chosen, it is a microcontroller board based on the ATmega2560 at 16 MHz crystal oscillator. It has 54 digital input/output pins on which we essentially use 15 of them as PWM connections in parallel with UART (Asy-serial communication ports Tx and Rx) and I2C protocol for master slave communication with sensors (SDK, CLK). It has as a mission to execute the flight control program and manage the communication between electronic parts.

- IMU: MPU 6050 which is a 3-axis gyroscope and a 3-axis accelerometer. The tracking precision for the gyro at a full-scale range can be ±250, ±500, ±1000, and ±2000 °/sec (dps), for the accelerometer, a full-scale range of ±2g, ±4g, ±8g, and ±16g is possible.

- The Battery: as an energy source, a LiPo battery of 2000 to 5000 mAh with a high C factor of 30+ ($di/dt \geq 30\ amp/sec$) to ensure the endurance of more than ten minutes and the good reaction of motors.

- Transmitter with receiver to be used by Pin Change Interrupt (PCI) mode. a need of at least four channels to control (height, pitch, yaw, and roll).

The complete drone is approximately 1.2 kg. an extra power distribution card is added for a better setup.

4.2 Architecture design of the hardware

The connection between the electronics parts is shown in Fig.4-1 It is important to mount the Gyro sensor with orientation as mentioned in the sensor data sheet. For the Micro-controller, it should be set in a way it makes it easy to access to periphery and sensors. The Battery should be Fixed but never connected till confirmed use. A voltage divider is crucial to watch the battery level from being discharged during the flight, once batteries are discharged, rotations of motors are directly affected and stability may be altered as well. For the ESC, it is sufficient to switch two cables to change the rotation direction of the propeller, it is crucial to mount every two adjacent propellers in opposite rotations for Yaw stability.

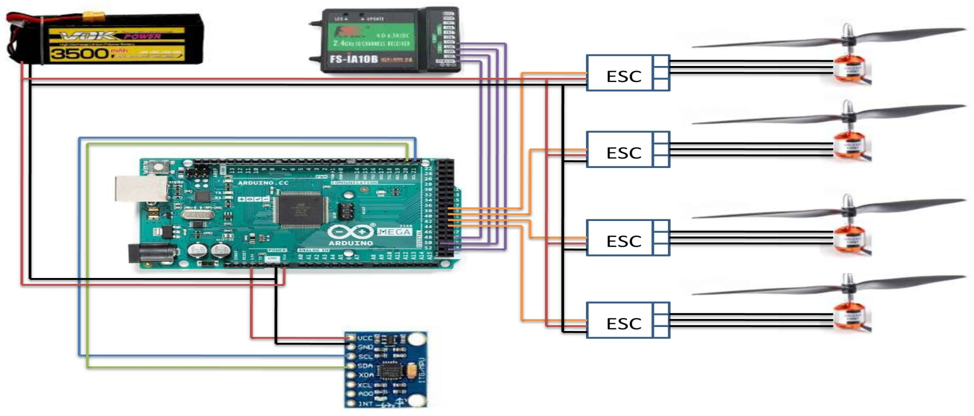


Figure 4-1: Hardware architecture

The receiver channels outputs are connected to Digital pins to read the PWM generated by the receiver with PCI mode.

4.3 Software architecture

The control code is developed in an Integrated Development Environment (IDE) in C language. The program ensures initially the calibration of the sensors, then, addresses a routine of a refresh rate of 250 Hz loop of the control, which makes a 4 ms time for each loop, based on the fact that the ESCs are 1000 to 2000 microsecond controllable, only 2 ms available time to read the sensor data and calculate the control outputs.

The Refresh rate is limited by the max refresh rate of the RC controller, and by the min laps of time of a full execution and processing time. that includes reading, corrections, and sending PWM.

The PCI routine is the way we interrupt the running program to execute a special task such as reading the new 2.4 GHz receiver signal on the PWM ports.

It is always judicious to make a gyro calibration in the static state to have a good reference, for the magnetometer, a calibration for the soft and hard iron distortion is crucial. the GYRO is connected just by CLK & SDA, where care should be made about getting two correct bytes from the same time sample and for the correct axis. for both the accelerometer and gyroscope, we use a complementary filter to smooth the reading data via the I2C protocol.

The Gyro and the accelerometer each provide two bytes of data for each axe in every iteration.

A PID correction should be set for possible rotation. The values of the PID command are calculated based on the length of the square signals of ESCs.

It is trivial that the PID of the roll and Pitch is similar, however, the Yaw angle & height PID's may be chosen as a slow response for better attitude control. The PID outputs should always keep motors running as they don't exceed a max of 2000 microsecond (1800 ms was chosen). The proportional action is used to rapidly answer the error, the derivative as a dumper, and the integral to kill the static error. The battery voltage drop-down should be considered and compensated.

4.4 Algorithm

The algorithm of control is illustrated as follows: [1]

[1]GitHub codes-Arduino IDE Code.

PID Control Algorithm

```
1. Include needed libraries
2. Define PID coefficients values for Roll, Yaw & Pitch
3. Declaration of variables:
    * Channels
    * Counters
    * Control Ui
    * Battery level
    * Calibration of gyro readings
    * PID intermediate variables,...
4. Setup of the program:
    * Define microcontroller as I2C master for communication
    * Define inputs & outputs with readiness led signs.
    * Communicate w/ GYRO and calibrate readings with a Function.
      ( During calibration, silence the ESC by zero Ui )
    * Enable "PCICR" & define mask register for pins.
    * Keep zero Ui command while waiting.
    * Define the state of the launch
5. The loop:
    * Call FCT of Gyro reading.
    * Filter the noise of reading and transfer to deg/sec.
    * Define STRT/STP sequence by joystick, use Moor machine.
    * Reset all PID for a new cycle for Roll Yaw Pitch.
    * Recalculate the set point from channels.
    * Calculate the PID FCT.
    * Define led warning and state-dependent.
    * Limit max throttle (Ui).
    * Calculate signals of ESCs.
    * Compensate the control by battery discharge.
    * Keep motors always with zero Ui in case of a waiting state.
    * Wait to complete 4 ms, (250 hz).
```

```
PID callingFCT
    * Calculate the error between gyro reading & set point.
    * Calculate Integration contribution for RYP w/ max value.
    * Calculate limited PID = PIDi + p*Error + d*(Error- exError ) for RYP.
    * Update ex-error.
Send back values.
```

```
ISR : interrupt subroutine
Once PCI is called by pin voltage change .
if (Last value = 0) then
    if (New reading is 1) then
      * Last value =1
      * Start timer
else
    if (New reading is 0) then
      * Last value =0
      * Receiver = time - timer
Send back the time of the pulse
```

4.5 Flight Experiments of the quadrotor model

Experimental results with the designed platform are presented below. The hovering mode was stable but we observed a little drift from the position due to gyro drift and level correction by the accelerometer. No GPS was mounted to correct the global position. The hovering mode experiment aimed to prove the stability of the design without pilot control and under normal conditions. Fig.4-2 shows the quadrotor on ground and then during hovering flight.

Fig.4-3 demonstrates the control of the quadrotor during flight with limited tilt angles. The quadrotor is agile and responsive to the transmitted control inputs.

4.6 Conclusion and Future work

An open-source quadrotor miniature model has been developed and built, the control was based on running a decentralized PID technique applied in an Arduino Mega development

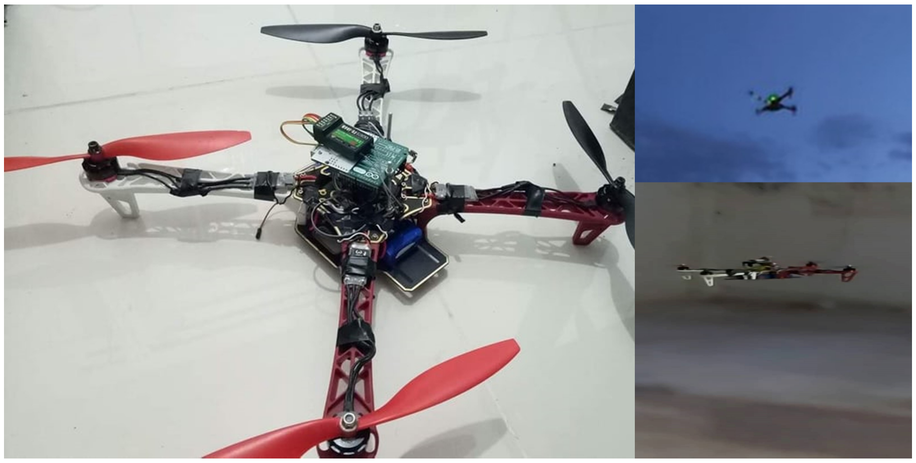


Figure 4-2: Open source developed drone in hovering mode

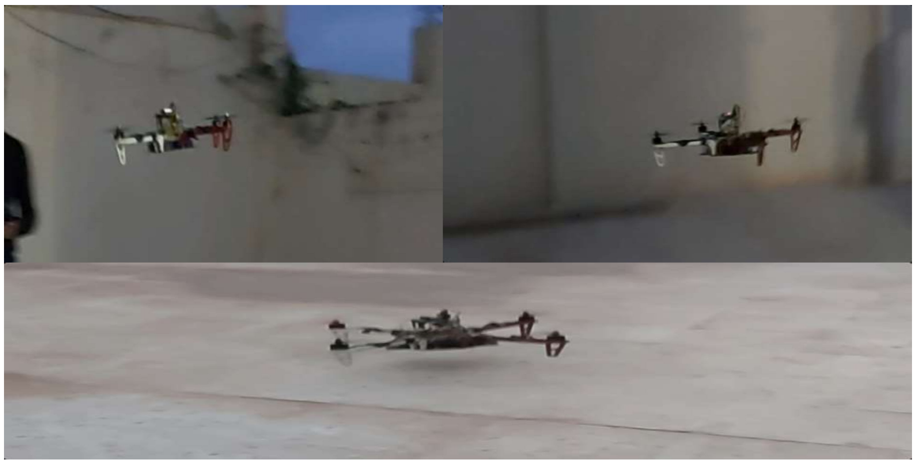


Figure 4-3: The designed quadrotor during flight

card that runs at 16 mega Hz processing microcontroller, Initially, a calibration is necessary to minimize the static error of gyro's measurement, that tends to drift after a while. The drone hardware listing was detailed, and the algorithm is explained, the main code of IDE in C language is available in the Appendix.

Future works may concern the development of a better program including self-leveling using a combination of the accelerometer with the magnetometer and precise heading exploiting the AHRS. It was proven that exploiting the quaternion is more precise than using Euler angles. A deeper challenge will be to build an optimal platform for quadrotors to execute intelligent tasks such as the developed ones in the next chapters, besides the autonomous control in either individual or swarm mission scenarios. Artificial intelligence methods could be investigated on the built quadrotor model for more autonomous applications.

Chapter 5

Intelligent Control

Random and unknown disturbances present reliability and a safety defy for quadrotor robust control, This chapter demonstrates various intelligent and adaptive techniques of control of quadrotors, exploiting a novel faster full-state observer based on an EKF enhanced by the Madgwick method, using data fusion of multiple asynchronous sensors, subjected to robust tracking of a pre-designed trajectories for different applications. The dynamics model of the quadrotor was derived using Newton Euler formalism as detailed in Eq.2.32, furthermore, its linearization was processed by the Jacobian matrix at every estimated state. The enhanced state Observer is essentially based on a Continuous-Discrete nonlinear Kalman filter Combined with the optimization of Madgwick method for quaternion orientation. The approach relies on flight dynamics predictions and gets updated by the onboard measurement of sensors at different feeding rates. Three intelligent methods were investigated, A discrete linear-quadratic Gaussian tracker was developed for best tracking performance and robustness while avoiding collision with predefined static obstacles. Model Predictive Control MPC has opted for quadrotor control to track the generated trajectory intelligently with the rejection of disturbance in a project of a smooth descent for a soft landing. Finally, a novel nested control strategy based on Adaptive RBFNN and NN supervised control embedded with IBS for a robust position and attitude trajectory tracking of quadrotor aerial robot in the presence of modeling uncertainties, sensing noise, and external bounded disturbance. Quaternion Orientations and AHRS were validated by experimental tests at less than one degree of precision error. The proposed adaptive and intelligent approaches of control of the quadrotor were all simulated for tracking control and validated in Matlab Simulink environment in the presence of Gaussian random perturbations and bounded unknown non-parametric disturbance, LQG performance has been compared to the linear inner-outer looping PID, to IBS, and decentralized Fuzzy logic control FLC strategies resulting in a validation and effectiveness of the LQG. Furthermore, Performance comparison between MPC and PID control validate the effectiveness and reliability of the proposed landing task solution. In addition, Simulation of adaptive RBFNN control philosophy proved robustness and effectiveness compared to PID, IBS, and offline decentralized CNN algorithms as it generates the control law by guarantying a fast convergence of parameters, better external disturbance compensation, and noise attenuation.

5.1 Stepping toward intelligent methods

UAVs intelligent control is considered one of the most discussed topics, especially with embedded sensors technology advances that allowed plenty of intelligent applications ensuring higher performance and robustness especially in commercial, agriculture, industrial, and even military sectors [103].

A reliable measurement is crucial for a state observer and performant control, due to the model and measurement noise and disturbance, a stochastic approach should be considered for the state observer, numerous researches opted for KF state observer and for adaptive multi models-based KF [102] [104]. In [66], Madgwick et al. developed a direct optimal estimation of quaternion orientation based on data fuse Newton Gradient descent algorithm for optimization [99] [66], the approach compensates for the Gyro drift, and it is much quicker than EKF method, however, it incorporates neither the dynamics of the model (it lacks dynamics prediction) nor the GPS position data which makes the filter limited in matter of full state observation.

Generally, EKF uses the Jacobian linearization rather than a Linear Time Invariant (LTI) as a system model besides the Navigation sensors for data update. EKF can improve SLAM, as simulated by Jiang et al. for quadrotor in The Gazebo data set combined by ROS [48], SLAM combined with state estimation through INS was researched also, for bio-inspired autonomous landing using a Kinect camera [30]. Multiple obstacle avoidance for a quadrotor was studied by [80]. In [109], researchers incorporated an EKF combined with Rauch Tung Striebel (RTS) smoother to generate accurate flight data.

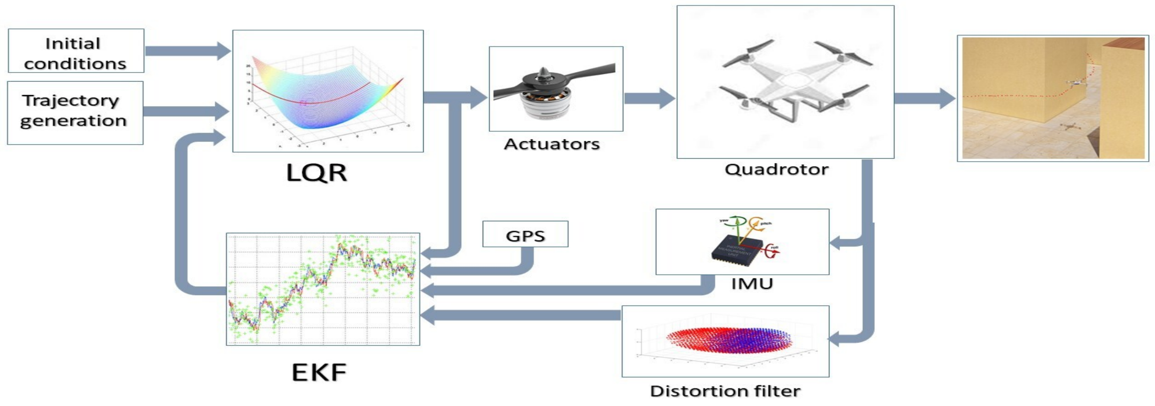


Figure 5-1: Elaborated architecture for optimal state estimation and Adaptive control for tracking: *IMU, Mag, and GPS provide enough data to the Madgwick filter that feeds quaternion orientation to the EKF, which calculates the best estimate of the augmented state of the system. Adaptive LQT controls the Quadrotor for optimal tracking of the generated Spline trajectory.*

Robust autonomous trajectory tracking with obstacle avoidance in presence of the model uncertainties and strong random disturbances has been always an objective to fulfill within the underacted quadrotor designs, especially, when considering a full state estimation during the flight [23] [90].

For optimal tracking, diverse control approaches of quadrotor were considered in the literature. Linear, Nonlinear and Intelligent methods were elaborated on the highly nonlinear underactuated model of the quadrotor system. Linear control approaches were elaborated between classic PID [19] [74] [17][61], and LQR Control, as it was considered with high-performance and robustness for quadrotor with a solution of steady Riccati equation has given the optimal gains of state feedback for the nested loops such In [68] [8],[4]. Nonlinear class of control has been addressed by many works, sibling various typologies, for robust trajectory control, Backstepping [111], [12], [55][97]. Nevertheless, internal and external random disturbances were not completely considered in those approaches although the huge perturbation to state vectors. Random disturbance comes primarily from model uncertainties and non-modeled exterior inputs, as it also results from sensing noise and actuators' ignored dynamics. Therefore comes a necessity to incorporate Gaussian noise in the control philosophy. Many researchers coped with control law subjected to Random disturbances such as LQG design, in [56], Lee et Al. developed an adaptive model-based LQG Control for path tracking with efficient noise attenuation. Adaptive control was applied in numerous papers, such in [59], a combination of Double Loop Integral Sliding Mode Control (IntSMC) and RBFNN to generate a control with perturbation compensation was successfully originated. Duori et al. adopted tracking error optimization via stochastic wind disturbance and solved the problem in efficient computational load. reinforcement learning has opted to optimize the stochastic control of the speed of the quadrotor [45]. Where in [23], a new Active Disturbance Rejection Control (ARDC) with a swarm intelligent method is studied for quadrotors control for trajectory tracking and obstacle avoidance. Researchers developed an extended state observer to enhance the ARDC performance.

A Gaussian noise is firstly considered during the state estimation approach which is based on the Combined EKF and Madgwick method to determine quaternion orientation, A discrete linearized model of the quadcopter is derived by the Jacobian method to ease the implementation of the observer and exploits various asynchronous sensors data. To elaborate on the control challenge as demonstrated in Fig.5-1, the Adaptive LQR approach subject to tracking was opted and demonstrated to ensure a performant tracking of a 3D generated trajectory avoiding obstacles and compensating the internal and external Gaussian white noises.

Initially, This chapter aims to rise the challenge of intelligent control such as an adaptive LQG tracking scheme for the control of quadrotor, based on an improved EKF strategy to estimate the state vector and consider sensors biases and magnetic distortion. We developed a Novel faster robust AHRS algorithm of asynchronous data fusion filter to estimate orientation based on merging EKF and Madgwick approach enhanced by compensation of bias and calibration of distortion with the integration of GPS localization data. After that, we designed an adaptive LQG tracker for the UAV that exploits AHRS-EKF algorithm to track a 3D Spline trajectory with disturbance rejection and cancellation of the Gaussian random noise *(which represent the wind gust disturbance that affects the general position and the high-frequency noise of actuators and sensors using various Gaussian white noises)*. A Comparison of adaptive LQG control with nested loop design PID, Integral Backstepping, and FLC control performances was conducted.

Note:

We considered a noise at a high sampling rate to model the uncertainty and actuators' performance due mechanical fatigue, friction, & vibration, and at low rates for wind gust *disturbance and unmodeled dynamics.*

For the landing task, MPC was chosen for the tracking performance instead of PID, MPC is adaptive and can handle the considered disturbance and Gaussian noise.

As the proposed soft landing approach separates the process into two interconnected parts: First, the trajectory optimal generation. Then, the MPC is responsible for tracking iteratively the desired trajectory to land. The smooth velocity profile is controlled by the FLC decision-maker.

Once a feed back observer is established, several techniques were proposed to deal with quadrotor trajectory tracking perturbed by internal and external uncertainties and random disturbances, linear techniques can not ensure convergence once we move away from the operating envelop, nonlinear techniques rely extremely on dynamics modeling, which opens the doors for instability due to unmodeled effects and external forces, in addition, both of them present a limited ability to adapt to uncertainties in a matter of performance.

Thus the need for intelligent methods, which demonstrate higher performance in the matter of all recited challenges. Adaptive NN based control design based on dynamic inversion was intensively investigated for quadrotors, the optimal compensation of inversion error subsequently improves robustness to parametric uncertainty and non-linear unmodeled dynamics [88]. The Adaptive NN efficiency is due to the capabilities of nonlinear approximation and adaptive adjustment. Our NN strategy introduces an adaptive control approach that ensures robust trajectory tracking for quadrotors. It effectively handles disturbance rejection, noise reduction, and adaptation to uncertainties in system dynamics.

Understanding the strong non-linear coupling of the four rotors in the dynamics of a quadrotor is essential; This understanding leads to the development of a valuable decentralized control scheme, which involves two nested subsystems. The outer loop focuses on tracking the translational positions, while the inner loop stabilizes and tracks the desired attitude by utilizing inverse compensated dynamics.

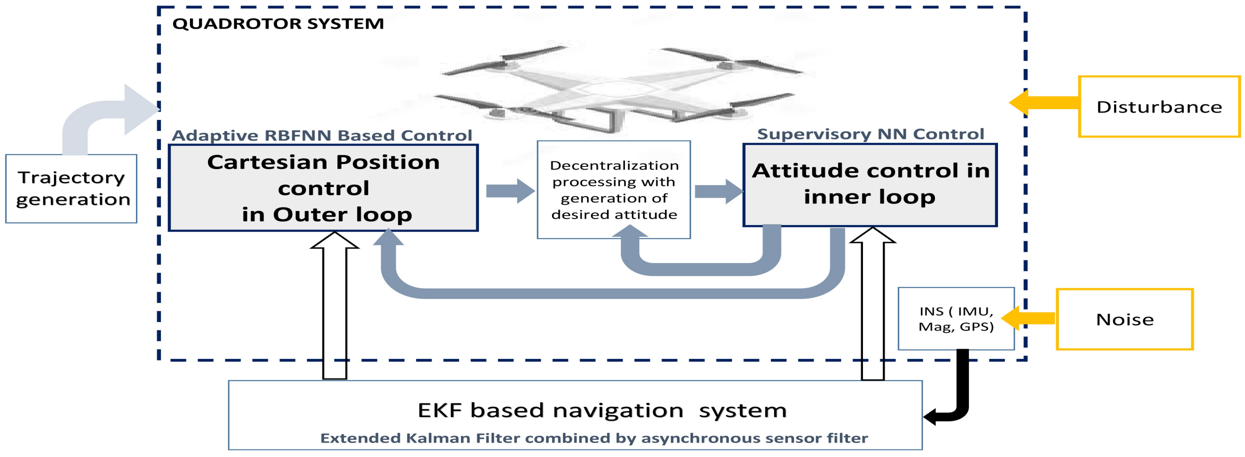

QUADROTOR SYSTEM
Disturbance
Adaptive RBFNN Based Control
Supervisory NN Control
Trajectory generation
Cartesian Position control in Outer loop
Decentralization processing with generation of desired attitude
Attitude control in inner loop
INS (IMU, Mag, GPS)
Noise
EKF based navigation system
Extended Kalman Filter combined by asynchronous sensor filter

Figure 5-2: Elaborated architecture for the adaptive RBFNN control of the translational position and the attitude supervised control, in combination with an EKF observer combined with an asynchronous filter *IMU, Mag, and GPS provide asynchornous data to EKF, the strategy set to track the desired trajectory.*

5.2 Optimal observer: Enhanced Extender Kalman Filter

To develop the AHRS, An IMU was used to provide data from the gyroscope, the magnetometer, and the accelerometer to track the motions of the quadrotor, the IMU consists of three angular rate gyroscopes and three accelerometers arranged in orthogonal orientations. These sensors measure angular velocity and linear acceleration, including the influence of the earth's gravity. However, it's important to note that these measurements inherently contain relative errors, bias, and noise.[66].

In this section, the main objective is to develop an improved Continuous Discrete EKF method for the estimation of the state vector and bias data of a miniature quadrotor UAV via asynchronous sensors data fusion for more consistency and accuracy in the presence of external disturbances including the Gaussian noises of the plant and measurement noises [102] [11].

The developed state estimator considers asynchronous feeding rates, rejects wrong data, and detects the absence of communication with sensors. During the processing of the continuous-discrete EKF, the quaternion orientation, angular velocity, linear position, ve-

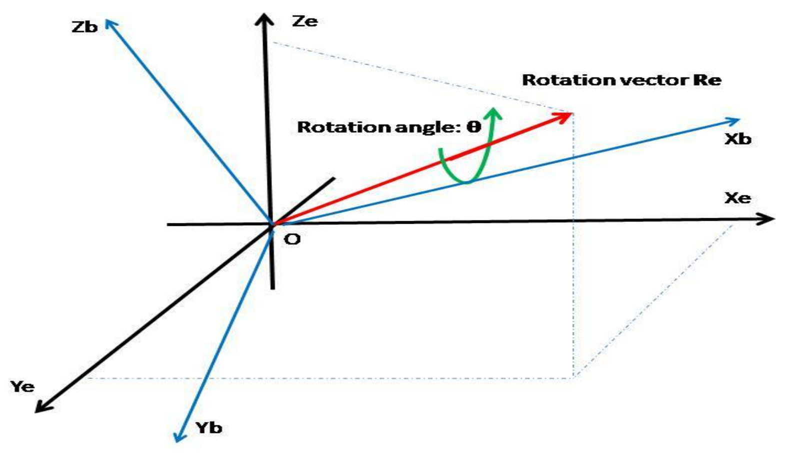


Figure 5-3: Quaternion representation, *Any rotation can be expressed by a vector and a rotation around it.*

locity and acceleration, sensor biases, and the geomagnetic vectors are all stochastically estimated.

A quaternion represented in Fig.5-3 is a four-dimensional complex number that represents the orientation of a body in motion in three-dimensional space. The Quaternion orientations were chosen to avoid the gimbal lock that may occur while using the Euler rotation angles.

In this thesis, the North East Down (NED) reference was also chosen to define the orientation, generally, Accelerometer and magnetometer are susceptible to cancel gyro integrative bias drift. Nevertheless, it is important to state that the strategy is not completely free neither of errors nor of sensors measurement noise, which affects the estimation accuracy [96].

5.2.1 Quaternion representation

The orientation of a body frame relative to the earth frame can be achieved via a rotation of an angle θ around the vector r in the earth referential. A quaternion is a four-dimensional complex number that represents the mentioned orientation of a body in motion in three-dimensional space.

$$q_B^E = [q_1 \ q_2 \ q_3 \ q_4] = [\cos\theta/2 \ \ -r_x \sin\theta/2 \ \ -r_y \sin\theta/2 \ \ -r_z \sin\theta/2]$$

A three-dimensional vector can be rotated by a quaternion using the relationship described as:

$$V_B = q_B^E \otimes V_E \otimes q_E^B \tag{5.1}$$

Where q_B^E is the conjugate of q_E^B and the symbol $\otimes$ indicates the quaternion multiplication, The quaternion product of two vectors $V_1(x_1, y_1, z_1)$ and $V_2(x_2, y_2, z_2)$ is the product of $q_1 = x_1 i + y_1 j + z_1 k$ and $q_2 = x_2 i + y_2 j + z_2 k$ as quaternions. The quaternion product q_1 q_2 works out to be:

$$q_1 * q_2 = -(x_1x_2 + y_1y_2 + z_1z_2) + (y_1z_2 - z_1y_2)i + (z_1x_2 - x_1z_2)j + (x_1y_2 - y_1x_2)k \tag{5.2}$$

Euler angles to quaternion conversion

For the quaternion representation, it is intuitive to define the quaternion of each basic rotation around every axe in Euler referential, then apply a quaternion multiplication:

$$q_B^E = q_{B\psi}^E \otimes q_{B\phi}^E \otimes q_{B\theta}^E$$

$$q_B^E = \begin{bmatrix} c(\phi/2)c(\theta/2)c(\psi/2) + s(\phi/2)s(\theta/2)s(\psi/2) \\ s(\phi/2)c(\theta/2)c(\psi/2) + c(\phi/2)s(\theta/2)s(\psi/2) \\ c(\phi/2)s(\theta/2)c(\psi/2) + s(\phi/2)c(\theta/2)s(\psi/2) \\ c(\phi/2)c(\theta/2)s(\psi/2) + s(\phi/2)s(\theta/2)c(\psi/2) \end{bmatrix} \tag{5.3}$$

Quaternion to Euler angles conversion

To go back to Euler angles formalism, we can use:

$$\begin{bmatrix} \psi \\ \theta \\ \phi \end{bmatrix} = \begin{bmatrix} \text{atan2}(2q_2q_3 - 2q_1q_4, 2q_1^2 + 2q_2^2 - 1) \\ -\arcsin(2q_2q_4 + 2q_1q_3) \\ \text{atan2}(2q_3q_4 - 2q_1q_2, 2q_1^2 + 2q_4^2 - 1) \end{bmatrix} \tag{5.4}$$

It is always convenient to normalize all quaternion orientations. the Fig.5-3 represents a quaternion rotation around an axe r_E by an angle θ.

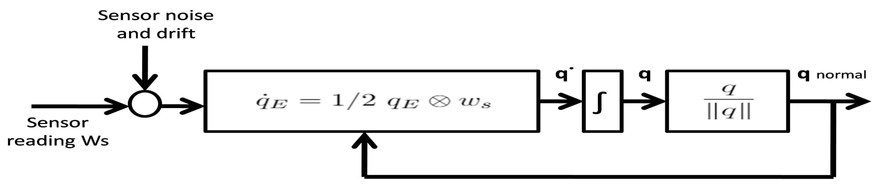


Figure 5-4: Gyroscopic angular rate quaternion filter, Normalized quaternion output.

5.2.2 Angular rate Orientation filter

Quaternion derivative can be expressed in a function of sensor angular rate w_s as follows:

$$w_s = [0\ w_x\ w_y\ w_z]^T \tag{5.5}$$

$$\dot{q}_E = 1/2\ q_E \otimes w_s \tag{5.6}$$

w_x, w_y, and w_z are the Gyroscope reading, Eq.(5.6) represents an integrative dynamic nature at a known feeding rate of the Quaternion angle in earth frame q_E. Nevertheless, the integral will not automatically produce a unit quaternion, thus, the normalization is added as shown in Fig.5-4. Integrating Gyroscope raw readings is known as dead-reckoning of angles relative to the earth frame. Still, it's too sensitive to noise and bias drift. rotations are calculated In discrete quaternion as:

$$\dot{q}^E_{w-est,i} = 1/2\ q_{est,i} * w_s \tag{5.7}$$

$$q^E_{w-est,i+1} = q_{est,i} + \dot{q}^E_{w-est,i}\Delta t \tag{5.8}$$

$q_{est,i}$ is the quaternion best estimate from the EKF at instant "i". $q^E_{w-est,i}$ is the angular rate estimate, relatively to the earth frame. The limitation of this approach is that it is prone to drift because of sensor bias. A Gyro bias vector is added to the augmented state estimation.

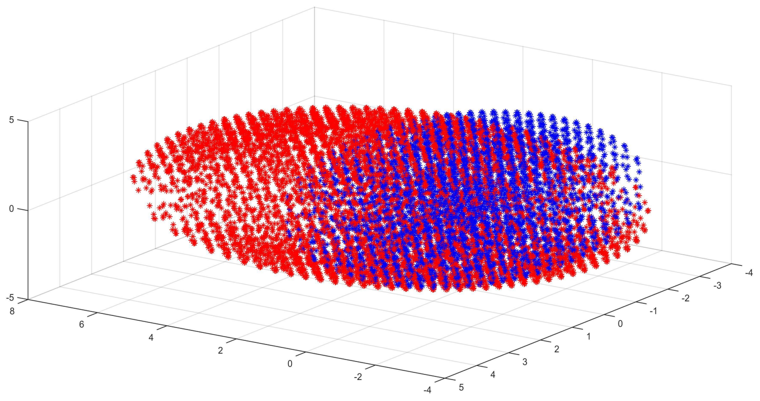


Figure 5-5: calibration of hard and soft iron magnetic distortion simulation: *Distorted Raw readings are in red, calibrated values in blue.*

5.2.3 Magnetometer and accelerations orientation filter

The accelerometer measures the magnitude and direction of the gravity field and linear acceleration of motion in the body frame, the magnetometer measures the magnitude and direction of the earth's magnetic field. The data fusion between those two sensors can lead to an orientation filter.

Magnetic distortion compensation

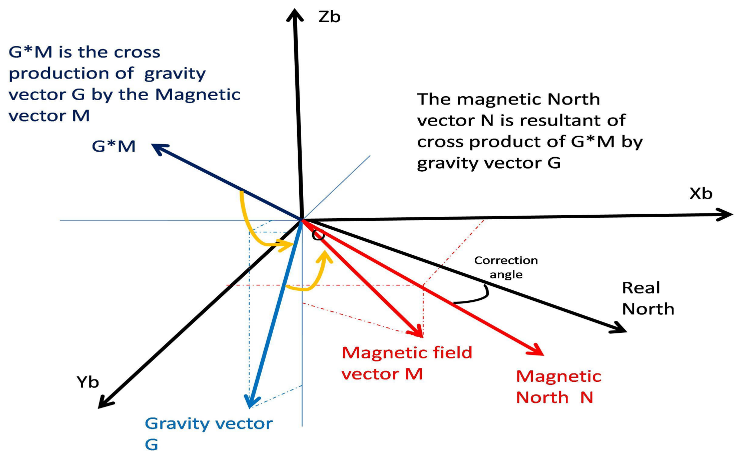
Magnetometers should be calibrated from both hard and soft iron distortions. Hard iron bias represents the offset of the center of the measurement sphere. Whereas Soft iron distortions deform the sphere to an ellipsoid as shown in Fig.5-5. The blue sphere shows the nominal magnetic field, and the red ellipsoid proves the distortion.

Eq. 5.9 represents a system of equations to solve by finding "C_i" coefficients for the elimination of both hard and soft iron distortion, H_i are known field values. [1]

$$Mag = \begin{bmatrix} c_1 & c_2 & c_3 \\ c_4 & c_5 & c_6 \\ c_7 & c_8 & c_9 \end{bmatrix} \begin{bmatrix} H_x \\ H_y \\ H_z \end{bmatrix} - \begin{bmatrix} c_{10} \\ c_{11} \\ c_{12} \end{bmatrix} \tag{5.9}$$

The accelerometer/magnetometer combination can determine the orientation of the body by measuring the gravitational acceleration in static conditions, while the magnetometer will

[1]GitHub codes- Calibration of Magnetometer.



Zb
G*M is the cross production of gravity vector G by the Magnetic vector M
G*M
The magnetic North vector N is resultant of cross product of G*M by gravity vector G
Xb
O
Correction angle
Real North
Magnetic field vector M
Magnetic North N
Yb
Gravity vector G

Figure 5-6: True orientation of magnetic field

indicate the direction of the geomagnetic north, the NED will be determined by the cross product of a gravitational vector and the magnetic north, and then by the cross product of a resultant vector with gravitational vector as demonstrated in Fig.5-6, in the last step, it is important to subtract the offset between the magnetic north and true north.

Still, the magnetometer is too sensitive to magnetic disturbance of the environment, and as a consequence, the whole determination of orientation is too noisy, thus, obtained attitude data won't be enough accurate for guidance. Besides that, the accelerometer detects and introduces all motion dynamic accelerations in readings Acc_r, especially if the sensor is off the rotation center.

$$Acc_r = g + Acc_b + \eta \tag{5.10}$$

g represents the gravitational acceleration, Acc_b is the body acceleration in the body frame, and the η is sensing noise. A well-known method to determine orientation is to use the following equations system:

$$\begin{aligned} \theta &= \arcsin(Acc_x/g) \\ \phi &= \arctan(-Acc_y/acc_z) \\ \psi &= \arctan(mag_z/mag_x) + D \end{aligned} \tag{5.11}$$

Where D is the declination between the true north and magnetic north. This value depends on the location on earth.[2]

Except, using this orientation for the quaternion estimation by Accelerometer/magnetometer is not that accurate, and for that, the intention is to calculate only the quaternion rotation between instant "i" and "$i+1$" and then consider it to be the best estimate of instant "i" noted as $q^E_{a-est,i}$. The motion of the quadrotor moves the body frame from initial orientation to the instant "i" and is noted by S_i:

$$S_i = q_{est,i} \otimes N_{est} \otimes q^*_{est,i} \tag{5.12}$$

Where N_{est} is the true north initial estimate. At the instant t_{i-1}:

$$q_{est,i+1} = q_{est,i} \otimes \Delta q_{i+1} \tag{5.13}$$

Δq_{i+1} represents the quatornian rotation during $[t_i,\ t_{i+1}]$, which is equivalent to the rotation $[\delta\phi, \delta\theta, \delta\psi]$, based on last best filter estimate $q_{est,i}$, we can write:

$$S_{i+1} = \Delta q_{i+1} \otimes S_i \otimes \Delta q^*_{i+1} \tag{5.14}$$

Madgwick formulated an optimization function to solve the quaternion orientation from an equivalent formula, where a direct geometric resolution can be done using the normalized cross product of S_{i+1} and S_i to determine the rotation vector R and calculate the 3D angle between the vectors as following:

$$\begin{gathered}
R = S_i * S_{i+1} \\
R = R/norm(R) \\
\beta = (\sqrt{(S_i(1)^2 + S_i(2)^2 + S_i(3)^2) * S_{i+1}(1)^2 + S_{i+1}(2)^2 + S_{i+1}(3)^2)} \\
\alpha = \arccos((S_i(1) * S_{i+1}(1) + S_i(2) * S_{i+1}(2) + S_i(3) * S_{i+1}(3))/\beta \\
\Delta q_{i+1} = [\cos(\alpha/2)\ R(1)\sin(\alpha/2)\ R(2)\sin(\alpha/2)\ R(3)\sin(\alpha/2)]
\end{gathered} \tag{5.15}$$

Finally, Apply a saturation for the value if does exceed the calculated threshold and call for Eq.5.13 to find the accelerometer/magnetometer best orientation estimate.

[2]GitHub codes-Orientation filter.

5.2.4 Data fusion for Orientation

Quatornian estimations of the quadrotor using the angular rate quaternion $q_{w-est,i}$ and by accelerometer/magnetometer method $q_{a-est,i}$, are data fused using an adaptive optimal complementary filter.

$$q_{est,i} = \gamma_i \; q_{a-est,i} + (1-\gamma_i) \; q_{w-est,i} \; , \qquad 0 < \gamma < 1 \tag{5.16}$$

γ_i is an iterative variable that finds optimality when the weight of angular rate divergence of $q_{w-est,i}$ equals weight convergence of $q_{a-est,i}$.

$$\gamma_i \; err_{a,m} = (1-\gamma_i) err_w \tag{5.17}$$

err_w is the magnitude of the derivative corresponding to the gyroscope measurement error and $err_{a,m}$ is the magnitude of the quaternion error of the accelerometer/magnetometer method. the γ value is processed as a penalty for the bad estimation method.

$$\gamma = err_w / (err_w + err_{a,m}) \tag{5.18}$$

A "NEO -6M" GPS sensor is used to provide position at a 5 Hz rate, and a standard National Marine Electronics Association (NMEA) messaging was exploited to extract data on longitude, latitude, and altitude [1]. The transformation between GPS waypoint and general Cartesian position $[x \; y \; z]^T$ coordinates is based on the following equations:

$$\begin{cases} x = R * \cos(lat) * \cos(lon) - x_0 \\ y = R * \cos(lat) * \sin(lon) - y_0 \\ z = R * \sin(lat) - z_0 \end{cases} \tag{5.19}$$

Where R is the approximate earth radius(6371 km) at the running region, "lon" and "lat" are the longitude and latitude converted to radians respectively. Data on altitude is referenced to World Geodetic System (WGS 84) which is almost equivalent to sea level.

5.2.5 Design of enhanced Kalman Filter gains

The EKF is a stochastic optimal filter that estimates the states with sensors' noise based on the spectral content of the measured and previously estimated data. [75]. Designing a continuous Kalman Filter on the direct non-linear model will be extremely challenging and will limit the ability of analysis and implementation besides the expensive computational of the Riccati differential equation. Therefore, we opted for the discrete LTV model via the Jacobian linearization method applied at the estimated operating point. The Madgwick filter method for data fusion is a very efficient way that doesn't incorporate the dynamics of the drone which makes the filter blind to prediction data, but it is less time expensive than EKF and shows good convergence results and stability at low velocities. [66].

In our solution, we used partial Estimation from Madgwick filters as a first stage to provide measurements and best estimations to the EKF.

After got linearized the dynamics model, an augmented model including disturbance model and sensors data is obtained, then discretized on the sampling time of $T_s = 0.01$ sec [7]. Based on The update rate of the IMU at 100 Hz and 5 Hz for the GPS. because of the asynchronous rating, a small verification program initiates the filter to verify the availability of data and confirms the validation of measurement iteration, Therefore, a multi-rate extended Kalman filter was considered.

The AHRS fused with GPS data will represent the measurement data, and predictions will be based on the Jacobian linearized model. An augmented state vector $Xaug = [x \;\; bias_{vector}]^T$ of 31 elements is tracked by the proposed filter. All states are described in Table.5.1

Table 5.1: State vector elements

States	Units	Index
Orientation (quaternion)		1:4
Orientation (Euler angles)	rad	5:7
Angular Velocity (w_x, w_y, w_z)	rad/s	8:10
Position (NED)	m	11:13
Velocity (NED)	m/s	14:16
Acceleration (NED)	m/s^2	17:19
Accelerometer Bias (XYZ)	m/s 2	20:22
Gyroscope Bias (XYZ)	m/s 2	23:25
Geomagnetic Field Vector (NED)	uT	26:28
Magnetometer Bias (XYZ)	uT	29:31

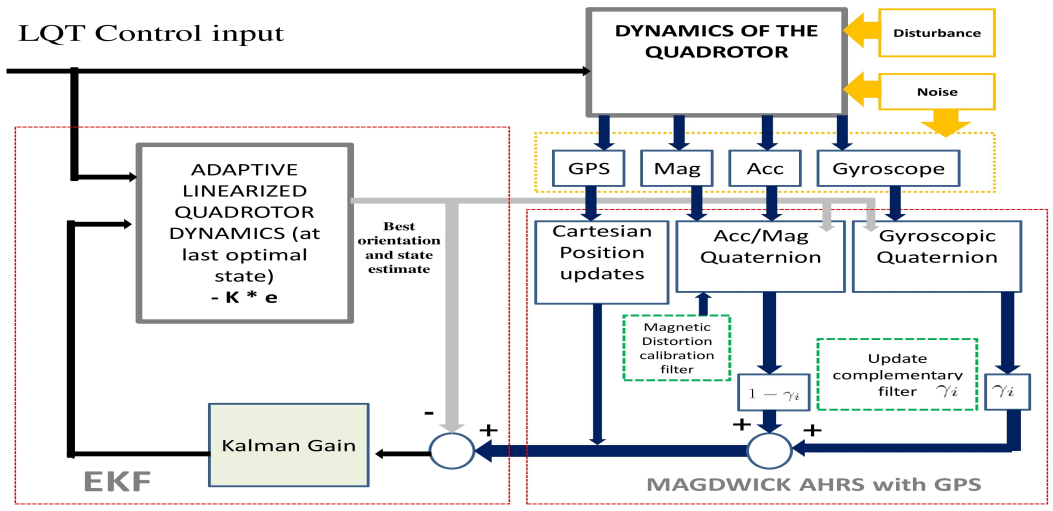

LQT Control input
DYNAMICS OF THE QUADROTOR
Disturbance
Noise
GPS
Mag
Acc
Gyroscope
ADAPTIVE LINEARIZED QUADROTOR DYNAMICS (at last optimal state) - K * e
Best orientation and state estimate
Cartesian Position updates
Acc/Mag Quaternion
Gyroscopic Quaternion
Magnetic Distortion calibration filter
Update complementary filter γ_i
$1-\gamma_i$
γ_i
Kalman Gain
EKF
MAGDWICK AHRS with GPS

Figure 5-7: Enhanced Madgwick/EKF Design with pre-filtering and data fusion

Quaternion orientation are resulting from EKF [3] combined with AHRS / GPS sensors filtering, The Euler angles are transformed from Quaternion orientation instantaneously, and The angular velocities are generated from quaternion dynamics instead of the raw data of the Gyroscope. Accelerations to earth reference and accelerometer bias are from dynamics and error calculated between accelerometers and final estimation. The Gyroscope biases are resulting from errors between gyroscope readings and estimation of angular rates. The same analogy for the magnetometer.

After the calibration of the magnetometer, The filter is initialized in a static condition at the origin. with pre-definition of sensors' noise covariance based on data sheets [34]. The flow chart 5.2.7 details the algorithm of the enhanced observer.

$$\begin{gathered}
R.Quater = 1e-2 * I(4,4) \\
R.Gyro = 100 * I(3*3) \\
R.Acc = 100 * I(3*3) \\
R.Mag = 1e-7 * I(3*3) \\
Q.system = 1e-3 * I(12*12)
\end{gathered} \tag{5.20}$$

[3]GitHub codes-Kalman Filter.

The iterative equations of EKF will be as follow[4]:

$$\begin{cases} A_i = \partial F/\partial x|_{x_i,u_i} \\ B_i = \partial F/\partial u|_{x_i,u_i} \\ \hat{x_0}^- = \hat{x}^-(0) \\ P_0^- = E[\tilde{x_0}^-\tilde{x_0}^-] \\ K_i = P_i^- C_i^T [C_i P_i^- C_i^T + R_i]^{-1} \\ \hat{x_i}^+ = \hat{x_i}^- + K_i[y_i - C_i\hat{x_i}] \\ P_i^+ = (I - K_iC_i)P_i^-(I - K_iC_i)^T + K_iR_iK_i^T \\ \hat{x_{i+1}}^- = A_i\hat{x_i}^+ + B_iu_i \\ \hat{P_{i+1}^-} = A_iP_i^+A_i^T + G_iQ_iG_i^T \end{cases} \tag{5.21}$$

5.2.6 Demonstration of Faster convergence than EKF

For a system dynamics written as:

$$\begin{aligned} \dot{X} &= AX + BU + GW \\ Y &= CX + V \end{aligned} \tag{5.22}$$

Where $W(t)$ and $V(t)$ are uncorrelated Gaussian process disturbance and sensing noise respectively. Let's consider $\hat{X}(t)$ the state vector estimate and $\hat{X}_M(t)$ the best estimate of state from Madgwick method.

$$\tilde{X}(t) = X(t) - \hat{X}(t) \tag{5.23}$$

An optimal Kalman observer dynamics can be illustrated as follows:

$$\dot{\hat{X}} = A\hat{X} + BU + K_e(Y - \hat{Y}) \tag{5.24}$$

K_e is the optimal Kalman Gain. The Error dynamics $\dot{\tilde{X}}(t)$ can be developed as follows:

$$\dot{\tilde{X}}(t) = \dot{X}(t) - \dot{\hat{X}}(t) \tag{5.25}$$

$$\dot{\tilde{X}}(t) = \dot{X}(t) - \dot{\hat{X}}_M(t) + \dot{\hat{X}}_M(t) - \dot{\hat{X}}(t) \tag{5.26}$$

[4]GitHub codes-Adaptive Kalman Filter.

Based on the mathematical exponential convergence of Madgwick formulation that uses Newton method [66], the estimation error $\tilde{X}_M(t)$ is a descending function and can be written as:

$$\dot{\hat{X}}_M(t) - \dot{\hat{X}}(t) = \eta\dot{\tilde{X}}(t) \tag{5.27}$$

$\eta \in R^n$, a diagonal matrix of η_i elements such as $\eta_i \in [0,1]$, the matrix η is trivially positive definite; by substitution,

$$\dot{\tilde{X}}(t) = \dot{X}(t) - \dot{\hat{X}}_M(t) + \eta\dot{\tilde{X}}(t) \tag{5.28}$$

As the Madgwick method error is part of the combined filter, we can write:

$$\dot{\tilde{X}}(t) = \dot{X}(t) - \dot{\hat{X}}_M(t) + \eta\dot{\tilde{X}}(t) \tag{5.29}$$

Therefor,

$$\dot{\tilde{X}}(t) = (I-\eta)^{-1}(\dot{X}(t) - \dot{\hat{X}}_M(t)) \tag{5.30}$$

$$\dot{\tilde{X}}(t) = (I-\eta)^{-1}(AX + BU + GW - A\hat{X} + BU + K_e(Y - \hat{Y})) \tag{5.31}$$

$$\dot{\tilde{X}}(t) = (I-\eta)^{-1}((A - K_eC)\tilde{X} + GW - K_eV) \tag{5.32}$$

$$\dot{\tilde{X}}(t) = (I-\eta)^{-1}A_0\tilde{X} + (I-\eta)^{-1}(GW - K_eV) \tag{5.33}$$

Both disturbance and noise drive the dynamics of error, however, If we can ensure the convergence of EKF alone, we can ensure the convergence with a bounded noisy input too. The error dynamic is mainly decided by the all left half plane eigenvalues of the matrix $(I-\eta)^{-1}A_0$ noted by $\lambda_i((I-\eta)^{-1}A_0)$.

$$\lambda_i(A_0) \geq \lambda_i((I-\eta)^{-1}A_0) \tag{5.34}$$

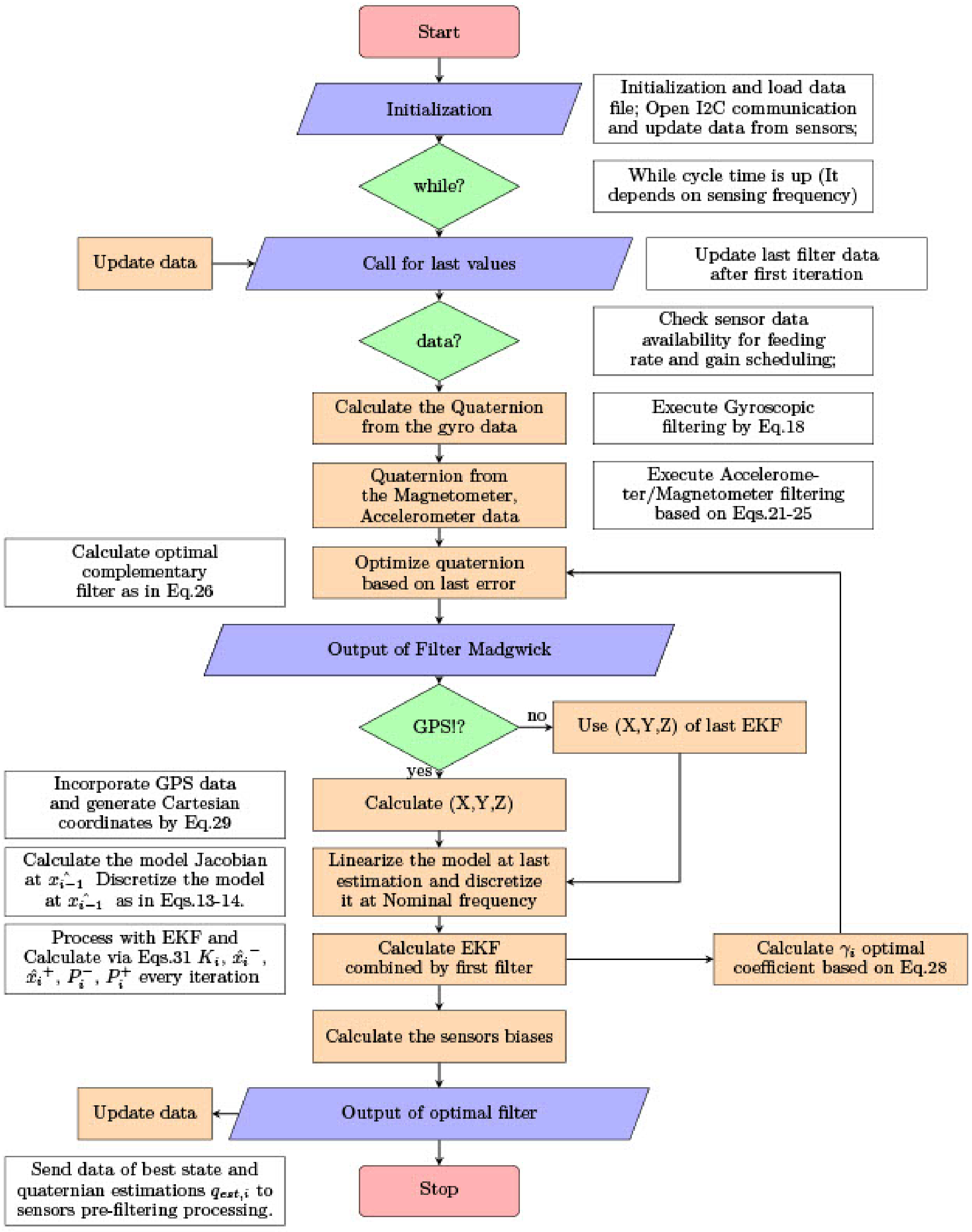
Thus, a Faster exponential convergence of the combined filter between Madgwick and EKF.

5.2.7 Flow chart of the Optimal Observer

5.2.8 Magnetometer calibration

Fig. 5-8 shows the hard and soft distortion calibration of the MPU 9250 magnetometer readings. the kit was communicated to Matlab Via Arduino, Readings were taken first and

3.6 Flow chart of the Optimal Observer


Start
Initialization
Initialization and load data file; Open I2C communication and update data from sensors;
while?
While cycle time is up (It depends on sensing frequency)
Update data
Call for last values
Update last filter data after first iteration
data?
Check sensor data availability for feeding rate and gain scheduling;
Calculate the Quaternion from the gyro data
Execute Gyroscopic filtering by Eq.18
Quaternion from the Magnetometer, Accelerometer data
Execute Accelerometer/Magnetometer filtering based on Eqs.21-25
Calculate optimal complementary filter as in Eq.26
Optimize quaternion based on last error
Output of Filter Madgwick
GPS!?
no
Use (X,Y,Z) of last EKF
yes
Incorporate GPS data and generate Cartesian coordinates by Eq.29
Calculate (X,Y,Z)
Calculate the model Jacobian at $\hat{x_{i-1}}$ Discretize the model at $\hat{x_{i-1}}$ as in Eqs.13-14.
Linearize the model at last estimation and discretize it at Nominal frequency
Process with EKF and Calculate via Eqs.31 K_i, $\hat{x}_i^-$, $\hat{x}_i^+$, P_i^-, P_i^+ every iteration
Calculate EKF combined by first filter
Calculate γ_i optimal coefficient based on Eq.28
Calculate the sensors biases
Update data
Output of optimal filter
Send data of best state and quaternian estimations $q_{est,i}$ to sensors pre-filtering processing.
Stop

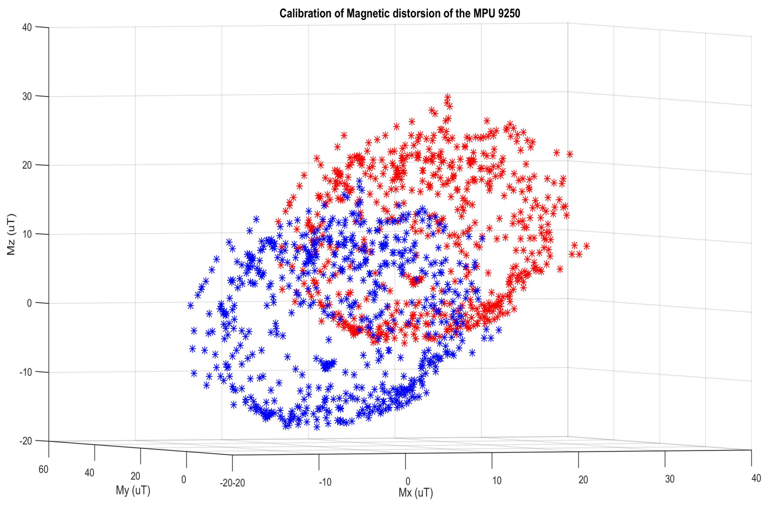


Figure 5-8: Online Calibration of Magnetometer by Matlab and the MPU 9250/Arduino.

set into a system of equations as shown in Eq.5.9. Results, showed noisy heading orientation by the Magnetometer alone which leads to the necessity of the developed filter.

5.2.9 Sensors fusion and EKF

Extensive simulation and implementation results in static to low-velocity conditions, showed that the proposed strategy has an excellent performance in terms of bias and states estimation.

The experimentation of Data fusion and EKF was performed using MPU 9250 and GPS at a low angular velocity, the algorithm was implemented in Matlab using Arduino Card to import data from a sensor via I2C ports. Fig.5-10 shows the orientation filter results.

Obtained estimations of angles indicated that the designed filter can achieve an accuracy in the order of less than 1 degree. Full state estimation in both indoor/outdoor environments is achieved via a new approach of EKF estimator exploiting multi-sensor data fusion. the GPS and the MPU9250 10 DOF sensor were used for angles and translation observation.

5.2.10 Gyroscope bias drift

Another depth of estimation algorithm performance is illustrated in the real-time gyro drift estimation, Figure.5-9 shows an estimation of angular rate with bias estimation. The gyro-

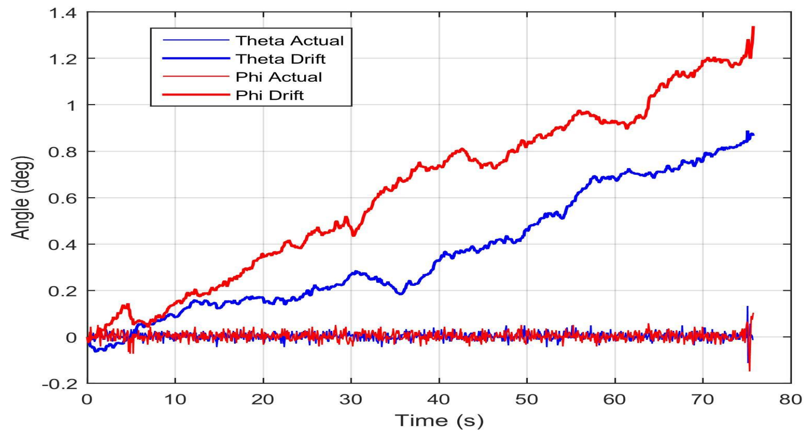


Figure 5-9: Drift of angles of Gyro by dead reckoning vs actual estimation by AHRS

Table 5.2: Mean orientation tracking errors

Angles of orientation	Roll ϕ	Pitch θ	Yaw ψ
Error (deg)	0.43	0.34	0.65

based estimation is divergent because of that bias and dead reckoning will drift away after a while.

5.2.11 Results of sensor fusion and EKF of the quadrotor

The Data fusion filter combined with the EKF exploits the MPU 9250 and the GPS with a calibrated magnetometer. AHRS algorithm calculates the quaternion orientation. Fig.5-10 presents the AHRS filter's real-time performance by interfacing Matlab with the sensor via Arduino. AHRS filter showed a responsive tracking of real motion orientation. Mean tracking errors are detailed in the table.5.2.

A Matlab simulation of the EKF observer of linear position is shown in Fig.5-11, the Observer can reject noise and uncertainty of dynamics. The EKF is crucial to ensure loop stability during tracking, in the presence of white noise applied as sensor noise and wind gusts. The wind gust was simulated as a Gaussian noise at a low feeding rate, whereas, a high feeding rate white noise was simulated and applied on actuators and sensor reading.

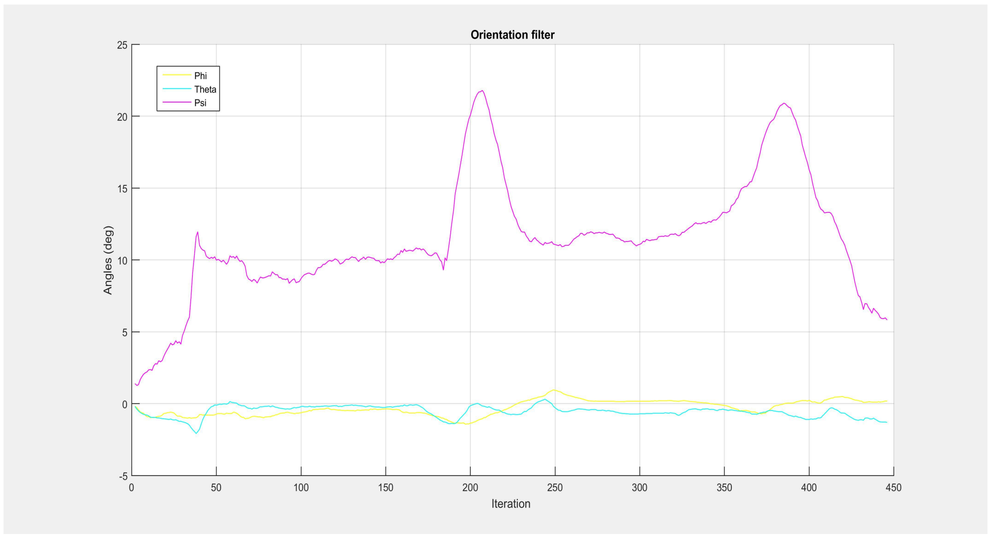


Figure 5-10: AHRS observer for the general position of the quadrotor, Real-time reading by interfacing Matlab and MPU-9250.

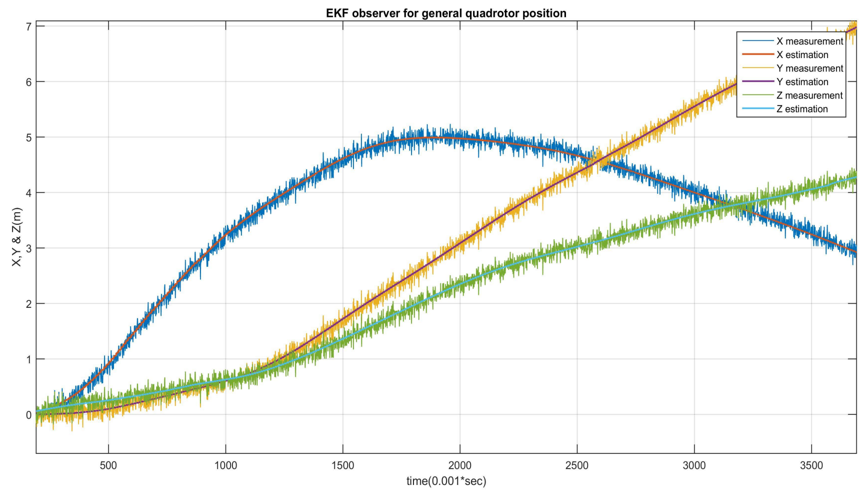


Figure 5-11: EKF observer for the general position of the quadrotor with Gaussian noise attenuation

5.3 Tracking by Adaptive Linear quadratic Gaussian Control

LQG is a fundamental optimal control problem. It has been chosen because of the quadrotor disturbed nature, it is suitable for dynamic models perturbed by a white Gaussian noise and disturbance, the control optimization has the objective to determine a state feedback control minimizing a quadratic cost criterion [50].

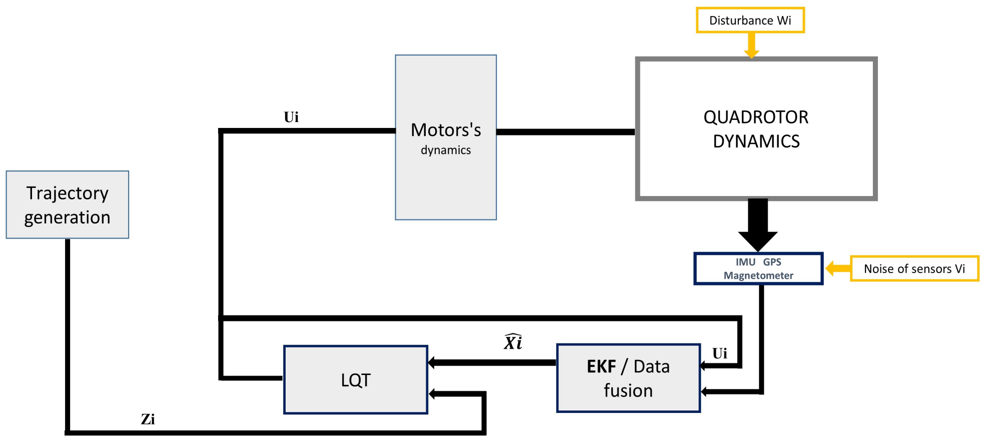


Figure 5-12: Scheme of adaptive LQG control structure (LQT: Linear quadratic tracker)

LQG is unique and it is optimized by combining a LQE with a LQR under the separation principle of states, which leads to designing the control and the observer independently.

The Adaptive LQG will be applied on a LTV model, Classical LQG implementation may be problematic at higher state dimension. Reduced model order for LQG affects the separation principle and the solution is no longer unique. LQG control may also have a robustness problem which is not guaranteed [98], The robustness of stability of the closed-loop system must be checked separately within the LQG design. To promote robustness some of the system parameters may be assumed stochastic instead of deterministic [38]. The considered LQG design is shown in Fig.5-12. where a combination between an LQT and an enhanced Kalman LQE was assembled.

5.3.1 Linear Quadratic Gaussian Tracker

The LQR is a full-state feedback optimal controller, which is derived from an optimization process of the control problem applied on an LTV (rather linearized) plant. Discrete linearized dynamics of quadrotor at instant indexed by "i" can be represented as:

$$\begin{aligned} \dot{x}_{i+1} &= A_i x_i + B_i u_i + w_i \\ y_i &= C_i x_i + v_i \end{aligned} \tag{5.35}$$

Where v_i,w_i represent discrete-time Gaussian white noise processes with covariance matrices Q and R respectively.

The quadratic cost function "J" for tracking the desired reference is noted by Z_i[14].

$$\begin{gathered} J = 1/2(C_N x_N - Z_N)^T S_N (C_N x_N - Z_N) + \\ 1/2 \sum_{i=1}^{N-1} \left((C_i x_i - Z_i)^T Q_i (C_i x_i - Z_i) + u_i^T R_i u_i \right) \end{gathered} \tag{5.36}$$

Such,

- Pairs $\{A_i, B_i\ \}$, controllable
- $S_N \geq 0\ ;\ Q \geq 0\ (psdf)\ \&\ R > 0\ (pdf)$

The aim is to find an affine optimal control of the quadrotor to track the desired Spline trajectory generated noted by Z_i. Control will be expressed in a state-feedback term for nominal correction and a feed-forward term that anticipates our desired reference position [14].

The control will be such:

$$u_i = -K_i \hat{X}_i + L_i g_{i+1} \tag{5.37}$$

Using the Hamiltonian on last performance index J:

$$\begin{gathered} H = 1/2 \sum_{i=1}^{N-1} \left((C_i x_i - Z_i)^T Q_i (C_i x_i - Z_i) + u_i^T R_i u_i \right) + \\ + \lambda_{i+1}^T (A_i x_i + B_i u_i) \end{gathered} \tag{5.38}$$

- The state equation:

$$x_{i+1} = A_i x_i + B_i u_i \tag{5.39}$$

- The Co-state equation:

$$\lambda_i = C_i^T Q_i C_i x_i - C_i^T Q Z_i + A_i^T \lambda_{i+1} \tag{5.40}$$

- The Optimality:

$$u_i = -R_i^{-1} B_i^T \lambda_{i+1} \tag{5.41}$$

- Boundary condition:

$$\lambda_N = C_N^T S_N C_N x_N - C_N^T S_N Z_N \tag{5.42}$$

Assuming,

$$\lambda_i = P_i x_i - g_i \tag{5.43}$$

Substitute λ_{i+1} in control equation and inject the result in state equation:

$$\dot{x}_{i+1} = \Big(I + B_i R_i^{-1} B_i^T P_{i+1}\Big)^{-1} \Big(A_i x_i + B_i R_i^{-1} B_i^T g_{i+1}\Big) \tag{5.44}$$

Let's put $V_i = C_i^T Q_i C_i$ and $W_i = C_i^T Q_i$,

From the costate equation we can get:

$$\begin{gathered} \lambda_i = V_i x_i - W_i Z_i + A_i^T (P_{i+1} \Big(I + B_i R_i^{-1} B_i^T P_{i+1}\Big)^{-1} \\ (A_i x_i + B_i R_i^{-1} B_i g_{i+1}) - g_{i+1}) \end{gathered} \tag{5.45}$$

Substitute the costate by the assumption:

$$\begin{gathered} P_i x_i - g_i = V_i x_i - W_i Z_i - A_i^T g_{i+1} + A_i^T P_{i+1} \\ \Big(I + B_i R_i^{-1} B_i^T P_{i+1}\Big)^{-1} (A_i x_i + B_i R_i^{-1} B_i g_{i+1}) \end{gathered} \tag{5.46}$$

This later represents a polynomial equation of first order that holds good for all x_i, so:

$$\begin{cases} P_i = V_i + A_i^T P_{i+1} \Big(I + B_i R_i^{-1} B_i^T P_{i+1}\Big)^{-1} A_i \\ g_i = W_i Z_i + A_i^T \Big(I - P_{i+1} (I + B_i R_i^{-1} B_i^T P_{i+1})^{-1} \\ \qquad B_i R_i^{-1} B_i^T\Big) g_{i+1} \end{cases} \tag{5.47}$$

The system of equations represented in Eq.5.47 is recursive, and its solvable for the whole horizon if $I + B_i R_i^{-1} B_i^T P_{i+1}$ is invertible and final values P_N and g_N are known. as λ_N satisfies the boundary condition and the assumption at time N:

$$\lambda_N = P_N^T x_N - g_N = C_N^T S_N C_N x_N - C_N^T S_N Z_N \tag{5.48}$$

Therefor:

$$\begin{cases} P_N = C_N^T S_N C_N \\ g_N = C_N^T S_N Z_N \end{cases} \tag{5.49}$$

Finally a control solution maybe written by

$$\begin{aligned} u_i = &-\big[(R_i + B_i^T P_{i+1} B_i)^{-1} B_i^T P_{i+1} A_i\big] x_i \\ &+\big[(R_i + B_i^T P_{i+1} B_i)^{-1} B_i^T\big] g_{i+1} \end{aligned} \tag{5.50}$$

The control strategy for optimal tracking is implemented iteratively, incorporating two distinct components. The first component is a state-feedback control, which utilizes the current state of the system to determine the control input. The second component is a predictive tracking control input, which considers trajectory data to optimize the tracking performance.

5.3.2 Control simulation

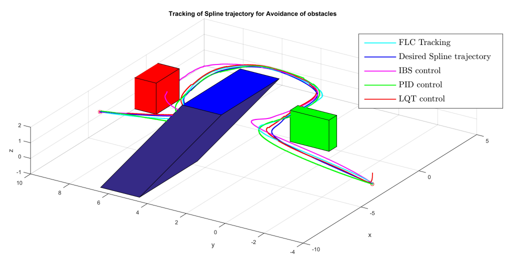
Besides the adaptive LQG control, three other approaches have been searched to validate the performance of the opted method. The Linear PID, the non-linear IBS, and the intelligent FLC were developed to track the Spline trajectory. The PID approach was based on converting the MIMO coupled model to multi-SISO systems by decoupling the subsystems as developed by [73], The inner outer nested loops strategy was considered for that. The IBS is a nonlinear approach that contributes the integral benefits in the backstepping design, Four controllers resulting from the under-actuation nature of quadrotor which is based on just four controllers capability. Lyapunov theory was used to derive the attitude and altitude controllers. In the FLC strategy, Model inversion was first used, then four FLC controllers for essential quadrotor control. Each inference unit used the error and the error rate of change as inputs[5].

All control approaches were simulated for the same tracking task to avoid the obstacles.

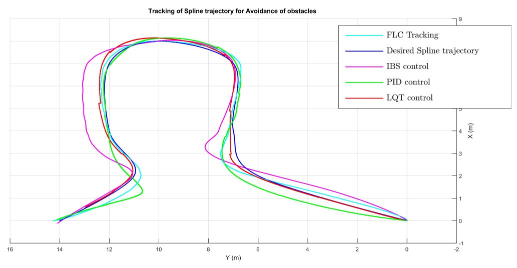
The simulation of the adaptive LQG control showed excellent stabilization and tracking of desired Spline trajectory as demonstrated in Fig.5-14, a comparison to the three other control approaches was established for validation, The error of the tracking of the Adaptive LQG method was minimal compared to other methods performance. Adaptive LQG was validated as an approach for quadrotor control to track the desired Spline curves to avoid obstacles, the rejection of perturbation was assured relative to added noises.

The comparison to PID, IBS, and FLC control, showed that the adaptive LQG demonstrates better performance although the expensive time of calculation and algorithmic com-

[5]GitHub codes-Obstacles avoidance simulation.



(a) Panoramic Tracking with Obstacles avoidance



(b) Top view of tracking

Figure 5-13: Top view of the tracking of the Spline trajectory, the green trajectory is based on PID control, the sky blue colored trajectory is for the FLC method, where the blue and red are the desired Spline and A-LQG control trajectories respectively, the IBS control approach is in magenta.

plexity.

Linear position tracking showed minimal tracking error, the wind gust effect by low-rate Gaussian noise is observable on the quadrotor position, with the quick correction to track. Angles tracking was also excellent and better than PID attitude results, all observing the noise effect introduced on the sensor by the high-rate Gaussian noise.

The IBS control loses tracking narrow turns and showed a weak rejection of perturbations, on the other hand the PID, showed better tracking with less overshooting in turns, FLC comes close in a matter of performance of tracking to the optimal adaptive LQG.

5.4 Model Predictive Control

5.4.1 Control and Tracking of designed trajectory

MPC is an optimal control that satisfies a set of constraints. MPC is chosen based on the quadrotor nature, the multiple manipulative variables with constraints to meet, and the disturbance which has been considered as model uncertainties, and as white Gaussian noise during the simulation. Jacobian of quadrotor dynamics is used to linearize the model, the Adaptive design will be based on linearization at every operating points [38]-[91].

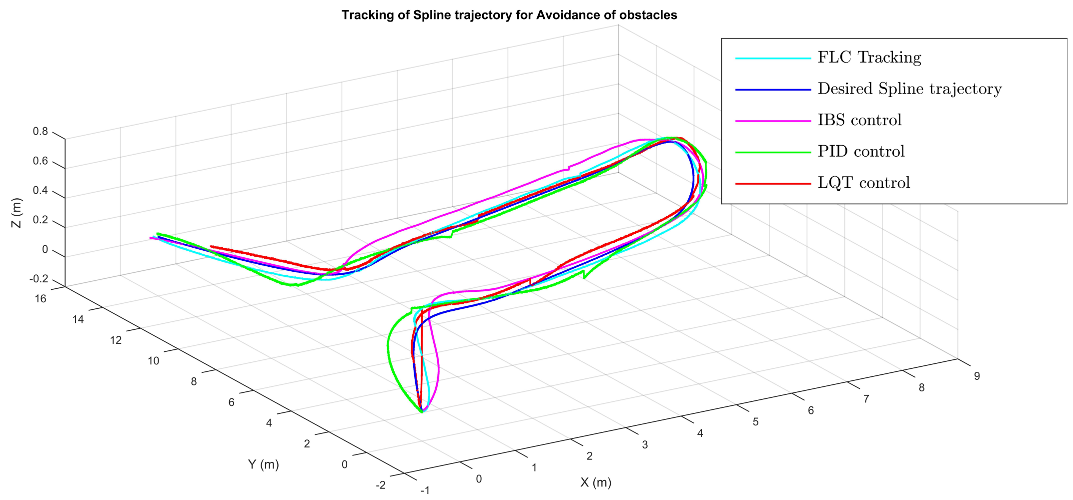


Figure 5-14: 3D Spline trajectory tracking performance by PID, IBS, and FLC vs Adaptive LGQ, the Desired trajectory is in blue which is Spline designed, Green trajectory represents the PID control results, and IBS and FLC are in magenta and sky blue color, where we can see tracking errors affected by the introduced noise and perturbation, However, Adaptive LQG control trajectory in Red demonstrates better robustness and performant tracking

5.4.2 Tracking Control strategy

The MPC is a finite-horizon optimization based on the dynamics model. Fig. 5-1 shows the control strategy with MPC to track the generated min Jerk trajectory with the fuzzy velocity control. The system can be written as follows:

$$\dot{x}_{i+1} = A_i x_i + B_i u_i + w_i \tag{5.51}$$

$$y_i = C_i x_i + v_i \tag{5.52}$$

v_i,w_i are discrete-time Gaussian white noises with the respective covariance matrices Q and R. To ensure optimal tracking [14], a quadratic cost function "J" is defined as:

$$\begin{aligned} J = 1/2\ (C_N x_N - Z_N)^T\ S_N\ (C_N x_N - Z_N) + \\ +1/2 \textstyle\sum_{i=1}^{N-1} \left((C_i x_i - Z_i)^T Q_i (C_i x_i - Z_i) + u_i^T R_i u_i\right) \end{aligned} \tag{5.53}$$

Z_i represent the Minimal Jerk trajectory along the horizon N. The optimal Control can

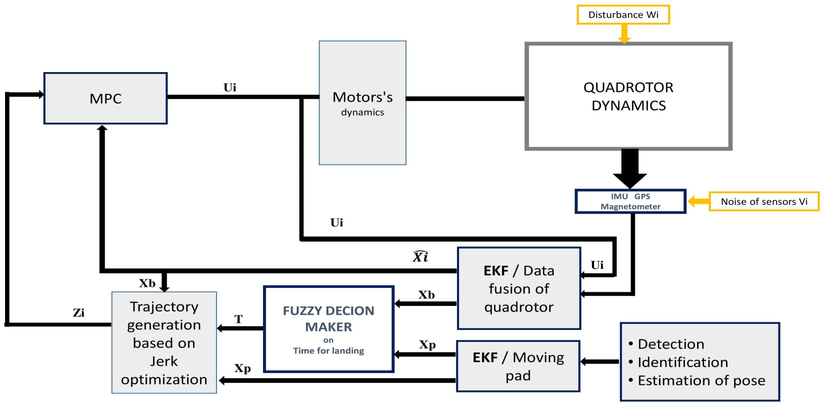


Figure 5-15: Complete Scheme of MPC control with Fuzzy min Jerk trajectory

be demonstrated to be [14]:

$$u_i = -\left[(R_i + B_i^T P_{i+1} B_i)^{-1} B_i^T P_{i+1} A_i\right] x_i + \left[(R_i + B_i^T P_{i+1} B_i)^{-1} B_i^T\right] g_{i+1} \tag{5.54}$$

P_i and G_i are recursively calculated as follows:

$$\begin{cases} P_i = V_i + A_i^T P_{i+1}\left(I + B_i R_i^{-1} B_i^T P_{i+1}\right)^{-1} A_i \\ g_i = W_i Z_i + A_i^T\left(I - P_{i+1}(I + B_i R_i^{-1} B_i^T P_{i+1})^{-1} B_i R_i^{-1} B_i^T\right) g_{i+1} \end{cases} \tag{5.55}$$

With $V_i = C_i^T Q_i C_i$ and $W_i = C_i^T Q_i$,

$$\begin{cases} P_N = C_N^T S_N C_N \\ g_N = C_N^T S_N Z_N \end{cases} \tag{5.56}$$

The control solution is calculated on the Horizon of N points to the landing on the pad, at every iteration the control is calculated repeatedly and applied just for the current iteration. The complete FLC/MPC control-based architecture is shown in Fig.5-2.

Stability of the control is ensured by the separation principle, essentially, Adaptive LQG is a combination of the KF and LQR. Thus, the design of optimal control can be derived independently of the observer.

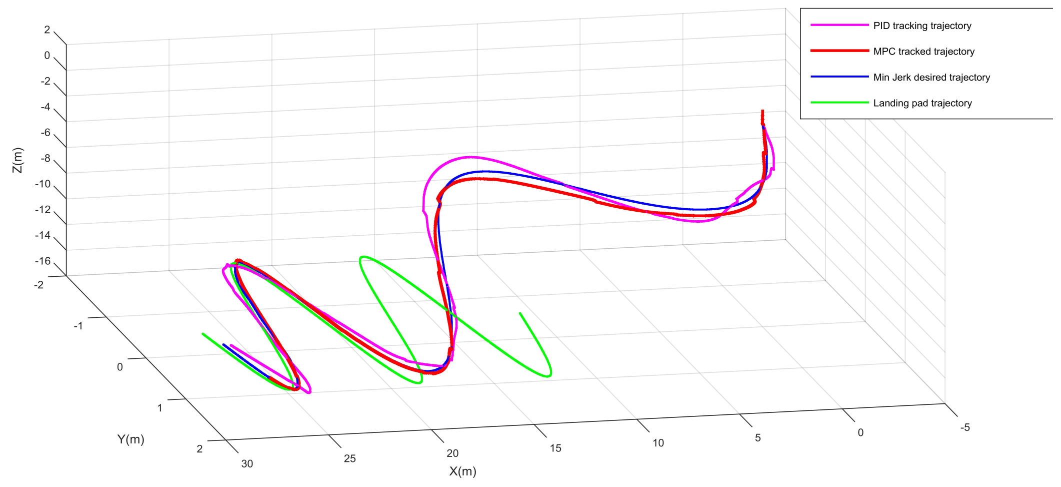


Figure 5-16: 3D Minimum Jerk trajectory tracking of a quadrotor performance controlled by PID vs MPC (Landing pad trajectory in Green, The optimal Min-Jerk trajectory in blue, the Quadrotor position controlled by MPC in red, and the magenta path is the quadrotor position by PID control).

The Control Algorithm of MPC can be designed as[6]:

Algorithm 2: MPC control Algorithm

```
Result: Control input $U_i$
Recall the last estimation of $x_i$, or initial conditions;
Verify initialization procedure;
while Control verification is Ok do
    Consider the appropriate $A_i$, $B_i$ and $C_i$ from best estimate of EKF and sensor
      available data;
    Calculate $V_i$ and $W_i$.;
    N=100 (number of points can be adaptive );
    Calculate $P_N$ and $g_N$ thru Eq.5.56 ;
    for $k$=$N$:$i$ do
        Calculate $P_k$ and $g_k$ with Eq.5.55 ;
    end
    Calculate $u_i$ by $P_i$ and $g_i$ as in Eq.5.54 ;
    Apply control ;
end
```

[6]GitHub codes-Soft Landing simulation.

5.4.3 Control simulation results

MPC tracking of the Min Jerk trajectory to land was simulated in a Matlab environment by following a landing pad in motion considering only the EKF as an observer. The moving pad position was simulated to be a forward sinusoidal motion, as shown in the green path in Fig.5-16, During the simulation, the Optimal trajectory is calculated dynamically and updated in every iteration, based on optimization dynamics in Eq.6.23, furthermore, the T value is estimated by the FLC. The MPC tracker calculates iteratively the optimal command to apply by the set of equations Eqs.5.54-5.56. The blue path shows the optimal calculated trajectory. The Red trajectory is the MPC simulated position of the quadrotor compared to the PID one in magenta. The simulation of MPC shows excellent stabilization and very good tracking of the desired trajectory even in the presence of simulated wind gusts and model uncertainties with Gaussian noise as demonstrated in Fig.5-16, PID control shows more deviation and less robustness to the sudden noise and wind perturbation. A comparison with PID control illustrates that the MPC is more performant and more robust to the added disturbance.

5.5 Radial Basis Functions Neural Network based compensation control

NN methods have become increasingly efficient and applicable in control. In this section, we investigate a novel nested control strategy based on Adaptive RBFNN and NN supervised control embedded with IBS for a robust trucking of quadrotors aerial robot in the presence of modeling uncertainties, sensing noise, and external bounded disturbance. The decentralized inverse dynamics were considered in the design, the outer loop is controlled by an adaptive RBFNN that approximates the unknown external disturbance and adaptively compensates for them, differently to the IBS supervised control that stabilizes the attitude in the inner loop that adaptively corrects and compensates the disturbance, in addition to avoiding initial instability of attitude during NN convergence. An adaptive EKF was exploited, and the stability analysis was elaborated using Lyapunov stability theory. Simulation of adaptive RBFNN control philosophy proved robustness and effectiveness compared to PID, IBS, and offline decentralized CNN algorithms as it generates the control law by guarantying a fast convergence of parameters, better external disturbance compensation, and noise attenuation.

5.5.1 Toward Adaptive RBF Neural Network-Based Control

Modern control tasks demand a high level of complexity from control theory. Various advanced systems and applications, such as avionics and navigation systems, flight controllers, adaptive trajectory trackers, autonomous robots, and high-energy processes, require sophisticated control approaches within distributed systems of supervision and control. Meeting these demands involves developing intricate control philosophies. Those control systems fall typically into a class of highly disturbed and uncertain dynamics control; due to mechanical fatigue, faults of components, environmental conditions like gust winds and abnormal atmospheric pressure, severe model non-linearities, and measurement uncertainties. In addition to the noise of sensors and unmodeled effects[103] [79] [27].

In recent times, a multitude of effective control methods have been introduced to address these challenges, each employing distinct philosophies and approaches, such as robust control [64] [12], adaptive control [113] [55], and Neural network-based control [49] [59] [35] [112]. Incorporating an in-flight full-state observer in the presence of uncertainties and noise may establish a reliable feedback to the control block [23] [90] [22]. [104]. For the challenges of the disturbed trajectory tracking for quadrotors, Linear techniques are insufficient to guarantee convergence when functioning outside the operating envelope, while nonlinear techniques require a full knowledge of dynamics, which is not always guaranteed to meet as an assumption, which prone them to instability due to unmodeled effects and external forces. Furthermore, both approaches have limitations in their capacity to adapt to uncertainties. Thus, the need for intelligent methods, which demonstrate higher performance in the matter of all recited challenges. Adaptive Neural Network (NN) based control design based on dynamic inversion was intensively investigated for quadrotors, the optimal compensation of inversion error subsequently improves robustness to parametric uncertainty and non-linear unmodeled dynamics [88]. The Adaptive Neural Network (NN) efficiency is due to the capabilities of nonlinear approximation and adaptive adjustment.

The universal Approximation Theorem for RBFNN proved that the Gaussian functions network can approximate any bounded integrable, continuous function [88], especially in presence of discontinuities and dynamics change in time, which drive harm to the robust control methods [60] [111]. This section presents an adaptive control scheme for robust tracking of trajectory for the quadrotor, with disturbance rejection, noise attenuation, and dynamics uncertainty adaptation. Adaptive RBFNNs control was introduced for position tracking in [108], an Adaptive RBFNNs/integral sliding mode control for a quadrotor aircraft was opted in [59], in [49] a Robust adaptive RBFNN based compensation control of quadrotor was simulated. The adaptive technique can adjust parameters automatically based on dynamic changes for an agriculture drone [112]. where in [35], an Adaptive NN fault-tolerant control was investigated via fast terminal sliding mode. However, those researchers did not consider either the initial attitude instability during the network parameters convergence or optimal feedback due to the noisy disturbed nature of measurement.

It is essential to understand the quadrotor strong non-linear coupling of the four-rotors in the aircraft dynamics; This understanding paves the way for the development of a valuable decentralized control scheme that can be achieved by designing two nested subsystems. An outer loop controls the translational positions tracking, meanwhile, the inner loop stabilizes and tracks the desired attitude generated by the inverse compensated dynamics.

Consequently, an Adaptive RBFNN control scheme was opted for the outer loop of a disturbed quadrotor aircraft to achieve robust trajectory tracking objectives of the aircraft. The RBFNN is trained online via an adaptive law derived based on Lyapunov theory, offering an adaptation to any change that may occur during flight [34]. Moreover, RBFNN supervised control of the Integrator Backstepping for the control of attitude was designed, guarantying a fast convergence, and better initial attitude stability, contrary to other papers where initial severe swings have been observed during network convergence [45] [49], it is mandatory to establish a reasonable initial oscillation of attitude otherwise, the quadrotor may fall in a non-recoverable unstable envelope, furthermore, the integral action contributes to the cancellation of the static error. The stability analysis of each control loop combined with optimal observer was examined. The RBFNNs usually train much faster than back-propagation networks, which makes them a better solution than Convolution Neural Network (CNN)s[7] for the adaptive control of unknown or uncertain models.

In addition, the model uncertainties, noise, and disturbances are considered and formulated as unknown state dependent and independent inputs or Gaussian signals. The control technique for the translational position considers the known part of the model and adaptively

[7]GitHub codes - CNN control.

estimates the unknown part with the disturbance based on combining the RBF neural network approximation capabilities with the adaptive control technique. In consequence, this design of a robust tracking controller of quadrotor aircraft positions ensures the rejection of the external bounded disturbance. As well, the strategy guarantees a performant attitude subsystem tracking and stabilization with the supervised NN control.

Besides the implementation of the adaptive EKF to mitigate sensing noise and the destabilizing effect, which its stability analysis is demonstrated by Lyapunov theorem and separation principle while integrating it in the control nested loops . This section highlights the following keys:

1. Conception of a nested NN design based on adaptive RBFNN to control position in the outer loop and a supervised NN to control the attitude that carries out superior performance such as a faster convergence and capable of disturbance rejection and stabilize attitude at the initial phase, with higher precision in comparison to some nonlinear and intelligent control approaches which are limited to robustness against the assessed type of noise and perturbation, as demonstrated in Fig.5-2.

2. Comparison of different algorithms of control with the opted RBF strategy relative to the tracking error and robustness to disturbance and unmodeled effects. a PID, an IBS based on nonlinear dynamics, and a decentralized offline trained Multi Layer Perceptron Neural Network (MLPNN) have been evaluated in comparison. In Matlab simulation, motors dynamics and mechanical noise have been all added to the model.

5.5.2 RBF Network and Compensation control

Our goal is to develop an adaptive control system that effectively addresses the impact of bounded disturbances and internal uncertainties caused by various factors such as modeling errors, wear and tear, mechanical fatigue, estimation uncertainties, and environmental effects like wind gusts and drag variations. To achieve this, we construct a disturbance functional vector that encompasses all noise and perturbations, enabling us to effectively compensate for these disruptive dynamics.

In [13], turbulence on the quadrotor was studied in unstructured dynamics with consideration of the aerodynamic model as a disturbance on the quadrotor. Nevertheless, in [49], the disturbance was considered in structured dynamics, in this this section, disturbance and uncertainties are considered as unknown unmodeled bounded functions.

The RBF network adaptation can ultimately ensure the control performance for uncertain systems models [46] [24] [2]. Essentially, the adaptation law is derived using the Lyapunov theory so that the stability of the control loop is guaranteed with fast convergence of adaptation weights.

The philosophy of the RBFNN adaptive control laws are mainly derived from Lyapunov theory for either the adaptive compensation control in the linear position, or Attitude Supervisory control law embedded with IBS Control.

RBFNNs are widely addressed because of their nonlinear universal approximation for any nonlinear function over a compact set with arbitrary accuracy by optimizing a performance index, and because of the simple network structure that converges faster and minimizes the extensive calculations compared to conventional Multi Layer Perceptron (MLP)s. A Nonlinear function $f(x)$ can be developed as:

$$f(x) = w^T h(x) + \epsilon \tag{5.57}$$

Where w is the weights matrix, h is a vector of functions, and ϵ is the error of approximation. To use RBFNN to approximate the function f in control systems, it is fitter either to choose the system states as the inputs of the neural network, or the tracking error and its derivative as the input vector. An approximation of f, noted as $\hat{f}$, can be written as follows:

$$\hat{f}(x) = \hat{w}^{*T} h(x) \tag{5.58}$$

Where $\hat{w}^*$ is the best estimate of the weights matrix. The outputs of MIMO adaptive RBFNN can be illustrated as:

$$y_i(t) = \sum_{j=1}^{m} w_{ji} h_j(t), i = 1, ..., n \tag{5.59}$$

With,

$$h_j(t) = \exp(-\frac{|x(t) - c_j(t)|^2}{2 * b_j^2}), m = 1, ..., m \tag{5.60}$$

Where, $x = [x_i]^T$ is the input vector; $|x(t) - c_j(t)|$ is the Euclidean distance between the center c_j and the network inputs, b_j notes a positive scalar called a width. $h_j(t)$ is the nonlinear output of m hidden nodes in the hidden layer.

NOTE:

It is critical to bear in mind that Designing the RBFNN for approximation based on the gradient descent method to adjust weights, can guarantee only local optimization, and not closed-loop system stability, therefore, the online adaptive RBFNN compensation control method is better designed based on the Lyapunov stability theory, thus, the stability of the closed-loop system can be achieved in the separation of the observer.

5.5.3 RBFNN Position Adaptive compensation Control Based on disturbance Approximation

In practice, the perfect quadrotor model could be hard to obtain, as external disturbances are always present, let's Consider the translational positions dynamics model of the quadrotor as mentioned in system of Eqs.5-2 in the presence of disturbance, we can write:

$$\begin{cases} \ddot{x} = (s\psi s\phi + c\psi s\theta c\phi)\ U_1/m + d_x \\ \ddot{y} = (-c\psi s\phi + s\psi s\theta c\phi)\ U_1/m + d_y \\ \ddot{z} = c\theta c\phi\ U_1/m - g + d_z \end{cases} \tag{5.61}$$

Where, $d = [d_x(t), d_y(t), d_z(t)]^T$ is the vector that represents the dynamics of disturbance and uncertainties.

Assumption 1. $d = [d_x, d_y, d_z]^T$ are continuously differentiable bounded functions. For the nominal model, We use:

$$\begin{cases} u_x = (s\psi s\phi + c\psi s\theta c\phi)\ U_1/m \\ u_y = (-c\psi s\phi + s\psi s\theta c\phi)\ U_1/m \\ u_z = c\theta c\phi\ U_1/m - g \end{cases} \tag{5.62}$$

However, d is an unknown nonlinear vector of functions. g, u_x, u_z, and u_y are known nonlinear functions from the nominal model.

Assuming that the ideal continuous bounded translational position of the quadrotor is

$[x_d, y_d, z_d]^T$, we define a real time differentiable error.

$$e = \begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} - \begin{bmatrix} x_d \\ y_d \\ z_d \end{bmatrix} \tag{5.63}$$

The control maybe designed as follows:

$$\begin{cases} u_x = (\ddot{x}_d - k_{vx}\dot{e}_x - k_{px}e_x) - d_x \\ u_y = (\ddot{y}_d - k_{vy}\dot{e}_y - k_{py}e_x) - d_y \\ u_z = (\ddot{z}_d - k_{vz}\dot{e}_z - k_{pz}e_z) - d_z \end{cases} \tag{5.64}$$

Where, $k_{vx}, k_{vy}, k_{vz}, k_{px}, k_{py}, k_{pz}$ are well chosen positive parameters, The closed loop systems in Eq.5.64 will follow the error dynamics e such as:

$$\ddot{e} + k_v\dot{e} + k_p e = 0 \tag{5.65}$$

Using three RBFNNs to design an approximation $\hat{d}$ to the vector d with bounded approximation error ϵ.

$$d = \begin{bmatrix} w_x^{*T}h_x + \epsilon_x \\ w_y^{*T}h_y + \epsilon_y \\ w_z^{*T}h_z + \epsilon_z \end{bmatrix} \tag{5.66}$$

w_i^* such $i \in \{x, y, z\}$ denotes the optimal weights estimation. For every disturbance component d_i of the vector d, we use inputs from the respective error $[e_i, \dot{e}_i]$; d_x can be written as:

$$\hat{d}_x = \hat{w}_x^T h_x \tag{5.67}$$

Where $\hat{w}_x$ is the estimated weight matrix of w_x , By analogy we develop the $\hat{d}_y$ and $\hat{d}_z$. Fig. 5-17 illustrates the RBF neural-based closed-loop adaptive control scheme.

As $d = [d_x, d_y, d_z]^T$ is unknown, the RBF design should approximate it by best estimation $\hat{d}^*$, and compensate it. Taking ϵ as a very small positive constant, we may converge to w^* such as:

$$max \parallel d - \hat{d}^* \parallel \leq \epsilon \tag{5.68}$$

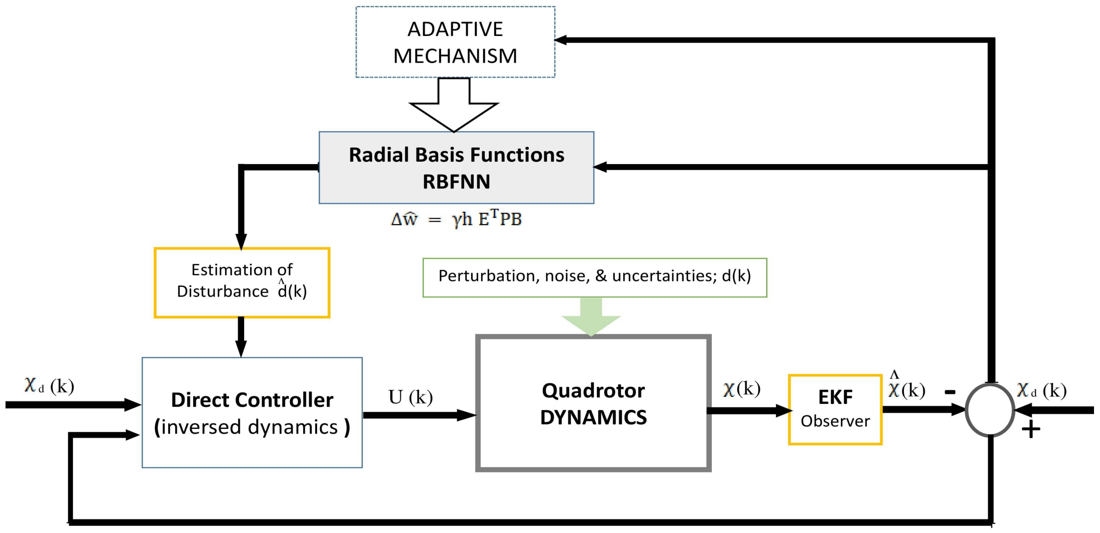

ADAPTIVE MECHANISM
Radial Basis Functions RBFNN
$\Delta\hat{w} = \gamma h\, E^T PB$
Estimation of Disturbance $\hat{d}$(k)
Perturbation, noise, & uncertainties; d(k)
χ_d (k)
Direct Controller (inversed dynamics)
U (k)
Quadrotor DYNAMICS
χ(k)
EKF Observer
$\hat{\chi}$(k)
-
+
χ_d (k)

Figure 5-17: Control strategy to adaptively Compensate the disturbance based on RBFNN

Define η and η_0 as the approximation error and its bounded value:

$$\eta = d - \hat{d}^* \quad ; \quad \eta_0 = sup \parallel d - \hat{d}^* \parallel \tag{5.69}$$

Injecting the disturbance approximation $\hat{d}$ into Eq.5.64, the closed loop dynamics of error gives:

$$\ddot{e} + k_v \dot{e} + k_p e = d - \hat{d}^* \tag{5.70}$$

Where, $k_v = diag[k_{vx}, k_{vy}, k_{vz}]$ and, $k_p = diag[k_{px}, k_{py}, k_{pz}]$ both positive definite. By taking, $E = [e_x, \dot{e}_x, e_y, \dot{e}_y, e_z, \dot{e}_z]^T$, Eq.5.70 can be represented in state space as follows:

$$\dot{E} = AE + B(d - \hat{d}^*) \tag{5.71}$$

where,

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ -k_{px} & -k_{vx} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -k_{py} & -k_{vy} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -k_{pz} & -k_{vz} \end{bmatrix} ; \; B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.72}$$

$$d - \hat{d} = d - \hat{d}^* + \hat{d}^* - \hat{d} = \eta - \tilde{w}^T h \tag{5.73}$$

where, $\tilde{w} = \hat{w} - w^*$

$$\dot{E} = AE + B(\eta - \tilde{w}^T h) \tag{5.74}$$

The Eq.5.74 illustrates a closed loop dynamics for the error.

5.5.4 Stability Analysis of the RBFNN compensation control loops

For the stability of the linear position error dynamics in Eq.5.74, if we design an adaptive compensation control to approximate the disturbance, the bounded trajectory tracking of the position in outer loop should satisfy: $lim_{t\to+\infty}|E(t)| \leq \epsilon$

Choosing a Lyapunov candidate function as:

$$V = \frac{1}{2}E^T PE + \frac{1}{2\gamma} \parallel \tilde{w} \parallel^2 \tag{5.75}$$

With $\gamma > 0, Q > 0$, and The matrix P is symmetric, positive definite, and satisfies the following Lyapunov equation.

$$A^T P + PA = -Q \tag{5.76}$$

By defining,

$$\parallel R \parallel^2 = \sum_{i,j} |r_{ij}|^2 = tr(RR^T) = tr(R^T R) \tag{5.77}$$

Where $tr(.)$ is the trace of matrix.

$$\parallel \tilde{w} \parallel^2 = tr(\tilde{w}^T \tilde{w}) \tag{5.78}$$

By the derivation of V is:

$$\dot{V} = \frac{1}{2}(E^T P \dot{E} + \dot{E}^T P E) + \frac{1}{\gamma} tr(\dot{\tilde{w}}^T \tilde{w}) \tag{5.79}$$

$$\dot{V} = \frac{1}{2}(E^T P(AE + B(\eta - \tilde{w}^T h)) + (E^T A^T + B^T(\eta - \tilde{w}^T h)^T)PE) + \frac{1}{\gamma} tr(\dot{\tilde{w}}^T \tilde{w}) \tag{5.80}$$

$$\dot{V} = \frac{1}{2}(E^T(PA + A^T P)E + (E^T PB\eta - E^T PB\tilde{w}^T h + \eta^T B^T PE - h^T \tilde{w} B^T PE) + \frac{1}{\gamma} tr(\dot{\tilde{w}}^T \tilde{w}) \tag{5.81}$$

$$\dot{V} = -\frac{1}{2} E^T Q E + \eta^T B^T PE - h^T \tilde{w} B^T PE) + \frac{1}{\gamma} tr(\dot{\tilde{w}}^T \tilde{w}) \tag{5.82}$$

We have,

$$E^T PB\tilde{w}^T h = h^T \tilde{w} B^T PE \tag{5.83}$$

$$E^T PB\eta = \eta^T B^T PE \tag{5.84}$$

Due to the diagonal nature of P, B, and E So we may have;

$$h^T \tilde{w} B^T PE = tr(B^T PEh^T \tilde{w}) \tag{5.85}$$

Then,

$$\dot{V} = -\frac{1}{2} E^T Q E + \frac{1}{\gamma} tr(-\gamma B^T PEh^T \tilde{w} + \dot{\tilde{w}}^T \tilde{w}) + \eta^T B^T PE \tag{5.86}$$

Designing an adaptive law as:

$$\dot{\hat{w}}^T = \gamma B^T PEh^T \tag{5.87}$$

So,

$$\dot{\hat{w}} = \gamma h E^T PB \tag{5.88}$$

Since $\dot{\hat{w}} = \dot{\tilde{w}}$; Injecting Eq.5.88 into Eq.5.86

$$\dot{V} = -\frac{1}{2}E^T Q E + \eta^T B^T P E \tag{5.89}$$

We know that,

$$\parallel \eta^T \parallel \leq \parallel \eta_0 \parallel \quad ; \parallel B \parallel = \beta \tag{5.90}$$

$$\dot{V} \leq -\frac{1}{2}\lambda_{min}(Q) \parallel E \parallel^2 + \parallel \eta_0 \parallel \beta\lambda_{max}(P) \parallel E \parallel \tag{5.91}$$

$$\dot{V} \leq -\frac{1}{2} \parallel E \parallel \Big(\lambda_{min}(Q) \parallel E \parallel -2 \parallel \eta_0 \parallel \beta\lambda_{max}(P)\Big) \tag{5.92}$$

$\lambda_{max}(P)$, $\lambda_{min}(Q)$ denote the maximum eigenvalue of matrix P and the minimum eigenvalue of matrix Q respectively. to Satisfy the condition of stability, $\dot{V} \leq 0$ we should have:

$$\lambda_{min}(Q) \geq 2\frac{\beta\lambda_{max}(P)}{\parallel E \parallel} \parallel \eta_0 \parallel \tag{5.93}$$

We may observe that Eq.5.94 shows that the radius of convergence of the error vector E is smaller as the max eigenvalue of Q is bigger, or the min value of the eigenvalue of P is smaller, or by a smaller value of η_0.

$$\parallel E \parallel = 2\frac{\beta\lambda_{max}(P)}{\lambda_{min}(Q)} \parallel \eta_0 \parallel \tag{5.94}$$

5.5.5 Stability Analysis of the adaptive RBFNN control with the optimal observer

The adaptive RBFNN compensation control is combined with an adaptive Extended Kalman Filter, the stability analysis of the linearized models can rely on the separation principle of states, however, this will ensure only local stability of the linearized system in the closed loop. Therefore, we opted for signal dynamics stability analysis to justify the fact of designing the control and the observer independently.

proof The EKF as detailed in Subsection.5.21 is based on the Jacobian linearization at every estimated state, estimated states and outputs of the optimal filter are used to feed back the direct controller (Inverse dynamics controller) and the RBF network as detailed in

Fig.5-17. Considering the closed-loop dynamics of the translational positions in Eq.5.70:

$$\ddot{e} + k_v\dot{e} + k_p e = d - \hat{d}^* \tag{5.95}$$

Taking $\chi = [x, y, z]^T$ as translation coordinates, $\chi_d = [x_d, y_d, z_d]^T$ as the desired ones, and $\hat{\chi} = [\hat{x}, \hat{y}, \hat{z}]^T$ as partial estimation output of the Optimal observer:

$$e = \chi - \hat{\chi} + \hat{\chi} - \chi_d = \tilde{\chi} + \hat{e} \tag{5.96}$$

$$(\ddot{\chi} - \ddot{\hat{\chi}}) + (\ddot{\hat{\chi}} - \ddot{\chi}_d) + k_v(\dot{\chi} - \dot{\hat{\chi}}) + k_v(\dot{\hat{\chi}} - \dot{\chi}_d) + k_p(\chi - \hat{\chi}) + k_p(\hat{\chi} - \chi_d) = d - \hat{d}^* \tag{5.97}$$

$$\ddot{\hat{e}} + k_v\dot{\hat{e}} + k_p\hat{e} + \ddot{\tilde{\chi}} + k_v\dot{\tilde{\chi}} + +k_p\tilde{\chi} = d - \hat{d}^* \tag{5.98}$$

From Eqt.5.21 an optimal observer error's dynamics can be designed as follows:

$$\dot{\tilde{\chi}} = (A_i - K_iC_i)\tilde{\chi} + v(i) + K_i n(i) \tag{5.99}$$

Where $v(i)$ and $n(i)$ are orthogonal process noise and measurement noise respectively, and K_i is the optimal gain matrix that ensures that eigenvalues of $A_i - K_iC_i$ in the left half of the plane if (A_i, C_i) is observable. Therefore, the dynamics of the error of the observer dictated by $\ddot{\tilde{\chi}} + k_v\dot{\tilde{\chi}} + k_p\tilde{\chi}$ are exponentially stable and converging to a null vector. By taking $\tilde{\chi}_1 = \tilde{\chi}$, and $\tilde{\chi}_2 = \dot{\tilde{\chi}}$, we can develop:

$$\begin{bmatrix} \dot{\tilde{\chi}}_1 \\ \dot{\tilde{\chi}}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k_p & -k_v \end{bmatrix} \begin{bmatrix} \tilde{\chi}_1 \\ \tilde{\chi}_2 \end{bmatrix} \tag{5.100}$$

The system in Eq.5.100, shows stable dynamics for the estimation error $\tilde{\chi}$.

In addition, the dynamics dictated by $\ddot{\hat{e}} + k_v\dot{\hat{e}} + k_p\hat{e} = d - \hat{d}^*$ have been proven stable by Lyapunov candidate function in Eq.5.75, and the weight update mentionned in Eq. 5.88. Therefore, the combination between the EKF and the RBFNN compensation control presents stable dynamics. Furthermore, the dynamics of the observer are independent of the dynamics of the controller, which makes it possible to design them separately.

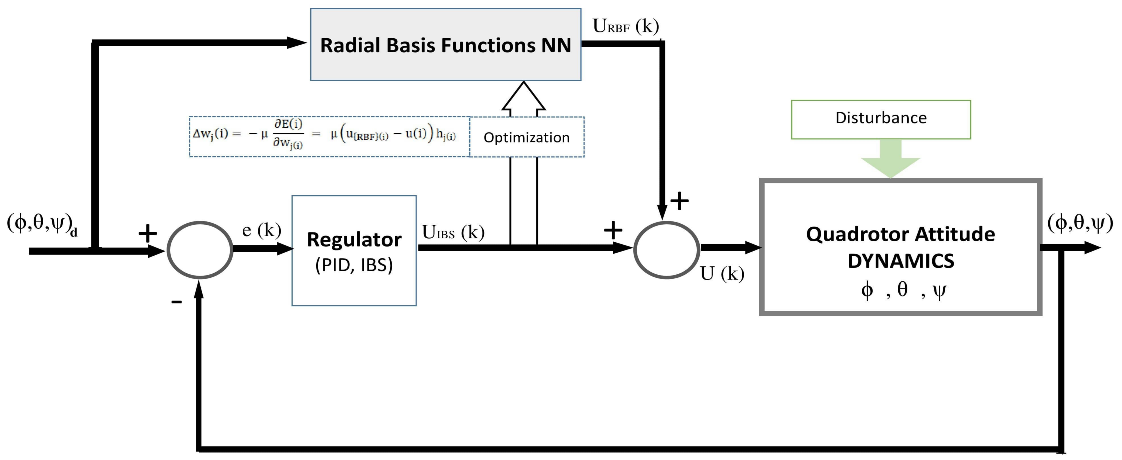

Radial Basis Functions NN
U_{RBF} (k)
$\Delta w_j(i) = -\mu \frac{\partial E(i)}{\partial w_{j(i)}} = \mu\left(u_{\{RBF\}(i)} - u(i)\right) h_{j(i)}$
Optimization
Disturbance
$(\phi,\theta,\psi)_d$
+
-
e (k)
Regulator
(PID, IBS)
U_{IBS} (k)
+
+
U (k)
Quadrotor Attitude
DYNAMICS
ϕ , θ , ψ
(ϕ,θ,ψ)

Figure 5-18: Scheme of the Attitude RBFNN supervisory Control

5.5.6 Attitude RBFNN supervisory Control

In this subsection, a supervisory attitude control based on RBFNN embedded with IBS is proposed. A supervised control has been designed to stabilize the attitude angles in the rotational inner loop in the nested design to avoid the initial instability that may occur during the parametric convergence NN.

Contrary to the adaptive control opted for the translational positions, the attitude angles present a strong coupling, thus an interlinked network scheme was designed, and the yaw angle rate $\dot{\psi}$ was considered in the input mapping as it affects directly both θ and ϕ. The mapping of $\dot{\psi}$ improves the fast convergence of the network parameters. An Integral action has been added in the embedded controller to eliminate the static error. The control scheme is given in Fig.5-18.

The supervised RBFNN is combined with a nonlinear Integrator Back Stepping (IBS) that contributes the integral benefits in the backstepping design, the attitude design of Control is based on three IBS controllers, and each one is optimized by its RBF network. The derivation of those IBS controllers is similar for all angles; thus, only one control will be derived in this paper.

Attitude Control

Considering the following subsystem for the roll angle ϕ :

$$\begin{cases} \dot{x}_7 = x_8 \\ \dot{x}_8 = 1/I_{xx}\ ((I_{yy} - I_{zz})\ x_{12}x_{10} + J_{TP}x_{10}\Omega + U_2) \end{cases} \tag{5.101}$$

Let's consider the error between the desired and actual roll angle:

$$e_1 = \phi_d - \phi = x_{7d} - x_7 \tag{5.102}$$

Considering the Lyapunov candidate function, $V_1 = \frac{1}{2}e_1^2$, Deriving V_1 as follows:

$$\dot{V}_1 = e_1\dot{e}_1 = e_1(\dot{x}_{7d} - x_8) \tag{5.103}$$

Choosing, $x_8 = \dot{x}_{7d} + k_1e_1$, with $k_1 > 0$ will ensure that $\dot{V}_1$ is a negative definite function, and the desired trajectory of x_8 is defined by $x_{8d} = \dot{x}_{7d} + k_1e_1$.

Denoting e_2 the error between the actual and desired roll angle rates.

$$e_2 = \dot{\phi}_d - \dot{\phi} = x_{8d} - x_8 \tag{5.104}$$

defining V_2 as a positive Lyapunov candidate function, $V_2 = V_1 + \frac{1}{2}e_2^2$, deriving V_2:

$$\dot{V}_2 = \dot{V}_1 + e_2\dot{e}_2 = \dot{V}_1 + e_2(\dot{x}_{8d} - \dot{x}_8) \tag{5.105}$$

$$\dot{V}_2 = \dot{V}_1 + e_2(\dot{x}_{8d} - 1/I_{xx}\ ((I_{yy} - I_{zz})\ x_{12}x_{10} + J_{TP}x_{10}\Omega + U_2)) \tag{5.106}$$

However, U_2 in Eq.5.106, is the sum of nonlinear backstepping control and the RBFNN output.

$$U_2 = U_{RBF} + U_{IBS} \tag{5.107}$$

Where, U_{IBS} is the Backstepping control of nominal model, and $U_{RBF} = W^T h(x_7)$, Injecting U_2 into Eq.5.106.

$$\dot{V}_2 = \dot{V}_1 + e_2(\dot{x}_{8d} - 1/I_{xx}\ ((I_{yy} - I_{zz})\ x_{12}x_{10} + J_{TP}x_{10}\Omega + U_{RBF} + U_{IBS})) \tag{5.108}$$

By taking $U_{IBS} = k_2 e_2 I_{xx} - I_{xx}\dot{x}_{8d} - ((I_{yy} - I_{zz})\ x_{12}x_{10} - J_{TP}x_{10}\Omega)$, with $k_2 > 0$, we find:

$$\dot{V}_2 = \dot{V}_1 + e_2(-k_2 e_2 + U_{RBF}) \tag{5.109}$$

Assuming that the RBFNN of approximation of the control loop is a bounded function with bounded weights.

$$U_{max} = sup|U_{RBF}|. \tag{5.110}$$

Analysing Eq.5.109: If $(U_{RBF}\ e_2) < 0$, $\dot{V}_2$ will be negative definite, otherwise:

$$\dot{V}_2 < \dot{V}_1 + e_2(-k_2 e_2 + U_{max} sign(e_2)) \tag{5.111}$$

An appropriate choice of k_2 with consideration to $\dot{V}_1$ will ensure that $\dot{V}_2$ is negative. Fig.5-19 demonstrates the negative sign of $\dot{V}_2$ with an appropriate k_2 and a bounded error e_2. Thus the stability of the closed loop of the design in assumption of bounded RBF approximation input.

$$U_{IBS,\phi} = k_2 e_2 I_{xx} - I_{xx}(\dot{x}_{7d} - k_1 e_1) - ((I_{yy} - I_{zz})\ x_{12}x_{10} - J_{TP}x_{10}\Omega) \tag{5.112}$$

It is well known that adding $\alpha_\phi \int_0^t e_1(\tau)d\tau$ with $\alpha_\phi > 0$ to the control law $U_{IBS,\phi}$ will help to cancel the steady error.

Considering, $e_3 = x_{9d} + x_9$ and $e_5 = x_{11d} + x_11$. Systematically. The control inputs derived for θ and ψ errors are given by:

$$U_{IBS,\theta} = k_4 e_4 I_{yy} - I_{yy}(\dot{x}_{9d} - k_3 e_3) - ((I_{zz} - I_{xx})\ x_{12}x_8 - J_{TP}x_8\Omega) + \alpha_\theta \int_0^t e_3(\tau)d\tau \tag{5.113}$$

$$U_{IBS,\psi} = k_6 e_6 I_{zz} - I_{zz}(\dot{x}_{11d} - k_5 e_5) - ((I_{xx} - I_{yy})\ x_{10}x_8) + \alpha_\psi \int_0^t e_5(\tau)d\tau \tag{5.114}$$

With $(k_3, k_4, k_5, k_6, \alpha_\theta, \alpha_\psi) > 0$

The desired trajectory contains only information about the desired linear position $[x_d, y_d, z_d]$

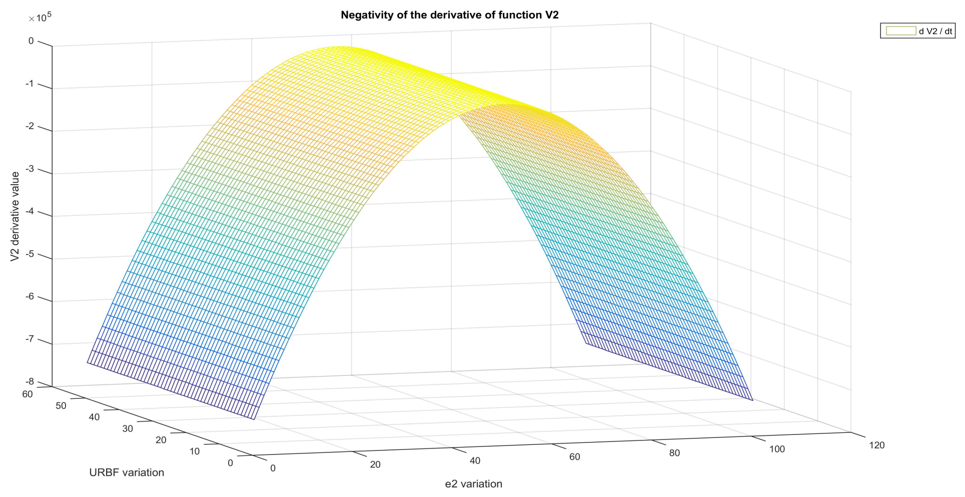


Figure 5-19: 3D plot of the derivative of V_2 in function of e_2 and U_{RBF}

and the yaw angle ψ_d. the desired attitude trajectory is derived from the virtual control inputs of the inner attitude loop, Using Eq.5.62 we can analytically develop the following:

$$\begin{aligned} \phi_d(t) &= \arcsin\big(\big(\sin\psi(t)u_x - \cos\psi_d(t))\big)\big)/(u_x^2(t)+u_y^2(t))\big) \\ \theta_d(t) &= \arcsin\big(\big(u_x(t) - \sin\psi_d(t)\sin\phi_d(t)\big)\big)/(\cos(\psi_d(t))\cos(\phi_d(t))\big) \end{aligned} \tag{5.115}$$

The RBF network uses the radial basis matrix $[h_\theta, h_\phi, h_\psi]$ defined by Gaussian functions as mentioned in Eq.5.60. Three control outputs from the adaptive network stabilize the quadrotor and compensate for the unmodeled uncertainty and disturbance.

The criterion of optimization for each angle control can be defined as follows:

$$E(i) = \frac{1}{2}(u_{RBF}(i)^2 - u(i)^2) \tag{5.116}$$

u_{RBF} is the network control output, and $u(i)$ is the quadrotor control inputs. A steeped descent design can contribute to a fast convergence:

$$\Delta w_j(i) = -\mu\frac{\partial E(i)}{\partial w_j(i)} = \mu(u_{RBF}(i) - u(i))h_j(i) \tag{5.117}$$

Using the learning rate μ and the momentum weight ξ that works the same as damper in PID control,

$$w_j(i) = w_j(i-1) + \mu(u_{RBF}(i) - u(i))h_j + \xi(w_j(i-1) - w_j(i-2)) \tag{5.118}$$

By an analogy of variation we find:

$$b_j(i) = b_j(i-1) + \mu(u_{RBF}(i) - u(i))w_j h_j \frac{|x(i) - c_j(i)|^3}{b_j^3} + \xi(b_j(i-1) - b_j(i-2)) \quad (5.119)$$

$$C_{ji}(i) = C_{ji}(i-1) + \mu(u_{RBF}(i) - u(i))w_j \frac{x_j(i) - c_{ji}(i)}{2b_j} + \xi(C_{ji}(i-1) - C_{ji}(i-2)) \quad (5.120)$$

$\mu \in]0.1[$ is the learning rate and $\xi \in]0.1[$is the momentum factor. It is important to choose initially appropriate values for c_j and b_j to cover all space of inputs validating the fast valid mapping of the model.

Parameters equations are iterative, and once the convergence is achieved, the quadrotor attitude dynamics are controlled for tracking by the RBFNN.

5.5.7 RBFNN Simulation results and interpretations

Several simulations had been carried out in this section to demonstrate the effectiveness and performance of the proposed control strategy for robust trajectory tracking challenges. The path has been generated arbitrarily as a helical trajectory, the table.5.3 gives the used quadrotor parameters during the simulation.

Table 5.3: The quadrotor parameters used for the simulation

Symbol	Value	Unit
m	1	kg
g	9.81	m/s^2
l	0.2	m
d	0.00002	$-$
I_{xx}, I_{yy}	0.008	kgm^2
I_{zz}	0.015	kgm^2

Besides the adaptive RBFNN-based control, other strategies have been simulated in parallel to validate the performance of the opted approach. The Linear PID, the non-linear IBS, and the intelligent MLP were all simulated to track the same trajectory in the sight of the added 6 DOF noise and perturbation to demonstrate the effect of disturbance and the ability of rejection and compensation. The PID design was based on the decentralization of the coupled Multiple Inputs Multiple Outputs (MIMO) dynamics to uncoupled Single Input Single Output (SISO) systems as studied in [73], IBS control was simulated using a similar nested looping strategy with four uncoupled controllers derived by the Lyapunov

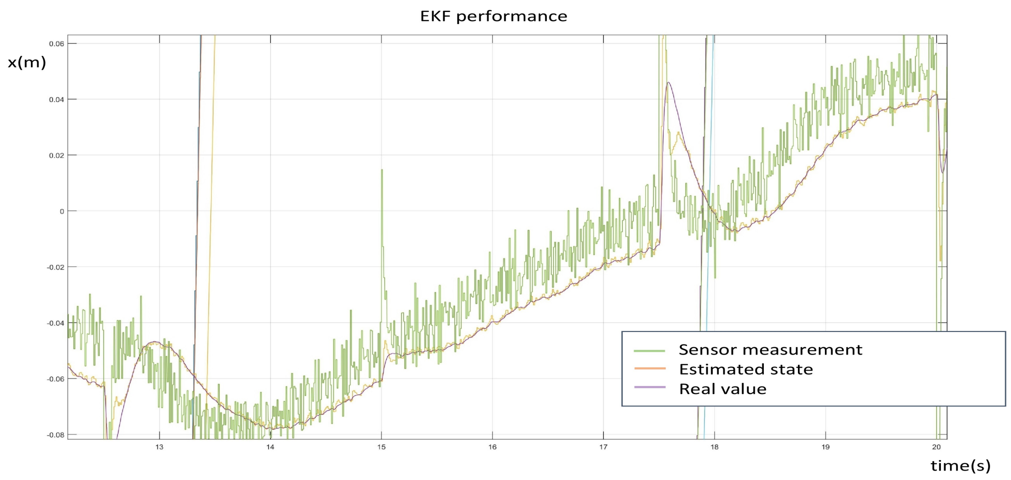


Figure 5-20: EKF results for one of the observed states

theorem to stabilize the altitude and the attitude. The MLP was based on model inversion decentralized controllers via an offline data training set for trajectory tracking.

Sensing noise and uncertainties were added to the simulation model as added Gaussian signals that affect the sensor's feedback and actuators, gust wind was simulated with aerodynamic drag by disturbing the translational positions by different rate Gaussian low-frequency signals. EKF presented a valid solution to ensure reliable feedback in the noisy environment, Fig.5-20 displays excellent stable observation of one of the quadrotor states.

The trajectory is defined in the R^3 Cartesian by:

$$\begin{aligned} x_d &= 5cos(\pi t/9)(m) \quad & y_d &= 5sin(\pi t/9)(m) \\ z_d &= t(m) \quad & &with\ static\ \psi_d\ (rad) \end{aligned} \tag{5.121}$$

Desired attitude angles ϕ_d and ψ_d are obtained utilizing Eq.5.115. The simulated initial conditions of the quadrotor are such as $X(:,0) = 0$ with $X \in R^{12}$.

The used initial parameters for control as mentioned in Eq.5.120 are given relative to the input signal mapping of the network extracted from the PID simulation, thus we used as initial parameters and weights for the Network of control of angle $Theta$ as one of the RBFNN blocks:

$$w_{theta} = ones(5,1);$$

$$b_{theta} = [0.6; 0.6; 0.6; 0.6; 0.6];$$

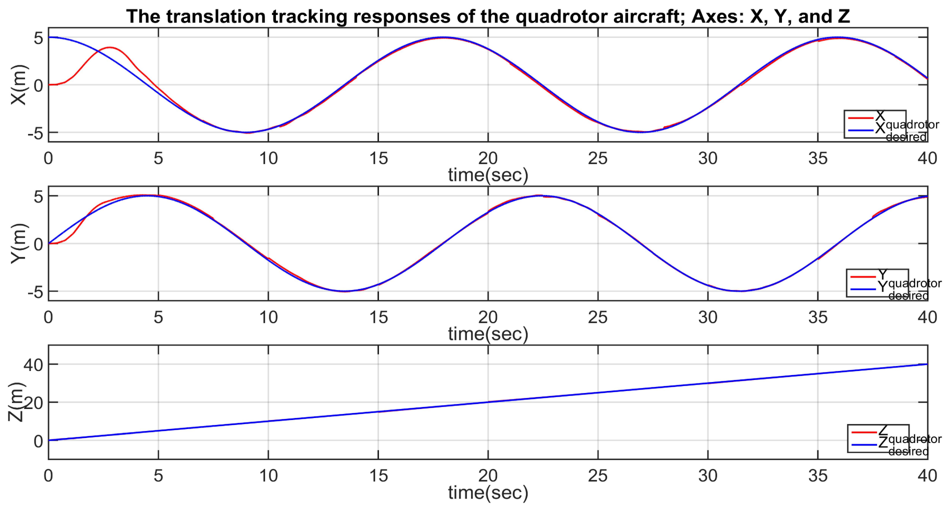


Figure 5-21: Quadrotor Position tracking of desired trajectory using the Proposed Adaptive RBFNN based compensation control

$$c_{theta} = [-1.2; -0.5; 0; 0.5; 1.2];$$

$$\xi = 1.79; \alpha = 0.7; \eta = 0.0005; \gamma = 0.0001$$

$$P_x = \begin{bmatrix} 8.0778 & 0.7000 \\ 0.7000 & 0.3556 \end{bmatrix}; \Lambda_x = \begin{bmatrix} 0 & 1 \\ -5 & -9 \end{bmatrix}; Q_x = \begin{bmatrix} 7 & 0 \\ 0 & 5 \end{bmatrix}$$

Response curves of the linear position with $[x, y, z]$ and angular position pitch, yaw, and roll angles $[\theta, \phi, \psi]$ of the quadrotor are illustrated in Fig.5-21 and Fig.5-22 respectively.

The simulation of the adaptive RBFNN control nested with the supervised NN attitude-based control demonstrates excellent stabilization and tracking of the desired trajectory with robustness and compensation of the disturbance as shown in Fig.5-21 for the translational positions and in Fig.5-22 for the attitude stabilization.

The Adaptive RBFNN strategy was validated for quadrotor aircraft trajectory tracking, and the rejection of perturbation, and the disturbance compensation were both assured with noise attenuation. In addition, it demonstrated better performance although the expensive calculation and algorithm complexity relative to other direct approaches.

The trajectory tracking errors of linear and angular positions are shown in Fig. 5-23. The initial position tracking error is due to the launch of the tracking from a different initial spot to examine the correction response. Angular errors were at the range of $[-0.02, 0.02]Rad$.

The disturbance approximation $d_i(t)$ that affects every translational dimension is il-

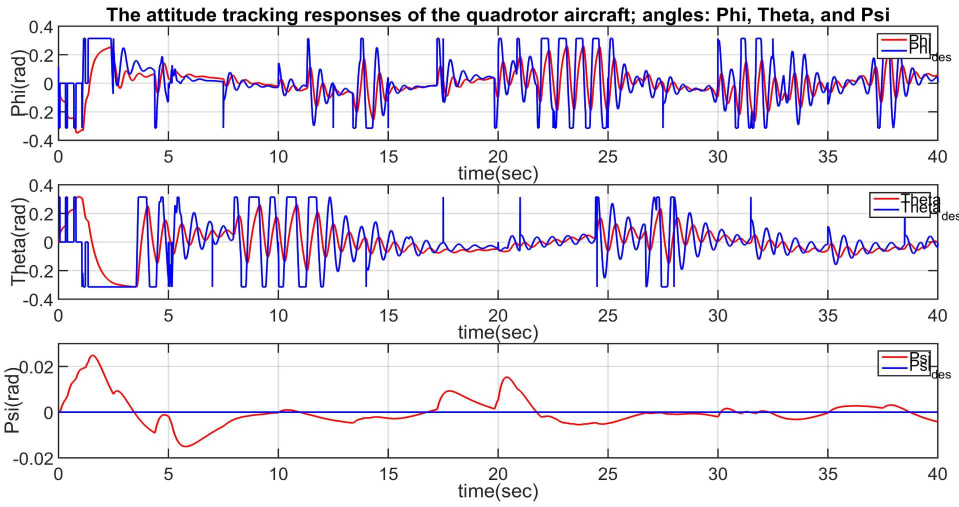


Figure 5-22: Quadrotor Attitude tracking using the Proposed Adaptive RBFNN supervisory control combined with Integrator Back stepping

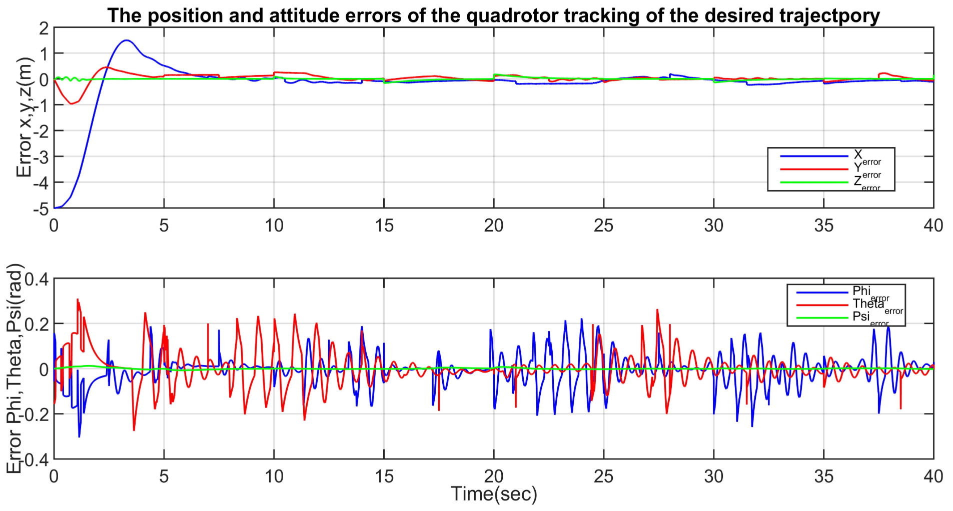


Figure 5-23: Linear and angular positions Tracking error

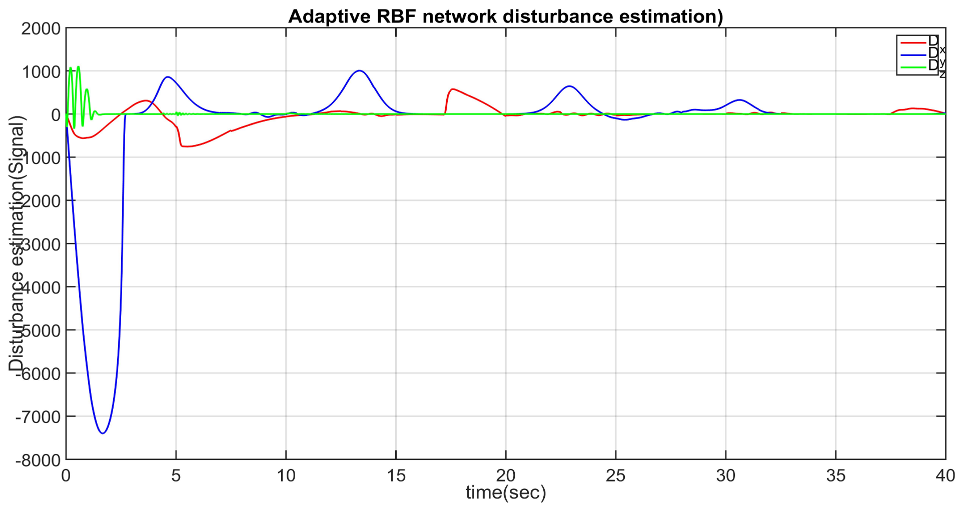


Figure 5-24: Disturbance vector estimation during simulation $\hat{d}(t)$

lustrated in Fig.5-24. the disturbance is considered an unknown function that has been estimated and compensated by the proposed controller. Fig.5-25, shows that the adaptive estimation networks outputs are bounded, with bounded small time-varying errors of estimation of disturbance, which justifies the possibility of adaptive compensation.

NN parameters convergence of a chosen hidden layer are shown in Fig.5-25. As a consequence, the proposed RBFNN-based adaptive scheme for disturbance compensation and robust tracking control has a strong potential to deal with uncertain models in a disturbed environment.

The upper view of linear position tracking comparison is demonstrated in Fig. 5-26, the wind gust effect by low-rate Gaussian noise is observable on the quadrotor position, contrary to the hidden attitude noise which is introduced to the angular feedback sensors by a high-rate Gaussian noise.

Comparison with other control strategies was conducted for validation, and The error of the tracking by the Adaptive RBFNN approach was minimal in the shadow of the tracking performances of other methods with the quickest corrective response to disturbance. PID, IBS, and MLP showed limited trajectory tracking performance in the presence of severe disturbance and noise. contrary to the proposed adaptive RBFNN strategy, as demonstrated in Fig.5-27.

The IBS control lost the trajectory tracking at the start and during the narrow curves, in addition, it showed a modest compensation of perturbations and tracking, on the other

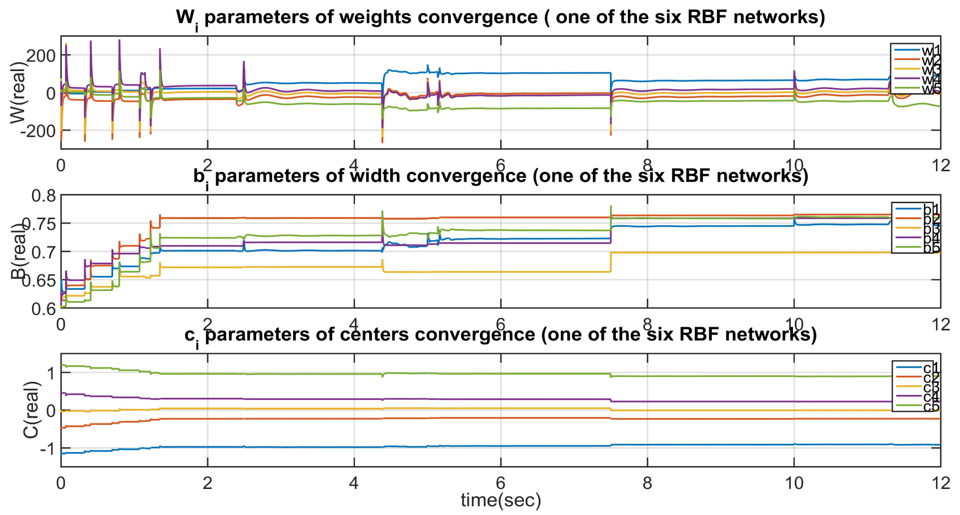


Figure 5-25: Weights $\hat{w}$, widths $\hat{b}$, and centers $\hat{c}$ convergence in one of the blocks

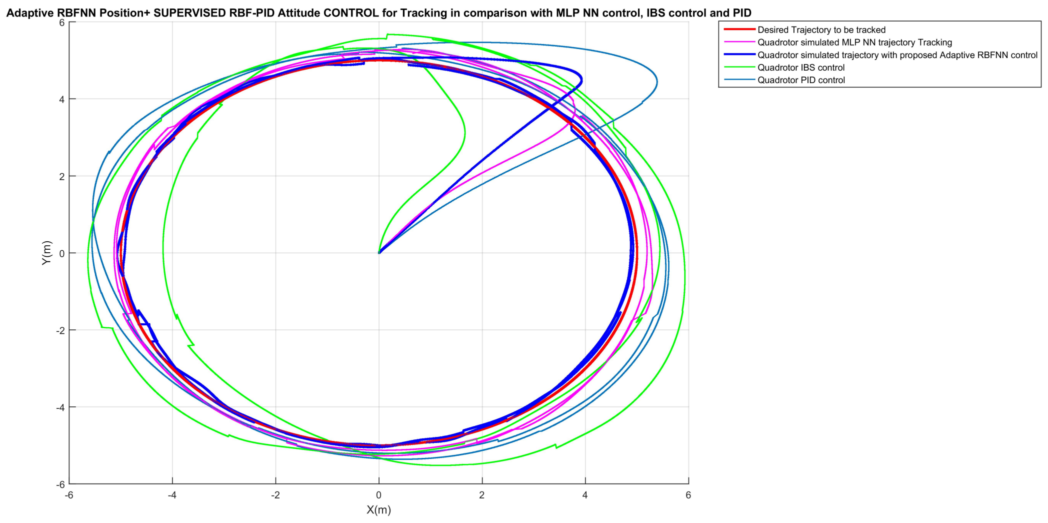


Figure 5-26: Top view of tracking comparison; the desired helical trajectory in red, the green trajectory is IBS-based control, the blue colored trajectory is for the proposed adaptive RBFNN control, where the yellow and magenta are the PID and MLP control trajectories respectively

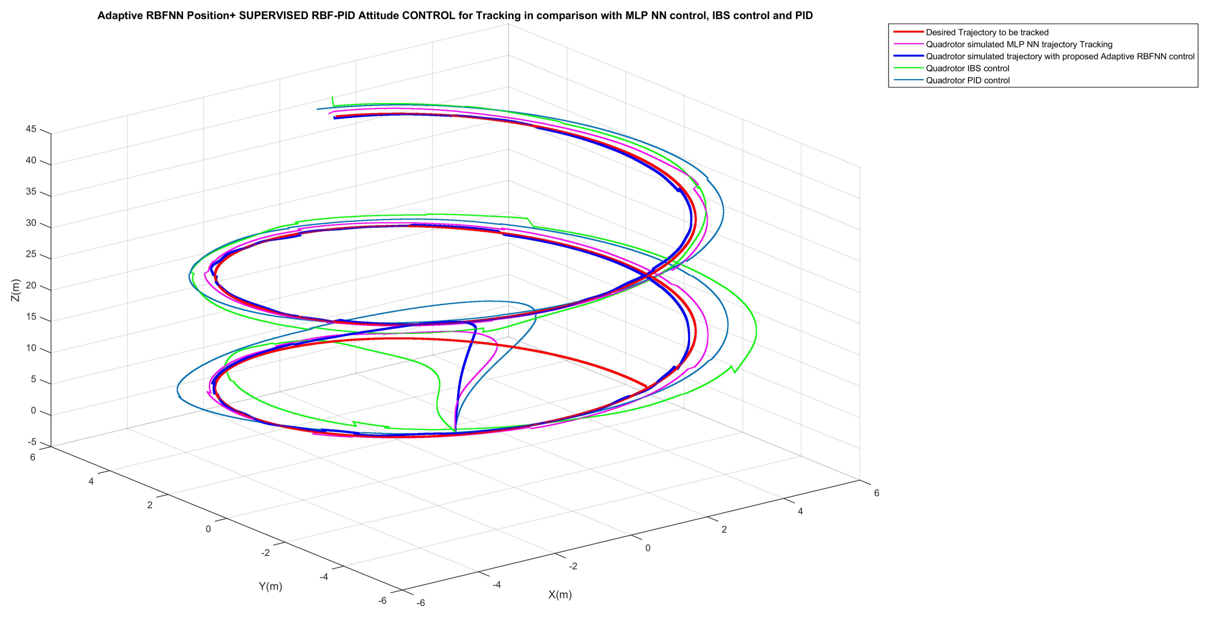


Figure 5-27: 3D trajectory tracking performance by PID, IBS, and CNN vs Adaptive RBFNN, the Desired trajectory is in red which is a helical design, the Green trajectory represents the IBS control results, PID and MLP are in magenta and sky blue color, where we can see tracking errors affected by the introduced noise and perturbation, Adaptive RBFNN control trajectory in Blue demonstrates the most robust tracking

hand, the PID, showed relatively better tracking with less overshooting in turns, and CNN control proved better performance than previous control methods in a matter of tracking of the desired trajectory as shown in Fig.5-27, The proposed adaptive RBFNN combined with supervisory control demonstrates clearly superior performance for tracking as it rejects all added disturbances.

It is noteworthy to state that the choice of supervised NN control for the disturbed attitude was crucial, several papers used the adaptive NN disturbance compensation for the attitude, however, the initial error at the start of the tracking sequence is significant (over $e_{(\theta,\phi,\psi)} > \pi Rad$) which is unrealistic and can't be applied in real life scenarios as it is beyond the recovery envelope of the quadrotor. [8]

[8]GitHub codes - RBFNN control.

Chapter 6

Intelligent Applications

In this chapter, several developed applications are discussed, and the trajectory generation either dynamically or statically is shown, a trajectory as a time reference path has been calculated to avoid static obstacles, and was generated for bio-inspired landing, where the velocity was decided by a fuzzy logic controller, the optimal trajectory was discussed but never used as it is much expensive in matter of computational time. Besides that, the soft landing process on a special design landing pad was studied, starting from the design of the pad to its detection, identification to its markers, and then their localization in the 3D domain to attributes and calculate the dynamic trajectory for landing. Obstacle avoidance is considered as well as a vital task for quadrotor control, however generating the correct dynamic trajectory and ensuring the right control will ensure satisfactory avoidance.

6.1 Trajectory generation

6.1.1 Dynamic optimal Trajectory generation

Optimal trajectory generation considering the dynamic model is the process of designing a trajectory that optimizes an index of performance while satisfying a set of constraints and following the dynamical path of the process model. In other words, it is a technique for computing an open-loop solution to an optimal control problem. There are numerous methods to obtain the dynamics-based optimal trajectory, there is no best method, however, a technique maybe more adequate for a precise problem than others. For quadrotor control, the optimal trajectory is nonlinear programming, which is time expensive method considering adding more constraints for obstacle avoidance which complicates exponentially the size of the problem.

Optimality problem

Let's cosider a dynamical system such $x \in R^n$:

$$\dot{x} = f(x, u) \tag{6.1}$$

Finding a trajectory that follows the path of 6.1, and that minimizes the performance index functional:

$$J = \int_{t_o}^{t_f} L(x, x, t)dt + \phi(xf, uf, tf) \tag{6.2}$$

With respect to the following trajectory constraints:

$$\begin{aligned} {}_0 &\leq \psi_0(x_0, u_0) \leq u_0, \\ l_f &\leq \psi_f(x_f, u_f) \leq u_f \\ l_t &\leq S(x, u, t) \leq u_t \end{aligned} \tag{6.3}$$

The functions $\psi_0 \in R^{N_0}$,$\psi_f \in R^{N_f}$,$S \in R^{N_t}$ are assumed to be as least C^2 functions *(function has both a continuous first derivative and a continuous second derivative).* The final time t_f could be either fixed or free. It is important to examine the Necessary Conditions of Optimality for Constrained Systems in advance, as an optimization problem may not have a solution. The solution to such a mathematical problem entails satisfying the Necessary Conditions of Optimality for Constrained Systems. These conditions necessitate the utilization of the Hamiltonian H, as well as the auxiliary functions Γ and Φ, to reformulate the performance index by incorporating the dynamics and transforming inequalities into equations based on unknown parameters.

Such methodology is expansive in a matter of calculation time, and incorporating it in a closed-loop control is almost beyond the objectives of miniature quadrotor control.

6.1.2 3D SPLINE Trajectory generation

The desired trajectory aims to avoid static obstacles, the UAV has to follow a time referenced path between the initial and the terminal position. The generation of the desired trajectory is performed separately from the control, this trajectory design challenge is to prove the autonomous ability of the quadrotor for navigation.

Dynamic optimal trajectory planning for obstacles avoidance is out of the scope of this study, so, we will be limited just to consider the way-points planning generation of smooth 3D Spline trajectory. One way to do it, is by defining the Cartesian coordinates of critical points in a remote supervision mode with constraints to avoid the obstacles, then generate a third-degree Cartesian polynomial between every two waypoints ensuring the smoothness between every two adjacent trajectories polynomials.

Design of the way point trajectory

By considering a motion from initial state: $A_i = [x_i, y_i, z_i]$ with $V_i = [v_{xi}, v_{yi}, v_{zi}]$ to an intermediate state of vector space, denoted: $A_{i+1} = [x_{i+1}, y_{i+1}, z_{i+1}]$ with $V_{i+1} = [v_{xi+1}, v_{yi+1}, v_{zi+1}]$.

Spline Trajectory interpolated by piece-wise third degree polynomials for every sub-interval, avoiding the *'Runges phenomenon'* for higher degrees can be expressed as :

$$\begin{cases} P_{x,i}(t) = a_{x,i} * t^3 + b_{x,i} * t^2 + c_{x,i} * t + d_{x,i} \\ P_{y,i}(t) = a_{y,i} * t^3 + b_{y,i} * t^2 + c_{y,i} * t + d_{y,i} \\ P_{z,i}(t) = a_{z,i} * t^3 + b_{z,i} * t^2 + c_{z,i} * t + d_{z,i} \end{cases} \tag{6.4}$$

Besides, the determined initial moving and final stopping states, every polynomial satisfies the intermediate initial state at "t_i" and intermediate final state at "t_{i+1}" with smoothness verified by:

$$\begin{cases} P_{x,i}(t_{i+1}) = P_{x,i+1}(t_i) \\ P_{y,i}(t_{i+1}) = P_{y,i+1}(t_i) \\ P_{z,i}(t_{i+1}) = P_{z,i+1}(t_i) \end{cases} \tag{6.5}$$

$$\begin{cases} \dot{P}_{x,i}(t_{i+1}) = \dot{P}_{x,i+1}(t_i) \\ \dot{P}_{y,i}(t_{i+1}) = \dot{P}_{y,i+1}(t_i) \\ \dot{P}_{z,i}(t_{i+1}) = \dot{P}_{z,i+1}(t_i) \end{cases} \tag{6.6}$$

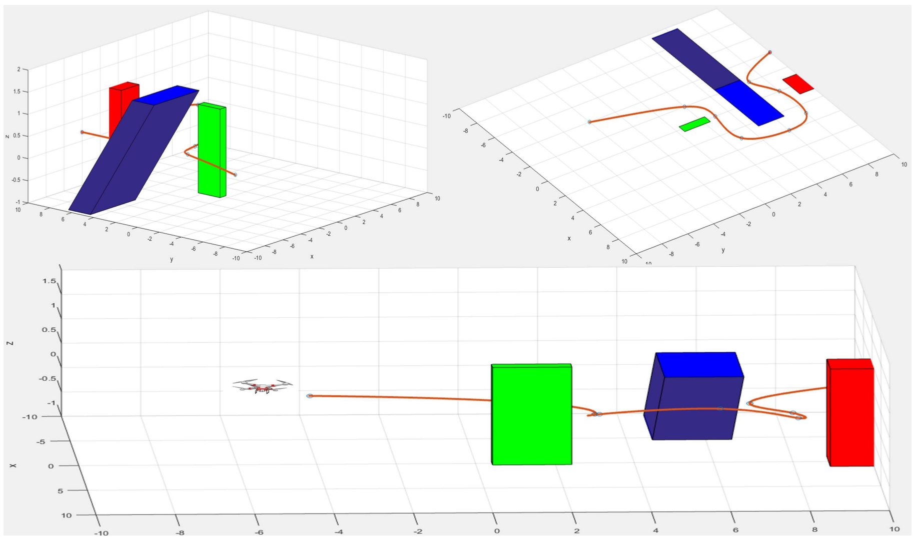


Figure 6-1: 3D Spline generated trajectory to avoid obtacles.

By solving those systems of equations, all parameters of polynomials are defined. An execution example is mentioned in Fig.6-1, where the UAV avoids collision with pre-defined blocks ensuring smoothness by Spline trajectory[1].

6.1.3 Trajectory generation by Minimum Jerk

Trajectory generation from optimization of time or effort is not within the scope of this Section. The geometric treatment of the problem simplifies both the modeling and control law for the system [77], min Jerk was considered in this work, Vijay et al proved that a min snap is equivalent to a min effort for quadrotor trajectory generation [69].

Minimum Jerk trajectory

Neville Hogan developed the notion of jerk optimization by minimizing the position third-time derivative denoted by three dotted X, aiming for the smoothness of the 3D position $X(t)$, the optimization of jerk can be written as:

$$J = \int_{t0}^{tf} \dddot{X}(t)^2 dt = \partial^3 X(t)/\partial t^3 \tag{6.7}$$

Euler Lagrange Equation is used to solve the optimization considering $L = \dddot{X}(t)^2$, the Necessary condition satisfied by the "optimal" function.

[1]GitHub codes - SPLINE trajectory.

$$\partial L/\partial X - d/dt(\partial L/\partial \dot{X}) + d^2/dt^2(\partial L/\partial \ddot{X}) - d^3/dt^3(\partial L/\partial \dddot{X}) = 0 \tag{6.8}$$

The solution of such an optimization is: $X^{(6)}(t) = 0$ Hence

$$X(t) = c_5 t^5 + c_4 t^4 + c_3 t^3 + c_2 t^2 + c_1 t + c_0 \tag{6.9}$$

Where c_i are found by satisfying the boundary conditions of initial and final states. As we aim to land on a moving landing pad, a control loop of the vision and estimation updates the target position, thus, the final state changes such the optimization as well.

For every iteration of the estimation, the optimization process calculates the parameters online using the initial dynamic state and the final pose estimate.

For 3D jerk optimization in our quadrotor landing case, we can write:

$$J = \int_{t0}^{tf} \dddot{x}(t)^2 + \dddot{y}(t)^2 + \dddot{z}(t)^2 dt \tag{6.10}$$

Which define a minimum jerk 3D trajectory:

$$\begin{cases} x(t) = c_{x5}t^5 + c_{x4}t^4 + c_{x3}t^3 + c_{x2}t^2 + c_{x1}t + c_{x0} \\ y(t) = c_{y5}t^5 + c_{y4}t^4 + c_{y3}t^3 + c_{y2}t^2 + c_{y1}t + c_{y0} \\ z(t) = c_{z5}t^5 + c_{z4}t^4 + c_{z3}t^3 + c_{z2}t^2 + c_{z1}t + c_{z0} \end{cases} \tag{6.11}$$

where c_{ij} ($i \in \{x, y, z\}$ and $j \in \{1, 2, 3, 4, 5, 0\}$) are found by satisfying the boundary conditions of initial and final states. During landing, a control loop of the vision and estimation updates the target position, thus, the final state changes such as the optimization as well. For every iteration of the estimation, the optimization process calculates the parameters online using the initial dynamic state and the final pose estimate.

To find c_{ij}, it is necessary to solve the three systems of six algebraic equations each, between the initial and final states for every dimension. The problem of the solution is that dynamic resolution, requires a good estimate of quadrotor position, velocity, and acceleration, where the need for the KF with sensors data fusion which is detailed in section 5.2. Additionally, the final time of each iteration should be calculated based on the required speed for that position. This latter is decided by a fuzzy logic controller.

The minimum jerk 3D trajectory mentioned by Eqs.(6.11) describes system smoothness between initial and final times. For the aimed case, the final time is unknown, so the problem is referred to be a feed forward control. Based on this approach, the optimization will be divided into segments, for each,we look to find a new smooth trajectory depending on the quadrotor estimated states and the relative pose to the landing pad in motion.

A simple reformulation of this problem lies in normalizing the time between $[t_0, t_f]$:

$$\tau = (t - t_0)/T \tag{6.12}$$

$$T = t_f - t_0. \tag{6.13}$$

The 3D minimum jerk Trajectory will be generated by:

$$\begin{cases} x(t) = c_{x5}\tau^5 + c_{x4}\tau^4 + c_{x3}\tau^3 + c_{x2}\tau^2 + c_{x1}\tau + c_{x0} \\ y(t) = c_{y5}\tau^5 + c_{y4}\tau^4 + c_{y3}\tau^3 + c_{y2}\tau^2 + c_{y1}\tau + c_{y0} \\ z(t) = c_{z5}\tau^5 + c_{z4}\tau^4 + c_{z3}\tau^3 + c_{z2}\tau^2 + c_{z1}\tau + c_{z0} \end{cases} \tag{6.14}$$

where,

$$d\tau/dt = 1/T. \tag{6.15}$$

Hence, the velocity in time reference is:

$$\begin{cases} \dot{x}(t) = 5c_{x5}\tau^4/T + 4c_{x4}\tau^3/T + 3c_{x3}\tau^2/T + 2c_{x2}\tau/T + c_{x1}/T \\ \dot{y}(t) = 5c_{y5}\tau^4/T + 4c_{y4}\tau^3/T + 3c_{y3}\tau^2/T + 2c_{y2}\tau/T + c_{y1}/T \\ \dot{z}(t) = 5c_{z5}\tau^4/T + 4c_{z4}\tau^3/T + 3c_{z3}\tau^2/T + 2c_{z2}\tau/T + c_{z1}/T \end{cases} \tag{6.16}$$

And,the Acceleration:

$$\begin{cases} \ddot{x}(t) = 20c_{x5}\tau^3/T^2 + 12c_{x4}\tau^2/T^2 + 6c_{x3}\tau/T^2 + 2c_{x2}/T^2 \\ \ddot{y}(t) = 20c_{y5}\tau^3/T^2 + 12c_{y4}\tau^2/T^2 + 6c_{y3}\tau/T^2 + 2c_{y2}/T^2 \\ \ddot{z}(t) = 20c_{z5}\tau^3/T^2 + 12c_{z4}\tau^2/T^2 + 6c_{z3}\tau/T^2 + 2c_{z2}/T^2 \end{cases} \tag{6.17}$$

The last system of Equations is used to form the algebraic equations between an initial state given by the enhanced EKF estimation at the initial instant of the current iteration and the final time state defined by the desired landing on pose estimate of the pad. The resolution of the system should be in function of T.

Initial state:

$$x(t_0) = x_0 \quad v_x(t_0) = v_{x0} \quad a_x(t_0) = a_{x0}$$
$$y(t_0) = y_0 \quad v_y(t_0) = v_{y0} \quad a_y(t_0) = a_{y0}$$
$$z(t_0) = z_0 \quad v_z(t_0) = v_{z0} \quad a_z(t_0) = a_{z0}$$

Where final state is trivial described by:

$$x(t_f) = x_pad \quad v_x(t_f) = 0 \quad a_x(t_f) = 0$$
$$y(t_f) = y_pad \quad v_y(t_f) = 0 \quad a_y(t_f) = 0$$
$$z(t_f) = z_pad \quad v_z(t_f) = 0 \quad a_z(t_f) = 0$$

Resolution of the system should be in function of T:

At $t = t_0$

$$\begin{gathered} c_{x0} = x_0 \quad c_{x1} = Tv_{x0} \quad c_{x2} = T^2 a_{x0}/2 \\ c_{y0} = y_0 \quad c_{y1} = Tv_{y0} \quad c_{y2} = T^2 a_{y0}/2 \\ c_{z0} = z_0 \quad c_{z1} = Tv_{z0} \quad c_{z2} = T^2 a_{z0}/2 \end{gathered} \tag{6.18}$$

at $t = t_f$, normalized time τ will be τ =1, thus:

$$\begin{aligned} c_{x3} &= 10(x_f - x_0) - 6Tv_{x0} - 3T^2 a_{x0}/2 \\ c_{x4} &= -15(x_f - x_0) + 8Tv_{x0} + 3T^2 a_{x0}/2 \\ c_{x5} &= 6(x_f - x_0) - 3Tv_{x0} - T^2 a_{x0}/2 \\ c_{y3} &= 10(y_f - y_0) - 6Tv_{y0} - 3T^2 a_{y0}/2 \\ c_{y4} &= -15(y_f - y_0) + 8Tv_{y0} + 3T^2 a_{y0}/2 \\ c_{y5} &= 6(y_f - y_0) - 3Tv_{y0} - T^2 a_{y0}/2 \\ c_{z3} &= 10(z_f - z_0) - 6Tv_{y0} - 3T^2 a_{z0}/2 \\ c_{z4} &= -15(z_f - z_0) + 8Tv_{z0} + 3T^2 a_{z0}/2 \\ c_{z5} &= 6(z_f - z_0) - 3Tv_{z0} - T^2 a_{z0}/2 \end{aligned} \tag{6.19}$$

Where all initial states are resulting from the estimation of Kalman filter, and the final ones result from the pose estimation of the moving pad.

Feedback controlled trajectory

The described trajectory by $P(t) : \big(x(t), y(t), z(t)\big)$ is known and it is T dependent. So at every instant labeled by t_0, we define:

$$P_x(t) = [x(t) \;\; \dot{x}(t) \;\; \ddot{x}(t)]^T \tag{6.20}$$

$$P_{x0} = [x_0 \;\; v_{x0} \;\; a_{x0}]^T \tag{6.21}$$

Deriving the jerk equations in Eqs.6.14, we find:

$$\begin{cases} \dddot{x}(t0) = 6c_{x3}/T^3 = 60/T^3(x_f - x_0) - 36/T^2 v_{x0} - 9/T a_{x0} \\ \dddot{y}(t0) = 6c_{y3}/T^3 = 60/T^3(y_f - y_0) - 36/T^2 v_{y0} - 9/T a_{y0} \\ \dddot{z}(t0) = 6c_{z3}/T^3 = 60/T^3(z_f - z_0) - 36/T^2 v_{z0} - 9/T a_{z0} \end{cases} \tag{6.22}$$

Eqs.(6.22) present three kinematics for the variables $x(t0), y(t0)$ & $z(t0)$ between initial time of each iteration t_0, and the estimated landing time t_f.

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -60/T^3 & -36/T^2 & -9/T \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 60/T^3 \end{bmatrix} x_f \tag{6.23}$$

Dynamics described in Eq.(6.23) will provide a smooth trajectory that minimizes the jerk. The unknown T value will be defined by an expert FLC, that considers many inputs to validate the right velocity to target and the estimated time for the landing. By analogy, we find dynamics for $y(t)$ and $z(t)$.

Jerk, Snap, crackle, and pop are the third, fourth, fifth, and sixth derivatives of position respectively. In [93] Vijay et al used snap optimization to generate trajectory with minimal effort for the quadrotor, Nevertheless, Richardson et al analyzed how X(t) would change as a function of 'n' in the optimization functional.

$$J = \int_{t0}^{tf} \big(x(t)^{(n)}\big)^2 dt \tag{6.24}$$

They demonstrated that at higher derivative order the more solution tends to prove a higher change of speed at the trajectory inflection point, change becomes narrower as we minimize jerk, snap and crackle [85]. in our project, we aimed essentially for smooth control and landing which justifies the minimization of the Jerk.

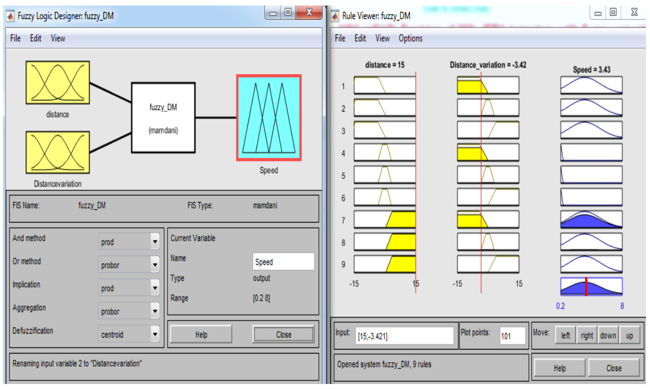
6.2 Fuzzy Velocity control

In [37], Alvika et al processed the velocity control by a log polynomial to smooth the trajectory. However, to bio-inspire the human piloting, a FLC was designed as a velocity controller, other FLC approaches for adaptive intelligent control were discussed in [72].

System of Eqs.(6.23) represents the time position of the quadrotor in the normalized time that depends on T, the total descent time. Using expert piloting data, we aimed to generate the T via fuzzy logic processing to emulate the human piloting grasp. For that, the FLC was conceptualized to generate the total time T based on instantaneous speed and distance to the target. The classical approach to determine the fuzzy rules base was processed, the antecedent and consequent partitions will be designed by human expert knowledge for quadrotor landing. Due to the complexity of time estimation, the average speed is based on the processing of quadrotor dynamics, the initial state, and relative pose and velocity.

The velocity FLC basically, consists of:

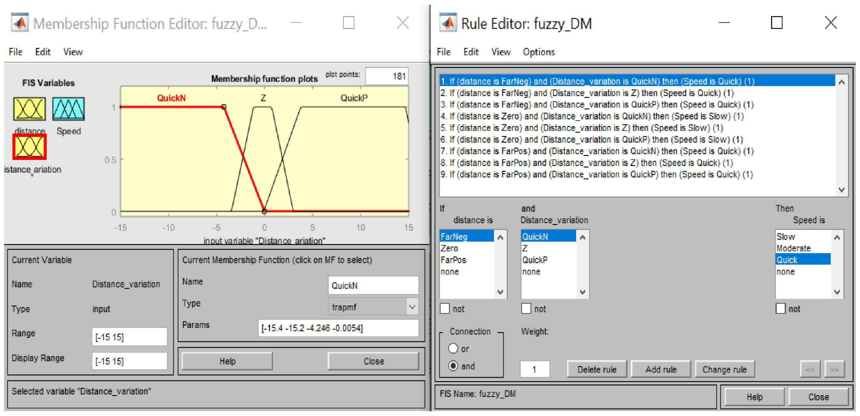
a) Fuzzification: Both of the estimation of localization from the EKF and the relative estimated pose from the camera, in addition to the relative velocity are transformed to functions of memberships defined by the linguistic variables. These fuzzy sets will be used by the inference mechanism to validate the next stage rules as demonstrated in Fig.6-2.



FIS Name: fuzzy_DM
FIS Type: mamdani
And method: prod
Or method: probor
Implication: prod
Aggregation: probor
Defuzzification: centroid
Current Variable
Name: Speed
Type: output
Range: [0.2 8]
Renaming input variable 2 to "Distancevariation"
Input: [15;-3.421]
Plot points: 101
Opened system fuzzy_DM, 9 rules

Figure 6-2: FLC model for velocity control

b) Inference mechanism: it is the processing stage that evaluates in every execution cycle each one of a set of *"if … then … statements"* already stored in the rules base as mentioned in Fig.6-4 6-3, which are a combination of valid rules filled by Experienced agent.



Current Variable
Name: Distance_variation
Type: input
Range: [-15 15]
Display Range: [-15 15]

Current Membership Function (click on MF to select)
Name: QuickN
Type: trapmf
Params: [-15.4 -15.2 -4.246 -0.0054]

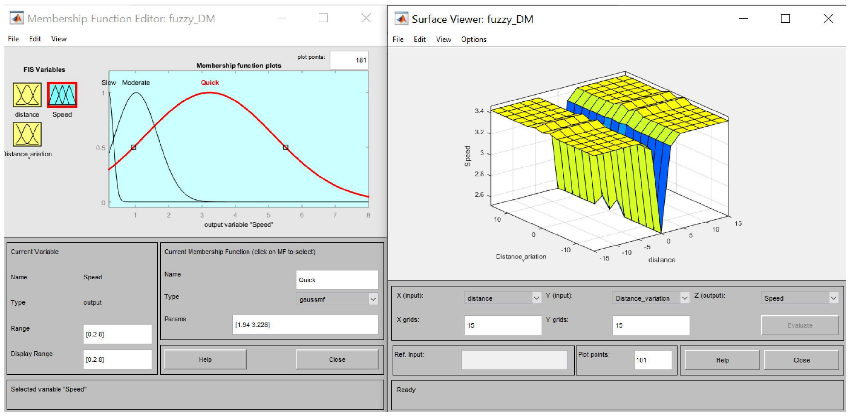
Selected variable "Distance_variation"

1. If (distance is FarNeg) and (Distance_variation is QuickN) then (Speed is Quick) (1)
2. If (distance is FarNeg) and (Distance_variation is Z) then (Speed is Quick) (1)
3. If (distance is FarNeg) and (Distance_variation is QuickP) then (Speed is Quick) (1)
4. If (distance is Zero) and (Distance_variation is QuickN) then (Speed is Slow) (1)
5. If (distance is Zero) and (Distance_variation is Z) then (Speed is Slow) (1)
6. If (distance is Zero) and (Distance_variation is QuickP) then (Speed is Slow) (1)
7. If (distance is FarPos) and (Distance_variation is QuickN) then (Speed is Quick) (1)
8. If (distance is FarPos) and (Distance_variation is Z) then (Speed is Quick) (1)
9. If (distance is FarPos) and (Distance_variation is QuickP) then (Speed is Quick) (1)

If distance is: FarNeg, Zero, FarPos, none
and Distance_variation: QuickN, Z, QuickP, none
Then Speed is: Slow, Moderate, Quick, none

FIS Name: fuzzy_DM

Figure 6-3: Fuzzy decision maker: Rules used in the Inference mechanism



Current Variable
Name: Speed
Type: output
Range: [0.2 8]
Display Range: [0.2 8]

Current Membership Function (click on MF to select)
Name: Quick
Type: gaussmf
Params: [1.94 3.228]

Selected variable "Speed"

X (input): distance
Y (input): Distance_variation
Z (output): Speed
X grids: 15
Y grids: 15
Ref. Input:
Plot points: 101

Ready

Figure 6-4: Fuzzy decision maker: Defuzzification and surface of control

c) Defuzzification: The output stage gives the average speed that converts the combined appropriate rules from the inference stage to a decision velocity of the quadrotor.

FLC is opted for velocity control, as it is more linguistic and to inspires the expert human pilot approach to land with enough robustness and softness. FLC inputs are described by the instantaneous relative position, velocity, and acceleration states of the quadrotor, output is the average speed to the target. Landing time T is then calculated. This method presents a soft landing because of the use of expert rules. The block diagram of the fuzzy logic velocity controller is illustrated in Fig. 6-5.

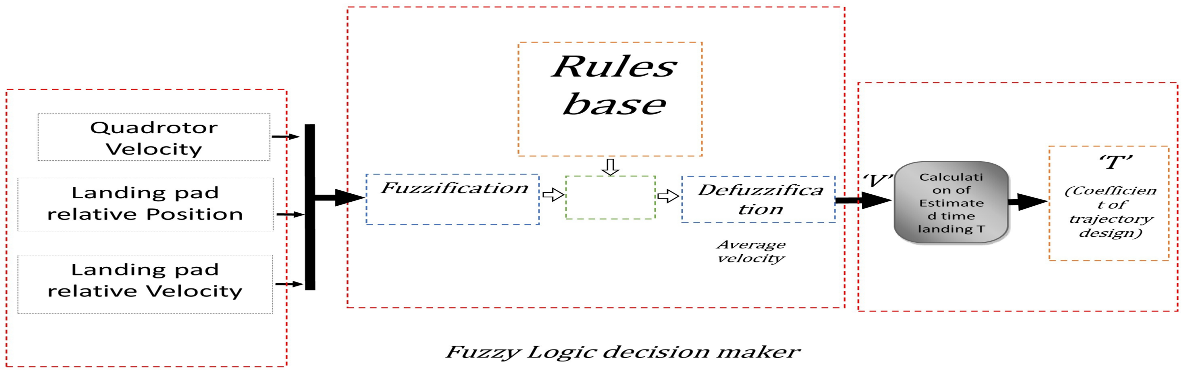

Quadrotor Velocity
Landing pad relative Position
Landing pad relative Velocity
Rules base
Fuzzification
Defuzzification
Average velocity
'V'
Calculation of Estimated time landing T
'T' (Coefficient of trajectory design)
Fuzzy Logic decision maker

Figure 6-5: Velocity control and time to land estimation by Fuzzy logic control scheme

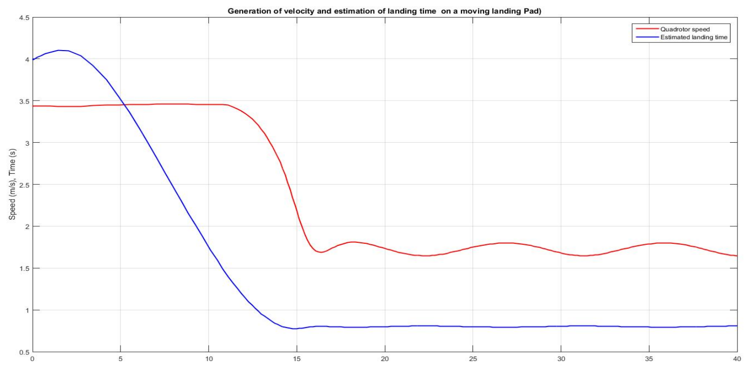


Figure 6-6: Estimation of velocity of landing and Time to land by the FLC decision maker

The total Time of landing T is based on the average velocity and distance to the landing pad and can be calculated as follows:

$$T = \sqrt{\Delta x_{p,q}^2 + \Delta y_{p,q}^2 + \Delta z_{p,q}^2}/V + T_{min} \tag{6.25}$$

$\Delta(x, y, z)_{p,q}$, represents the Cartesian distances between the quadrotor and pad, V is the average velocity estimated by the FLC, T_min is a safety margin[2].

[2]GitHub codes - Soft landing Trajectory.

6.2.1 Trajectory generation algorithm for the soft landing

```
Algorithm 3: Min Jerk Trajectory generation with Fuzzy velocity control
Result: Estimation of time for landing 'T' and Cartesian coordinates
Initialization and load data file;
Import state data from EKF;
Import data from the relative pose estimator ;
while 'Auto-landing is ON' & 'ArUco marker detected' do
    if All Data are Available then
        - Fuzzification of inputs to membership functions;
        - Process Inference mechanism ;
        - Defuzzification of outputs and decision making on Average velocity 'V';
    end
    Calculate the estimated time for landing 'T' from the average speed and
      instantaneous distance Eq.6.25;
    Calculate trajectory coefficients by Eq.6.19;
    Calculate $x(t)$, $y(t)$, and $z(t)$ based on Eq.6.14;
    Send desired Trajectory coordinates to tracking control;
end
if 'Auto-landing is OFF' OR 'ArUco Markers NOT detected' then
    - Hold quadrotor position;
    - Send flag to control station ;
    - Request new GPS position;
end
```

6.3 Vision based Soft Landing

6.3.1 Introduction

The quadrotor is a highly captivating type of UAV that has garnered significant attention in research and literature. This is primarily due to its remarkable flexibility, maneuverability, and the unique capability to hover. The quadrotor has found diverse applications in various fields, including detection, civil and military surveillance, commercial and medical services, and even industrial settings. Its versatility and functionality make it a compelling subject of study and exploration. [84]-[70]. Quadrotors have become able to perform complicated maneuvers and autonomous tasks even in harsh conditions or hazardous environments, thus comes the option of autonomous landing as a vital task for many missions and as a fail-safe in case of some technical issue during the flight such as GPS denial or low power.

Diverse papers tackled the quadrotor landing, Zhe et al. [105] developed a vision-based autonomous landing on a moving surface vessel with PID tracking by visual detection and estimation of the landing area, relative localization for UAV-based image was addressed [101]. In [28] Vertical Taking Off and Landing (VTOL) landing with a robust PID control was addressed. An efficient 3-D time-optimal trajectory tracking by a quadrotor model on a moving pad [44]. Yuhua et al. proposed an autonomous landing processing on a moving with Lyapunov stability demonstration [83]. However, those previous papers opted for PID control which unfortunately doesn't consider either disturbance or model uncertainty, In addition, it affects slightly the tracking performance. Some researchers considered the introduction of KF as uncertain control smooth lanfing processing [42] [43] [3].

To comply with the autonomous landing of the quadrotor, computer vision processing is crucial to determine the relative pose to dynamically generate the trajectory for the landing. Many algorithms of vision subject to detection and estimation for autonomous landing have been developed. ArUco markers were widely used because of the easiness of detection and the efficiency of the identification [43]-[83]-[105]- [70]. Whereas the pose estimation varies from direct methods that use the PnP algorithm [86]-[83]-[9], or sophisticated and intelligent methods based on neural networking [58]. The visual processing is so sensitive in the matter of data extraction (size, pattern, and pose) to camera type and its calibration, to the shutter effect, to the velocity of the drone's landing, and to the external light and mirror effect [86].

It is critical to consider the noise and disturbance model in the control and estimation during the robust landing process, especially in outdoor applications, thus the estimation of quadrotor state vector was enhanced by Madgwick data fusion of asynchronous sensors filter [99] combined with an Extended KF, another filter was set for best estimation of markers visual localization [33].

Based on the literature, some researchers used pursuit guidance while simultaneously descending with the UAV in altitude [37], wherein [30], a fusion of monocular Simultaneously Localization And Mapping (SLAM) and INS was utilized in the landing of the quadrotor in bio-inspired guidance method which is based on Tau theory. In [57], complex maneuvers were illustrated by a robust trucking control that can be constrained for landing. To enhance the bio-inspired soft landing, a novel Fuzzy decision maker process as detailed in last section is used to estimate the landing time via the average possible speed, the Fuzzy sets use the observed state vectors of both the quadrotor and moving pad to decide the velocity of the landing. This method controls the velocity limits to design the optimal trajectory to track for the landing as an expert guidance control. MPC is chosen for the tracking performance instead of classic PID, MPC is adaptive and can handle the considered disturbance and Gaussian noise.

The proposed soft landing approach separates the process into two interconnected parts: First, the detection of markers, identification of at least one visible marker depending on the height, and then determining the relative pose of the pad center with consideration to the motion of both the quadrotor and the landing pad. Then, the generation of minimal Jerk trajectory to ensure smoothness during landing, where MPC is responsible for tracking iteratively the desired trajectory to land. The smooth velocity profile is controlled by the FLC decision-maker. The opted philosophy is demonstrated in Fig.6-7.

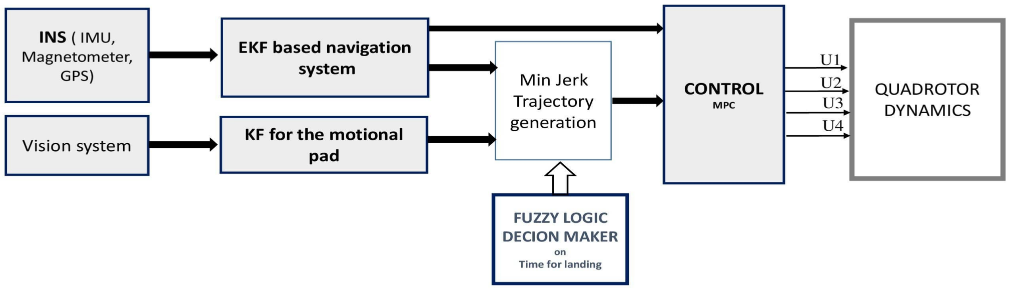

INS (IMU, Magnetometer, GPS)
Vision system
EKF based navigation system
KF for the motional pad
Min Jerk Trajectory generation
FUZZY LOGIC DECION MAKER on Time for landing
CONTROL MPC
U1
U2
U3
U4
QUADROTOR DYNAMICS

Figure 6-7: Autonomous Landing control processing

The landing application aimes to develop a complete process for the smooth autonomous landing of a quadrotor on a moving pad based on a vision algorithm using a monocular camera with robust position control. The iterative and adaptive Generation of trajectory was based on the optimal jerk. The process valued the following points:

- The design of a robust application of the pose estimation based on a special landing pad by multiple ArUco markers with different patterns at different sizes to ease the detection.

- The MPC based Tracking for better attenuation of external disturbance with a high performance comparison to PID control.
- Planning a mimic of natural decision making as expert human processing for landing by adding FLC for velocity control.
- Combine EKF with Magdwick data fusion of asynchronous sensors for state observation.

.

6.3.2 The detection of the relative pose estimation

Before engaging the landing control, the detection and the estimation of the relative pose of the moving landing pad are first processed. This subsection investigates this challenge, aiming to maximize the detection probability and to avoid miss-estimation samples and the limitation of the Field Of View (FOV).

Concept of detection of the landing pad

Many techniques for pad detection were investigated, such as color-based, or shape-based, other research was based on employing a complementary modular approach that uses data-driven methods for auto-docking such in [94]. In this thesis, we opted for the ArUco markers. The use of binary square fiducial markers provides enough data for the detection by the four corners, the identification from the binary codification, and the relative pose from projection processing. All those features are easy to be acquired by different types of cameras and from relatively higher altitudes. In addition, choosing different markers' designs at different sizes will ease the task of close detection at the same time as far ones, which ensures a robust detection all along the trajectory of landing. As the quadrotor got close to the landing pad, some markers leave slowly the Field Of View (FOV), at this level, the detection will be based only on the relatively smaller markers, contrary to larger markers that will be used at higher altitudes. Yuhua Qi et al. proposed a similar aspect of detection for a low-cost camera in [83]. The redundancy of markers will enhance the total probability of detection and recall, and precision augments by the use of small patterns at low altitudes.

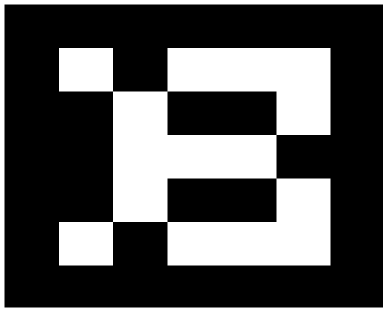


Figure 6-8: An Aruco marker used for detection

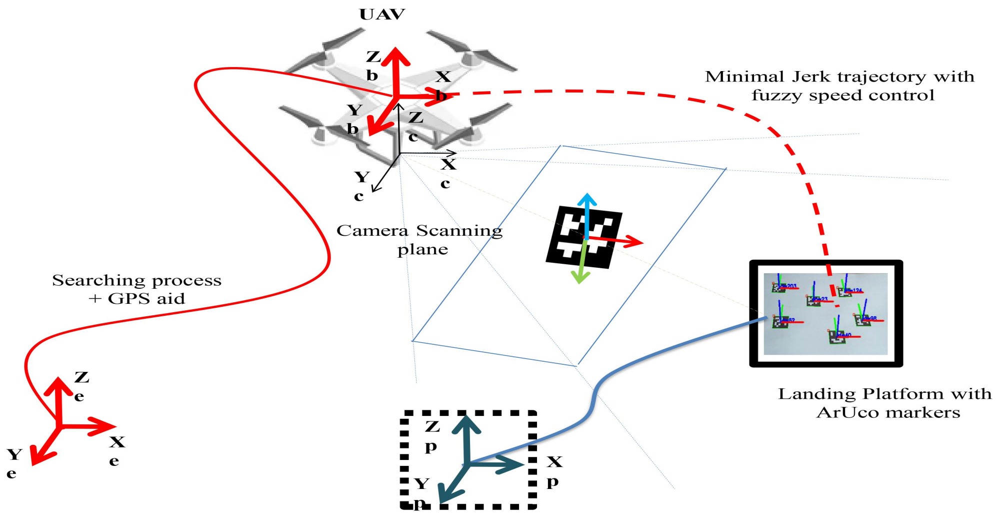
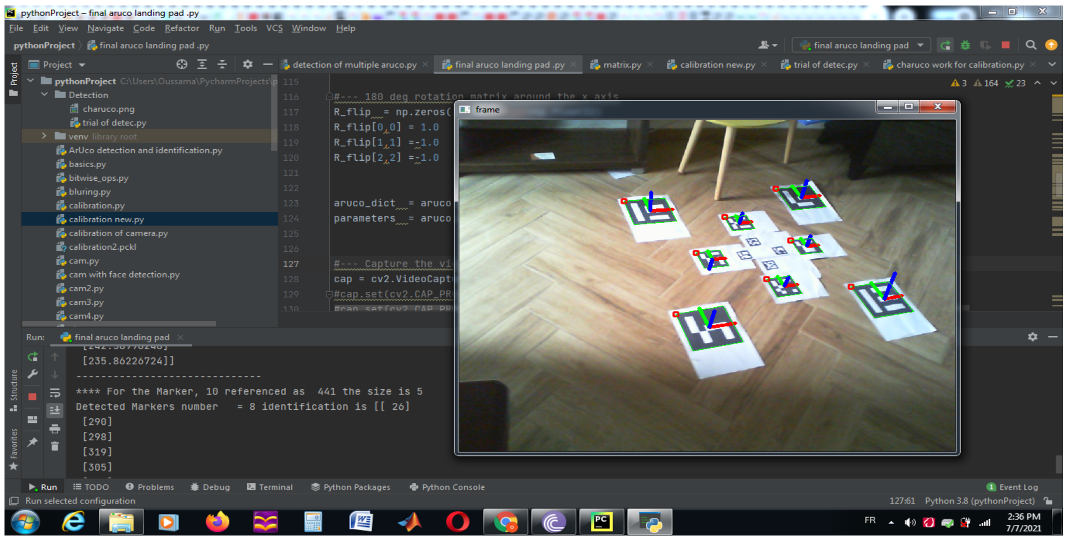

UAV
Z b
X b
Y b
Z c
X c
Y c
Minimal Jerk trajectory with fuzzy speed control
Camera Scanning plane
Searching process + GPS aid
Landing Platform with ArUco markers
Z e
X e
Y e
Z p
X p
Y p

Figure 6-9: Landing philosophy references of motion

Robustness of the pose estimation

The ArUco library was used as a base for markers design [36]. At every iteration image, the pose estimation process is initially based on the markers detection algorithm that recognizes the detected markers and identifies them by a binary codification. Recognition is performed by analyzing the inner codification and attributing the marker in a canonical form by differentiating between black and white surfaces using a simple Otsu thresholding technique, and excluding the wrong detection, besides identifying markers from the predefined list. Then,



```
#--- 180 deg rotation matr
R_flip  = np.zeros(
R_flip[0,0] = 1.0
R_flip[1,1] =-1.0
R_flip[2,2] =-1.0


aruco_dict  = aruco
parameters  = aruco


#--- Capture the vi
cap = cv2.VideoCapt
#cap.set(cv2.CAP_PR
```

```
 [235.86226724]]
------------------------------
**** For the Marker, 10 referenced as  441 the size is 5
Detected Markers number   = 8 identification is [[ 26]
 [290]
 [298]
 [319]
 [305]
```

Figure 6-10: Detection and estimation of the relative pose)

the algorithm ran an estimation process for the relative pose of the landing pad based on the OpenCV library. A specially designed filter decides which markers to be considered based on the average of actual relative distance estimations between the quadrotor and the landing pad, the filter extracts out the bad estimation based on the normalized variance to the mean. The marker estimate is considered if and only if:

$$|(\hat{Est}(i)^2 - (\sum_{j=1}^{N} \hat{Est}(j)/N)^2)/\mathbb{V}(Est)| > 0.2 \tag{6.26}$$

$\hat{Est}(i)$ represents the Cartesian distance estimate from the Marker i, $\mathbb{V}(Est)$ is the variance of all detected distances estimates. The inner codification choice of ArUco markers was based on simple detection and reading from a far distance and denser for smaller ones to improve close estimation. For better pose estimation performance, it is judicious to calibrate the camera, A special calibration code that uses OpenCV for camera calibration tools and Charuco image was used to extract the camera matrix and the distortion vector needed for the pose estimator[3].

6.3.3 Vision processing results

Tables 6.1. 6.2, and 6.3. were made considering the hovering mode of the quadrotor to optimize the pad design emulating the real aspect of vibration, flight disturbance, and high-

[3]GitHub codes - Camera calibration.

Table 6.1: Multiple markers vs single marker detectability and precision of the pose estimation

Markers	Detectability (recall)	Estimate precision
Single Marker	97.1	4.89 cm
Multiple markers	99.3	2.55 cm

Table 6.3: Precision of pose estimation at different sizes of markers from different heights. ((NA): Non Applicable values: are cases of no detection)

	0.3 m	0.6 m	1 m	3 m	7 m
5 cm	3.3 cm	5.1 cm	9.4 cm	(NA)	(NA)
10 cm	2.7 cm	3.4 cm	6.9 cm	19.6 cm	(NA)
18 cm	(NA)	(NA)	4.9 cm	12.9 cm	28.9 cm

frequency noise. For that many real-time images were taken and analyzed. Based on using one vs multiple markers and varying their sizes, experimental tests of detection probability and precision are explicit in the tables. 6.1. 6.2 and 6.3.

Besides the detection recall, the precision of the pose significantly increases by enabling multi measurement which attenuates the noise of calculation to a tolerated error margin of the general position determination.

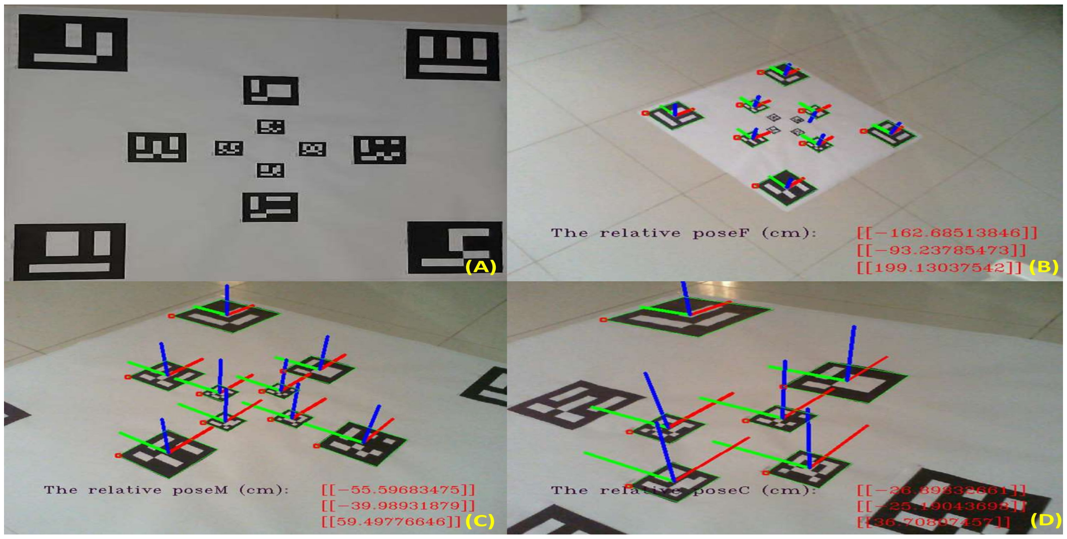
Table 6.2: Detection rate for different sizes of markers from different heights.

	0.3 m	0.6 m	1 m	3 m	7 m
5 cm	100 %	99.2 %	91.1 %	8.1 %	0 %
10 cm	95.2%	99.2 %	97.4 %	88.9 %	31 %
18 cm	0 %	87.7 %	97.9 %	99.8 %	88.6 %

Multiple markers increase the detection probability of the pad, and the estimation precision of the relative pose got better. In addition, the different sizes of markers play an important role in pose estimation and detection from different heights, low altitudes exploit small markers when larger ones are out of the FOV. Contrary to the high altitude detection, where we rely completely on larger markers as the smaller ones are less detectable with insufficient precision.

The designed landing pad was detected easily from different heights, and different angles of heading orientation, Fig.6-11 shows the detection and estimation from different positions.

Fig.6-11 shows different scenarios, Sub-Figure (A) shows the designed landing pad that contains multiple designs of different sizes, Sub-Figure (B) demonstrates the detection process and the pose estimation from a far pose, Cartesian coordinates are calculated and printed in red, the filter uses only larger markers even with the possible detection of closer ones, in Sub-Figure (C), The filter considers only medium size markers, estimation of the Cartesian pose coordinates is printed. In Sub-Figure (D), the pose estimation is based only on smaller markers as larger ones are out of FOV. The F, M, and C denotations show the filter choice of markers based on the relative distance[4].



```
The relative poseF (cm):   [[−162.68513846]]
                           [[−93.23785473]]
                           [[199.13037542]]
```

```
The relative poseM (cm):   [[−55.59683475]]
                           [[−39.98931879]]
                           [[59.49776646]]
```

```
The relative poseC (cm):   [[−26.89832661]]
                           [[−25.19043698]]
                           [[36.70807457]]
```

Figure 6-11: Detection and estimation of the relative pose from different positions (Sub-fig.A: shows the designed landing pad, Sub-fig.B: demonstrates a detection from 2.8 m away, Sub-fig.C demonstrates the pose estimation from 0.9 m, Sub-fig.D: shows a close pose estimation at the final stage of landing at 0.5 m)

Once the Pas is detected and the pose is estimated, the trajectory got generated and then tracked as explained in minimal Jerk trajectory tracking with MPC.

[4]GitHub codes - Pose estimation.

Chapter 7

Conclusion and future works

7.1 Conclusion

This thesis Initially presented a state of the art as a panoramic view of what researchers were developing recently as applications on quadrotors, then, dynamics modeling derivation was detailed based on Newton-Euler formalism to ease the perception of how the simulation model was obtained. Subsequently, Simulations of the synthesis of robust control laws for nonlinear multivariable uncertain systems based on three approaches PID, LQR & IBS validated the applicability of those control theories on autonomous quadrotors in order to track a surveillance Lissajous trajectory. Obtained results were satisfactory in a matter of performance and robustness despite adding noise and disturbance to the simulation model and the fact that the position of the quadrotor initially is far away from the targeted trajectory. Yet the drawbacks of those methods are relative to the desired robustness and the validation of the control model which remains so sensitive to the linearization process and the considered operation envelope. Hereafter, chapter IV discussed the application of building a quadrotor model with detailed hardware and software architectures.

Based on the fact that not all the states are measurable, and for the aim to attribute all states with high confidence. An optimal observer that was demonstrated quicker than standard EKF was developed based on a mathematical combination between Madgwick / EKF observer of the quadrotor which was enhanced by an asynchronous sensors fusion (MPU9250 and NEO-6M) to observe the quadrotor states vector and sensors bias for orientation precision of less than 1 deg. The full-state observer uses the quaternion AHRS filter of Madgwick data fusion of asynchronous sensors and EKF. The method showed quick convergence and very high performance as it is superior because of considering the model dynamics, in addition, the developed observer corrects the magnetic distortion, compensates for gyroscope bias drift, and cancels Gaussian noise. The observer can track all sensors' drifts and biases. The adaptive model was based on Jacobian linearization of the dynamical model to expand the operating space.

Intelligent control theory was investigated for robust control in the presence of uncertainties and perturbation. First, An adaptive LQG to track a generated Spline trajectory by remote supervision in order to avoid the known static obstacles. The adaptive LQG showed excellent tracking of the desired trajectory and was superior to the classic nested loop design based on PID control. Robustness to the added noise and disturbance was verified. This optimal method is highly computationally loaded, but It is showed high accuracy, It is possible to alleviate the computation cost by opting for a static linearized model but this will reduce performance, especially at higher control angles.

An optimal minimal Jerk trajectory tracking by MPC controller successfully opted subject to parametric uncertainties and external disturbance. Trajectory generation was adaptive based on both observers' results with fuzzy velocity control to estimate the landing time to bio-inspired an experienced human pilot. Subsequently, the trajectory is fed to the MPC for tracking, which considers the uncertainty of the model and disturbance. Comparison with PID proved the robustness and performance of the MPC tracking.

Robust position and attitude trajectory tracking challenge within disturbance and uncertainties for a quadrotor drone was investigated by exploiting the strong adaptive approximation capabilities of the RBFNN with an EKF observer, an outer loop adaptive NN compensated the external bounded disturbance where a supervised NN based control combined with IBS stabilizes the attitude in the inner loop compensating the unmodeled uncertainties and canceling the static error, the Supervised NN-based control avoids the initial attitude instability that may occur during parameters convergence of direct RBFNN control. Parameters update and convergence were analyzed and optimized using Lyapunov theory for the compensation control and via the optimized gradient descent algorithm for the Supervised control. A main feature of the proposed scheme is that the resulting closed-loop control system with the observer is guaranteed to be stable, similarly, the stability analyses of both NN control loops in presence of bounded disturbance and uncertainty were conducted and demonstrated using Lyapunov theory. Another major property of the proposed compensating controller is that this RBFNN-based controller is an add-on device that can be added directly to many other quadrotor control models.

The effectiveness of the proposed RBFNN compensation control scheme for robust tracking of trajectory has been validated for the quadrotor class of UAVs. A comparison with three other algorithms including an offline MLP showed better disturbance rejection and superior uncertainty compensation with faster correction.

Several applications of the quadrotor's autonomous control were developed, Autonomous soft landing on a moving platform that is detected and localized by an onboard camera was realized. The relative pose to the landing pad was based on the detection of the specially designed pad by the combined ArUco markers for best detection at different altitudes and angles. Identification of each marker and estimation of the pose was performed via Open-CV/python algorithm enhanced by stabilization filters. Relative pose estimation was satisfactory within 4 cm of accuracy.

The spline path for way points was first calculated then time referenced considering the velocity of the drone. obstacle avoidance has been investigated, static obstacles were avoided by a generated 3D Spline trajectory. A generation of trajectory can be done adaptively in case of dynamic obstacles, the direct optimal trajectory generation can be very expensive in a matter of computational time.

All simulations and codes are available in the GitHub links.

——————————-

7.2 Future works

Future works will rely on actual results to develop an online-autonomous trajectory planner based on a vision system. A Triple Module Redundancy (TMR) architecture could be used via a voting system to enhance the accuracy of sensors and ignore faulty readings.

Alternatively, intelligent control with completely model-free control is suggested, where no prior knowledge of the system's parameters or dynamics is required.

The challenge will be to build an optimal platform for quadrotors to execute intelligent tasks besides autonomous control in either individual or swarm mission scenarios. Artificial intelligence methods will be soon investigated on the built quadrotor model for more autonomous applications.

Bibliography

[1] UH AG. Neo-6: U-blox 6 gps modules data sheet, 2018.

[2] Vandana Agarwal and Surekha Bhanot. Radial basis function neural network-based face recognition using firefly algorithm. *Neural Computing and Applications*, 30(8):2643–2660, 2018.

[3] Carlos Aguilar-Ibanez, Miguel S Suarez-Castanon, Octavio Gutierrez-Frias, Jose de Jesus Rubio, and Jesus A Meda-Campana. A robust control strategy for landing an unmanned aerial vehicle on a vertically moving platform. *Complexity*, 2020, 2020.

[4] Faraz Ahmad, Pushpendra Kumar, Anamika Bhandari, and Pravin P Patil. Simulation of the quadcopter dynamics with lqr based control. *Materials Today: Proceedings*, 24:326–332, 2020.

[5] FVR-90 AIRFRAME. Designed for long endurance, 12 to 18 hours, with an 8 to 22 pound payload capacity., 2021.

[6] Erdinc Altug, James P Ostrowski, and Robert Mahony. Control of a quadrotor helicopter using visual feedback. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, volume 1, pages 72–77. IEEE, 2002.

[7] Boris Andrievsky, Nikolay V Kuznetsov, Olga A Kuznetsova, and Gennady A Leonov. Switching algorithm for data fusion of small low-cost uav navigation system. *IFAC Proceedings Volumes*, 46(30):206–211, 2013.

[8] Segun O Ariyibi and Ozan Tekinalp. Quaternion-based nonlinear attitude control of quadrotor formations carrying a slung load. *Aerospace Science and Technology*, 105:105995, 2020.

[9] Jan Bacik, Frantisek Durovsky, Pavol Fedor, and Daniela Perdukova. Autonomous flying with quadrocopter using fuzzy control and aruco markers. *Intelligent Service Robotics*, 10(3):185–194, 2017.

[10] Steven Bellens, Joris De Schutter, and Herman Bruyninckx. A hybrid pose/wrench control framework for quadrotor helicopters. In *2012 IEEE International Conference on Robotics and Automation*, pages 2269–2274. IEEE, 2012.

[11] Hamza Benzerrouk, Alexander Nebylov, and Hassen Salhi. Quadrotor uav state estimation based on high-degree cubature kalman filter. *IFAC-PapersOnLine*, 49(17):349–354, 2016.

[12] Mahathi Bhargavapuri, Soumya Ranjan Sahoo, Mangal Kothari, et al. Robust nonlinear control of a variable-pitch quadrotor with the flip maneuver. *Control Engineering Practice*, 87:26–42, 2019.

[13] Mahdis Bisheban and Taeyoung Lee. Geometric adaptive control for a quadrotor uav with wind disturbance rejection. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 2816–2821. IEEE, 2018.

[14] Martin Bohner and Nick Wintz. The linear quadratic tracker on time scales. *International Journal of Dynamical Systems and Differential Equations*, 3(4):423–447, 2011.

[15] Aseem V Borkar, Swaroop Hangal, Hemendra Arya, Arpita Sinha, and Leena Vachhani. Reconfigurable formations of quadrotors on lissajous curves for surveillance applications. *European Journal of Control*, 2020.

[16] Samir Bouabdallah, Pierpaolo Murrieri, and Roland Siegwart. Towards autonomous indoor micro vtol. *Autonomous robots*, 18(2):171–183, 2005.

[17] Samir Bouabdallah, Andre Noth, and Roland Siegwart. Pid vs lq control techniques applied to an indoor micro quadrotor. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, volume 3, pages 2451–2456. IEEE, 2004.

[18] Samir Bouabdallah and Roland Siegwart. Backstepping and sliding-mode techniques applied to an indoor micro quadrotor. In *Proceedings of the 2005 IEEE international conference on robotics and automation*, pages 2247–2252. IEEE, 2005.

[19] Oussama Bouaiss, Raihane Mechgoug, and Riadh Ajgou. Modeling, control and simulation of quadrotor uav. In *020 1st International Conference on Communications, Control Systems and Signal Processing (CCSSP)*, pages 340–345. IEEE, 2020.

[20] Oussama Bouaiss, Raihane Mechgoug, and Abdelmalik Taleb-Ahmed. Visual soft landing of an autonomous quadrotor on a moving pad using a combined fuzzy velocity control with model predictive control. *Signal, Image and Video Processing*, pages 1–10, 2022.

[21] David Cabecinhas, Rita Cunha, and Carlos Silvestre. A trajectory tracking control law for a quadrotor with slung load. *Automatica*, 106:384–389, 2019.

[22] Wenjing Cai, Jinhua She, Min Wu, and Yasuhiro Ohyama. Quadrotor waypoint-tracking control under exogenous disturbances based on equivalent-input-disturbance approach. *Journal of the Franklin Institute*, 2020.

[23] Zhihao Cai, Jiang Lou, Jiang Zhao, Kun Wu, Ningjun Liu, and Ying Xun Wang. Quadrotor trajectory tracking and obstacle avoidance by chaotic grey wolf optimization-based active disturbance rejection control. *Mechanical Systems and Signal Processing*, 128:636–654, 2019.

[24] Mou Chen, Shuzhi Sam Ge, and Bernard Voon Ee How. Robust adaptive neural network control for a class of uncertain mimo nonlinear systems with input nonlinearities. *IEEE Transactions on Neural Networks*, 21(5):796–812, 2010.

[25] G. d. Bothezat. Helicopter, u.s. patent 1573228, 1926.

[26] Xi Dai, Yuxin Mao, Tianpeng Huang, Na Qin, Deqing Huang, and Yanan Li. Automatic obstacle avoidance of quadrotor uav via cnn-based learning. *Neurocomputing*, 2020.

[27] Nihal Dalwadi, Dipankar Deb, and SM Muyeen. Observer based rotor failure compensation for biplane quadrotor with slung load. *Ain Shams Engineering Journal*, 13(6):101748, 2022.

[28] Li Danjun, Zhou Yan, Shi Zongying, and Lu Geng. Autonomous landing of quadrotor based on ground effect modelling. In *2015 34th Chinese Control Conference (CCC)*, pages 5647–5652. IEEE, 2015.

[29] Hemjyoti Das. Dynamic inversion control of quadrotor with a suspended load. *IFAC-PapersOnLine*, 51(1):172–177, 2018.

[30] Hemjyoti Das, Kaustubh Sridhar, and Radhakant Padhi. Bio-inspired landing of quadrotor using improved state estimation. *IFAC-PapersOnLine*, 51(1):462–467, 2018.

[31] C. Dillow. Get ready for drone nation, 2014.

[32] Carlos Alphonso F Ezequiel, Matthew Cua, Nathaniel C Libatique, Gregory L Tangonan, Raphael Alampay, Rollyn T Labuguen, Chrisandro M Favila, Jaime Luis E Honrado, Vinni Canos, Charles Devaney, et al. Uav aerial imaging applications for post-disaster assessment, environmental management and infrastructure development. In *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 274–283. IEEE, 2014.

[33] Sílvia Faria, José Lima, and Paulo Costa. Sensor fusion for mobile robot localization using extended kalman filter, uwb tof and aruco markers. In *International Conference on Optimization, Learning Algorithms and Applications*, pages 235–250. Springer, 2021.

[34] Kanishke Gamagedara, Taeyoung Lee, and Murray Snyder. Quadrotor state estimation with imu and delayed real-time kinematic gps. *IEEE Transactions on Aerospace and Electronic Systems*, 57(5):2661–2673, 2021.

[35] Benke Gao, Yan-Jun Liu, and Lei Liu. Adaptive neural fault-tolerant control of a quadrotor uav via fast terminal sliding mode. *Aerospace Science and Technology*, page 107818, 2022.

[36] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Manuel Jesús Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280–2292, 2014.

[37] Alvika Gautam, PB Sujit, and Srikanth Saripalli. Autonomous quadrotor landing using vision and pursuit guidance. *IFAC-PapersOnLine*, 50(1):10501–10506, 2017.

[38] Tryphon T Georgiou and Anders Lindquist. The separation principle in stochastic control, redux. *IEEE Transactions on Automatic Control*, 58(10):2481–2494, 2013.

[39] David Glade. Unmanned aerial vehicles: Implications for military operations. Technical report, Air Univ Press Maxwell Afb Al, 2000.

[40] Payam Shafiei Gohari, Hossein Mohammadi, and Sajjad Taghvaei. Using chaotic maps for 3d boundary surveillance by quadrotor robot. *Applied Soft Computing*, 76:68–77, 2019.

[41] Kexin Guo, Jindou Jia, Xiang Yu, Lei Guo, and Lihua Xie. Multiple observers based anti-disturbance control for a quadrotor uav against payload and wind disturbances. *Control Engineering Practice*, 102:104560, 2020.

[42] Jaime Rubio Hervas, Mahmut Reyhanoglu, and Hui Tang. Automatic landing control of unmanned aerial vehicles on moving platforms. In *2014 IEEE 23rd international symposium on industrial electronics (ISIE)*, pages 69–74. IEEE, 2014.

[43] Botao Hu, Lu Lu, and Sandipan Mishra. Fast, safe and precise landing of a quadrotor on an oscillating platform. In *2015 American Control Conference (ACC)*, pages 3836–3841. IEEE, 2015.

[44] Botao Hu and Sandipan Mishra. A time-optimal trajectory generation algorithm for quadrotor landing onto a moving platform. In *2017 American Control Conference (ACC)*, pages 4183–4188. IEEE, 2017.

[45] Huan Hu and Qing-ling Wang. Proximal policy optimization with an integral compensator for quadrotor control. *Frontiers of Information Technology & Electronic Engineering*, 21(5):777–795, 2020.

[46] Yifan Hu and Wenhui Liu. Fuzzy adaptive event-triggered control for a class of uncertain nonlinear systems subject to actuator dead-zone. In *2022 China Automation Congress (CAC)*, pages 4527–4532. IEEE, 2022.

[47] Youfang Huang, Wen Liu, Bo Li, Yongsheng Yang, and Bing Xiao. Finite-time formation tracking control with collision avoidance for quadrotor uavs. *Journal of the Franklin Institute*, 2020.

[48] Zutao Jiang, Jihua Zhu, Zhiyang Lin, Zhongyu Li, and Rui Guo. 3d mapping of outdoor environments by scan matching and motion averaging. *Neurocomputing*, 372:17–32, 2020.

[49] Xiao-Zheng Jin, Tao He, Xiao-Ming Wu, Hai Wang, and Jing Chi. Robust adaptive neural network-based compensation control of a class of quadrotor aircrafts. *Journal of the Franklin Institute*, 357(17):12241–12263, 2020.

[50] Aziz Kaba and Emre Kıyak. Optimizing a kalman filter with an evolutionary algorithm for nonlinear quadrotor attitude dynamics. *Journal of Computational Science*, 39:101051, 2020.

[51] Ahmed Khalifa and Mohamed Fanni. Experimental implementation of a new non-redundant 6-dof quadrotor manipulation system. *ISA Transactions*, 2020.

[52] Hassan K Khalil and Jessy W Grizzle. *Nonlinear systems*, volume 3. Prentice hall Upper Saddle River, NJ, 2002.

[53] Do Wan Kim. Fuzzy model-based control of a quadrotor. *Fuzzy Sets and Systems*, 371:136–147, 2019.

[54] Jinho Kim, S Andrew Gadsden, and Stephen A Wilkerson. A comprehensive survey of control strategies for autonomous quadrotors. *Canadian Journal of Electrical and Computer Engineering*, 43(1):3–16, 2019.

[55] N Koksal, H An, and B Fidan. Backstepping-based adaptive control of a quadrotor uav with guaranteed tracking performance. *ISA transactions*, 105:98–110, 2020.

[56] Kibeom Lee, Seungmin Jeon, Heegwon Kim, and Dongsuk Kum. Optimal path tracking control of autonomous vehicle: Adaptive full-state linear quadratic gaussian (lqg) control. *IEEE Access*, 7:109120–109133, 2019.

[57] Taeyoung Lee, Melvin Leok, and N Harris McClamroch. Nonlinear robust tracking control of a quadrotor uav on se (3). *Asian Journal of Control*, 15(2):391–408, 2013.

[58] Boxuan Li, Jiezhang Wu, Xiaojun Tan, and Benfei Wang. Aruco marker detection under occlusion using convolutional neural network. In *2020 5th International Conference on Automation, Control and Robotics Engineering (CACRE)*, pages 706–711. IEEE, 2020.

[59] Shushuai Li, Yaonan Wang, Jianhao Tan, and Yan Zheng. Adaptive rbfnns/integral sliding mode control for a quadrotor aircraft. *Neurocomputing*, 216:126–134, 2016.

[60] Hao Liu, Yafei Lyu, and Wanbing Zhao. Robust visual servoing formation tracking control for quadrotor uav team. *Aerospace Science and Technology*, 106:106061, 2020.

[61] Hao Liu, Yu Tian, Frank L Lewis, Yan Wan, and Kimon P Valavanis. Robust formation tracking control for multiple quadrotors under aggressive maneuvers. *Automatica*, 105:179–185, 2019.

[62] Yuyi Liu, Sujit Rajappa, Jan Maximilian Montenbruck, Paolo Stegagno, Heinrich Bülthoff, Frank Allgöwer, and Andreas Zell. Robust nonlinear control approach to nontrivial maneuvers and obstacle avoidance for quadrotor uav under disturbances. *Robotics and Autonomous Systems*, 98:317–332, 2017.

[63] Jakob Löber. *Optimal trajectory tracking of nonlinear dynamical systems*. Springer, 2016.

[64] Ricardo López-Gutiérrez, Abraham Efraim Rodriguez-Mata, Sergio Salazar, Ivan González-Hernández, and Rogelio Lozano. Robust quadrotor control: attitude and altitude real-time results. *Journal of Intelligent & Robotic Systems*, 88(2-4):299–312, 2017.

[65] David H Lyon. A military perspective on small unmanned aerial vehicles. *IEEE Instrumentation & Measurement Magazine*, 7(3):27–31, 2004.

[66] Sebastian Madgwick. An efficient orientation filter for inertial and inertial/magnetic sensor arrays. *Report x-io and University of Bristol (UK)*, 25:113–118, 2010.

[67] Luis A Márquez-Vega, Mario Aguilera-Ruiz, and Luis M Torres-Treviño. Multi-objective optimization of a quadrotor flock performing target zone search. *Swarm and Evolutionary Computation*, 60:100733, 2021.

[68] Luís Martins, Carlos Cardeira, and Paulo Oliveira. Linear quadratic regulator for trajectory tracking of a quadrotor. *IFAC-PapersOnLine*, 52(12):176–181, 2019.

[69] Daniel Mellinger and Vijay Kumar. Minimum snap trajectory generation and control for quadrotors. In *2011 IEEE international conference on robotics and automation*, pages 2520–2525. IEEE, 2011.

[70] Victor RF Miranda, Adriano Rezende, Thiago L Rocha, Héctor Azpúrua, Luciano CA Pimenta, and Gustavo M Freitas. Autonomous navigation system for a delivery drone. *arXiv preprint arXiv:2106.08878*, 2021.

[71] Roger Miranda-Colorado and Luis T Aguilar. Robust pid control of quadrotors with power reduction analysis. *ISA transactions*, 98:47–62, 2020.

[72] Hongwei Mo and Ghulam Farid. Nonlinear and adaptive intelligent control techniques for quadrotor uav–a survey. *Asian Journal of Control*, 21(2):989–1008, 2019.

[73] Nagarajapandian Mohan. Iterative learning control design for a non-linear multivariable system. *Journal of Control Engineering and Applied Informatics*, 23(2):32–39, 2021.

[74] Kartik Mohta, S Liu, Michael Watterson, Giuseppe Loianno, and Vijay Kumar. Estimation, control, and planning for high speed flight with a quadrotor using on-board sensing. *Journal of Field Robotics*, 2017.

[75] Eugene A Morelli. Estimating noise characteristics from flight test data using optimal fourier smoothing. *Journal of Aircraft*, 32(4):689–695, 1995.

[76] Sumaila Musa. Techniques for quadcopter modeling and design: A review. *Journal of Unmanned System Technology*, 5(3):66–75, 2018.

[77] Aradhana Nayak, Ravi N Banavar, and DHS Maithripala. Stabilizing a spherical pendulum on a quadrotor. *Asian Journal of Control*, 2020.

[78] Yul Y Nazaruddin, Angela Dian Andrini, and Boby Anditio. Pso based pid controller for quadrotor with virtual sensor. *IFAC-PapersOnLine*, 51(4):358–363, 2018.

[79] Yakoub Nettari, Serkan Kurt, and Moussa Labbadi. Adaptive robust control based on backstepping sliding mode techniques for quadrotor uav under external disturbances. *IFAC-PapersOnLine*, 55(12):252–257, 2022.

[80] Jongho Park and Hoki Baek. Stereo vision based obstacle collision avoidance for a quadrotor using ellipsoidal bounding box and hierarchical clustering. *Aerospace Science and Technology*, page 105882, 2020.

[81] Ricardo Pérez-Alcocer and Javier Moreno-Valenzuela. A novel lyapunov-based trajectory tracking controller for a quadrotor: Experimental analysis by using two motion tasks. *Mechatronics*, 61:58–68, 2019.

[82] Chen-Huan Pi, Kai-Chun Hu, Stone Cheng, and I-Chen Wu. Low-level autonomous control and tracking of quadrotor using reinforcement learning. *Control Engineering Practice*, 95:104222, 2020.

[83] Yuhua Qi, Jiaqi Jiang, Jin Wu, Jianan Wang, Chunyan Wang, and Jiayuan Shan. Autonomous landing solution of low-cost quadrotor on a moving platform. *Robotics and Autonomous Systems*, 119:64–76, 2019.

[84] Yuegui Feng Qianfei Zhou, Shuqing Ding. Corrosion inspection and evaluation of crane metal structure based on uav vision. *Signal, Image and Video Processing*, 16(1):5, 2022.

[85] Magnus JE Richardson and Tamar Flash. Comparing smooth arm movements with the two-thirds power law and the related segmented-control hypothesis. *Journal of neuroscience*, 22(18):8201–8211, 2002.

[86] Gerald Schweighofer and Axel Pinz. Robust pose estimation from a planar target. *IEEE transactions on pattern analysis and machine intelligence*, 28(12):2024–2030, 2006.

[87] Jeff S Shamma and James R Cloutier. Gain-scheduled missile autopilot design using linear parameter varying transformations. *Journal of guidance, Control, and dynamics*, 16(2):256–263, 1993.

[88] Yoonghyun Shin. *Neural network based adaptive control for nonlinear dynamic regimes*. Georgia Institute of Technology, 2005.

[89] Chinari Subhechha Subudhi and D Ezhilarasi. Modeling and trajectory tracking with cascaded pd controller for quadrotor. *Procedia computer science*, 133:952–959, 2018.

[90] Manuel Alejandro Vallejo-Alarcon. Robust backstepping control for highly demanding quadrotor flight. *Journal of Control Engineering and Applied Informatics*, 22(1):51–62, 2020.

[91] LG Van Willigenburg and Willem L De Koning. Numerical algorithms and issues concerning the discrete-time optimal projection equations. *European Journal of Control*, 6(1):93–110, 2000.

[92] Anurag Sai Vempati, Vipul Choudhary, and Laxmidhar Behera. Quadrotor: design, control and vision based localization. *IFAC Proceedings Volumes*, 47(1):1104–1110, 2014.

[93] Kummar Vijay. Unmaned aerial vehicles, 2017.

[94] Øystein Volden, Annette Stahl, and Thor I Fossen. Vision-based positioning system for auto-docking of unmanned surface vehicles (usvs). *International Journal of Intelligent Robotics and Applications*, pages 1–18, 2021.

[95] Ray Walters. Real life “constructicon” quadcopter robots being developed, 2011.

[96] Li Wang, Zheng Zhang, and Ping Sun. Quaternion-based kalman filter for ahrs using an adaptive-step gradient descent algorithm. *International Journal of Advanced Robotic Systems*, 12(9):131, 2015.

[97] Rui Wang and Jinkun Liu. Trajectory tracking control of a 6-dof quadrotor uav with input saturation via backstepping. *Journal of the Franklin Institute*, 355(7):3288–3309, 2018.

[98] Jan C Willems. Linear robust control: By m. green and djn limebeer, prentice-hall (1995). isbn 0-13-102278-4, 1995.

[99] Samuel Wilson, Henry Eberle, Yoshikatsu Hayashi, Sebastian OH Madgwick, Alison McGregor, Xingjian Jing, and Ravi Vaidyanathan. Formulation of a new gradient descent marg orientation algorithm: Case study on robot teleoperation. *Mechanical Systems and Signal Processing*, 130:183–200, 2019.

[100] Philippe Martin Wyder, Yan-Song Chen, Adrian J Lasrado, Rafael J Pelles, Robert Kwiatkowski, Edith OA Comas, Richard Kennedy, Arjun Mangla, Zixi Huang, Xiaotian Hu, et al. Autonomous drone hunter operating by deep learning and all-onboard computations in gps-denied environments. *PloS one*, 14(11):e0225092, 2019.

[101] Lei Xing and Wujiao Dai. A local feature extraction method for uav-based image registration based on virtual line descriptors. *Signal, Image and Video Processing*, 15(4):705–713, 2021.

[102] Jing-Jing Xiong and En-Hui Zheng. Optimal kalman filter for state estimation of a quadrotor uav. *Optik*, 126(21):2862–2868, 2015.

[103] Qingzheng Xu, Zhisheng Wang, and Ziyang Zhen. Information fusion estimation-based path following control of quadrotor uavs subjected to gaussian random disturbance. *ISA transactions*, 99:84–94, 2020.

[104] Yuan Xu, Yuriy S Shmaliy, Xiyuan Chen, Yueyang Li, and Wanfeng Ma. Robust inertial navigation system/ultra wide band integrated indoor quadrotor localization employing adaptive interacting multiple model-unbiased finite impulse response/kalman filter estimator. *Aerospace Science and Technology*, 98:105683, 2020.

[105] Zhe-Cheng Xu, Bin-Bin Hu, Bin Liu, XD Wang, and Hai-Tao Zhang. Vision-based autonomous landing of unmanned aerial vehicle on a motional unmanned surface vessel. In *2020 39th Chinese Control Conference (CCC)*, pages 6845–6850. IEEE, 2020.

[106] Warren R Young. *The Helicopters: The Epic of Flight.* Time-Life Books, 1982.

[107] Gan Yu, David Cabecinhas, Rita Cunha, and Carlos Silvestre. Quadrotor trajectory generation and tracking for aggressive maneuvers with attitude constraints. *IFAC-PapersOnLine*, 52(12):55–60, 2019.

[108] Lei Yu, Shumin Fei, and Xun Li. Robust adaptive neural tracking control for a class of switched affine nonlinear systems. *Neurocomputing*, 73(10-12):2274–2279, 2010.

[109] Yinan Yu, Pan Tang, Tao Song, and Defu Lin. A two-step method for system identification of low-cost quadrotor. *Aerosp. Sci. Technol.*, 96:105551, 2020.

[110] Yiping Yue, Hongjiu Yang, Fucai Liu, and Huaigang Zang. Cooperative control for multiple quadrotors under position deviations and aerodynamic drag. *Mechanical Systems and Signal Processing*, 147:107096, 2020.

[111] Juqian Zhang, Dawei Gu, Zhaohui Ren, and Bangchun Wen. Robust trajectory tracking controller for quadrotor helicopter based on a novel composite control scheme. *Aerospace Science and Technology*, 85:199–215, 2019.

[112] Zhiye Zhao and Xiaozheng Jin. Adaptive neural network-based sliding mode tracking control for agricultural quadrotor with variable payload. *Computers and Electrical Engineering*, 103:108336, 2022.

[113] Yao Zou and Bing Zhu. Adaptive trajectory tracking controller for quadrotor systems subject to parametric uncertainties. *Journal of the Franklin Institute*, 354(15):6724–6746, 2017.