DESIGN, FABRICATION, AND SOFT IMPACT MODELING AND SIMULATION OF A COLLISION-RESILIENT FOLDABLE MICRO QUADCOPTER

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF

MASTER OF SCIENCE

IN

MECHANICAL ENGINEERING

By

Amirali Abazari

September 2022

DESIGN, FABRICATION, AND SOFT IMPACT MODELING AND SIMULATION OF A COLLISION-RESILIENT FOLDABLE MICRO QUADCOPTER

By Amirali Abazari

September 2022

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Onur Özcan(Advisor)

Yıldıray Yıldız

Murat Efe

Approved for the Graduate School of Engineering and Science:

Orhan Arıkan
Director of the Graduate School

ABSTRACT

DESIGN, FABRICATION, AND SOFT IMPACT MODELING AND SIMULATION OF A COLLISION-RESILIENT FOLDABLE MICRO QUADCOPTER

Amirali Abazari
M.S. in Mechanical Engineering
Advisor: Onur Özcan
September 2022

Despite the appreciable advancements in mobile robot navigation and obstacle avoidance algorithms using an abundance of sensors and sensor fusion methods, the navigation of moving robots through confined and cluttered spaces is still a great challenge. The physical interaction and collision between mobile robots and surrounding obstacles in these environments are unavoidable. This becomes more concerning for the case of **V**ertical **T**ake-**O**ff and **L**anding (VTOL) **U**nmanned **A**erial **V**ehicles (UAV) that the system is naturally unstable, and a minor fault or disturbance may result in severe crashes.

In this thesis, a Collision-Resilient Quad-rotor **M**icro **A**erial **V**ehicle (MAV) is designed using the Origami-inspired design fabrication techniques. The quadcopter is lightweight ($220g$ max) and is designed for outdoor inspection and surveillance missions. This compliant drone provides an stable flight for a duration of $5 - 10\ min$, depends on the flight condition.

A dynamic model is derived for the quadcopter to represent the realistic features designed for this specific UAV. In addition to that, the impact of the compliant body to surrounding obstacles is modeled as visco-elastic contact force and is added to the quadcopter's dynamic model. The contact dynamic friction force between the protective soft bumpers and the surface is also modeled.

The developed dynamic model is then used to simulate the impact of the collision-resilient quadcopter in two different simulation environments; MATLAB *Simulink* and ROS *Gazebo*. A cascaded PID control scheme is suggested for low-level

(attitude) and high-level (global position) control of the drone in experiments and simulation. The result of these soft impact simulations closely imitate the collision-resilient properties of the actual quadcopter in experiments. Coefficient of Restitution (CoR) for the compliant drone impact, both in simulations and experiments, is in the interval [0.5 0.6]. This shows a great capacity for the drone to dampen the collisions.

Keywords: Quadrotor, Collision-Resilient, Foldable, UAV, VTOL, MAV, Soft Impact Dynamics, Soft Impact Simulation.

ÖZET

ÇARPIMAYA DAYANIKLI, MİKRO VE DÖRT PERVANELİ HELİKOPTER ROBOTLARIN TASARIMI, ÜRETİMİ, YUMUŞAK DARBE MODELLEMESİ VE SİMÜLASYONU

Amirali Abazari
Makine Mühendisliği, Yüksek Lisans
Tez Danışmanı: Onur Özcan
Eylül 2022

Çok sayıda sensör ve sensör birleştirme yöntemi kullanan mobil robot navigasyonu ve engellerden kaçınma algoritmalarındaki kayda değer gelişmelere rağmen, hareketli robotların kapalı ve dağınık alanlarda navigasyonu hala büyük bir zorluktur. Bu robotların etrafındaki objelere çarpması ve bu nesneler ile fiziksel etkileşimde bulunması kaçınılmazdır. Bu durum dikey iniş-kalkış yapabilen insansız hava araçları gibi doğal olarak dengesiz olan araçlarda daha da önemli hale gelmekte ve ufak hatalar ve bozunumlar sistemde ciddi kazalara sebep olmaktadır.

Bu tezde çarpışmaya dayanıklı, mikro ve dört pervaneli helikopter robotun origamiden esinlenen tasarım ve üretiminden bahsedilmiştir. Oldukça hafif (en fazla 200 gram) olan bu robot dış mekanların denetleme ve gözetlenmesi amacıyla tasarlanmıştır. Bu etkilere dayanıklı hava aracı dış etkilere bağlı olarak $5 - 10$ dakika arasında kararlı bir uçuş gerçekleştirebilmektedir.

Bu özel insansız hava aracı için tasarlanmış gerçekçi özellikleri temsil etmek için bir dinamik model türetilmiştir. Buna ek olarak, yumuşak gövdenin çevredeki nesneler ile çarpışması viskoelastik temas kuvveti olarak modellenmiş ve robotun dinamik modeline eklenmiştir. Ayrıca, yumuşak tamponlar ile yüzey arasındaki temas dinamik sürtünme kuvveti modellenmiştir.

Geliştirilen dinamik model çarpışmaya dayanıklı robotun etkisini simüle etmek için iki farklı ortamda kullanılmıştır. Bunlar MATLAB Simulink ve ROS Gazebo ortamlarıdır. Basamaklı yapıya sahip PID kontrolcü hem düşük-seviye (yönelim) hem de yüksek-seviye (global posizyon) kontrolde simülasyonlar ve deneyler için

kullanılmıştır. Bu yumuşak darbe simülasyonlarının sonuçları deneyler ile uyumlu sonuçlar vermiştir. Bu uyumlu hava aracı için eski haline gelme katsayısı (CoR) hem deneylerde hem de simülasyonda [0.5 0.6] aralığında kalmıştır. Bu insansız hava aracının çarpışmaları sönümleyebildiğini ortaya koymaktadır.

Anahtar sözcükler: Çarpışmaya dayanıklı, Katlanabilir, İnsansı Hava Aracı, Dikey iniş-kalkış, Mikro Hava Aracı, Yumuşak Etki Dinamikleri, Yumuşak Etki Simülasyonu.

Acknowledgement

First and foremost, I would like to thank my supervisor, Dr. Onur Özcan, for his persistent guidance, endless support, and kindness. It has been a great pleasure for me to be a member of his research team, where we have him, first as our friend, then as our mentor. This thesis could not be possible without him and his unwavering support.

I would also like to acknowledge Dr. Yıldıray Yıldız, our collaborating advisor in the project, for all the help and feedback he provided me during this research.

It has been a blessing for me to have such great lab mates to share all the time and company with them. Nima, Mustafa, Mohammad, Mert Ali, Didem, Doĝa, Talip, Muhammed, and Levent; thank you all; you guys are the best! I want to especially thank Alihan Bakır, my teammate in this project, for all we have been through during this project.

I appreciate the work done by our undergrad lab members during this project. Ömer, Cankan, Orkun, Gökmen, Noman, and Osman; thanks for all your assistance and help.

I like to show my deepest gratitude to all my close friends in Bilkent, especially Mousa, Maryam, Elmira, and Büşra. Living in the soulless and silent Bilkent campus during the COVID19 pandemic was not possible without your friendship and company.

Last but not least, I want to thank my family. My parents, Farah and Shahram, my sister Tarlan, my grandma Sayyareh, and my aunt Farideh. You are the light in my eyes and the warmth in my heart. I am so lucky to have you in my life.

Finally, I would like to acknowledge the Scientific and Technological Research Council of Turkey, TÜBİTAK, for the financial support of this research under Grant No. 118E937.

Contents

1 Introduction **1**

1.1 Motivation and Objectives 1

1.2 Contributions 2

1.3 Structure of the Thesis 3

2 Design and Fabrication **5**

2.1 Collision-Resilient Multi-Rotor UAV in Literature 5

2.2 Design of the Collision-Resilient Foldable Body 6

2.2.1 Define Foldable 6

2.2.2 Material Selection and Fabrication Methods 7

2.2.3 Foldable Arm Design 8

2.2.4 Central Housing Design 10

2.2.5 Protective Bumper Design 11

2.2.6 Landing Gear Design 12

2.2.7 Battery Housing Design 13

2.2.8 First-Person-View (FPV) Camera Case Design 14

2.2.9 Intermediate Electronics Holder Board Design 15

2.2.10 MiniCore Vs. New Version 15

2.3 Component Selection 17

2.3.1 Main Processor 17

2.3.2 Flight Controller 17

2.3.3 Inertial Measurement Unit (IMU) 18

2.3.4 Altitude Sensor (Barometer) 19

2.3.5 Global Positioning System (GPS) and Compass (Magnetometer) 20

2.3.6 BLDC Motors and Propellers 21

2.3.7 Power Storage (Battery) 21

2.3.8 FPV Camera and Video Transmission 22

2.4 System Design 23

3 Quadcopter and Soft Impact Dynamics 26

3.1 Multi-Rotor VTOL Dynamics in Literature 26

3.2 Quadcopter Dynamics 27

3.2.1 Quadcopter Configuration 28

3.2.2 Coordinate System Definition and Kinematics 28

3.2.3 Equations of Motion (EoM) 30

3.3 Soft Impact Dynamics 33

3.3.1 Definition of Collision as Impact Force 33

3.3.2 Impact Detection 34

3.3.3 Determining the Penetration Vector 37

3.3.4 Deriving the Impact Forces from the Penetration Vector . 39

4 Control, Simulation, and Experiments 42

4.1 Compliant Multi-rotor MAV Control in Literature 42

4.2 Flight Control Strategy and Controller Design in Simulink 44

4.2.1 Plant Design 46

4.2.2 Sensor Modeling 46

4.2.3 Estimator Block 48

4.2.4 Cascaded PID Controllers 49

4.2.5 Mixing Algorithm Block 53

4.2.6 Flight Simulation 54

4.2.7 Adding the Soft Impact Block 57

4.2.8 Soft Impact Simulation 57

4.2.9 Discussion on the Ordinary Differential Equation (ODE) Solving 63

4.3 Soft Impact Simulation in Gazebo 64

4.4 CoR Comparison for the Simulations and Experiments 68

4.5 Collision-Resilient Foldable Body in Outdoor Field Experiments . 69

4.5.1 Protective Bumpers and Landing Gear 70

4.5.2 FPV Camera Streaming During the Inspection Flight . . . 71

5 Conclusion and Future Works **72**

A Obtaining the Linear Coefficients of Thrust and Drag **83**

B Plant Model - MATLAB Function in Simulink **88**

C Complementary Filter Code - Estimator Block in Simulink **90**

D State Estimation Methods **91**

D.1 Sensor Fusion using Complementary Filter 91

D.2 Sensor Fusion Madgwick Filter: 93

E Mixing Algorithm MATLAB Function Block Code **96**

F Collision Detection Block MATLAB Function Code **97**

List of Figures

2.1 a) Transparent PET sheets in different colors. b) A roll of Kapton® sheet. c) A roll of TPU 3D-printing filament 7

2.2 Design of the foldable arms with the illustration of folding sequence, mating holes and slots, and the folding buffer zones. . . . 8

2.3 A pair of mating foldable arms 9

2.4 Design of the central housing - illustration of the folding sequence, mating screw holes, and mounting slots 10

2.5 Design of the foldable protective bumpers - illustration of the folding sequence and mating slots 11

2.6 Protective foldable bumper's installation on the arms 12

2.7 Landing gear design and assembly 12

2.8 Battery housing design 13

2.9 Battery housing assembly under the foldable drone 14

2.10 FPV camera case design and its assembly 15

2.11 Assembly of the drone electronics on the intermediate holder board 16

2.12 *MiniCoRe* (right) and the new version of the collision-resilient foldable micro quadcopter (left) 16

2.13 ESP32-DevKit-V1 micro-processing unit 17

2.14 GepRC STABLE F411 flight controller 18

2.15 MPU6000 chipset (left), and MPU9250 IMU kit (right) 19

2.16 MS5611 barometric pressure sensor 19

2.17 NEO-M8 GPS+Compass module 20

2.18 *EMAX* RS1106 6000KV and AVAN MINI 3" motor-propeller set . 21

2.19 Two parallel 2S Li-Po batteries used to power the collision-resilient foldable drone 22

2.20 FPV camera (left), 5.8GHz video receiver (right) 22

2.21 System demonstration for the collision-resilient foldable quadcopter 24

2.22 Mass distribution of different components of the collision-resilient foldable quadcopter 25

3.1 Quadcopter configurations, coordinate definition, and rotor sequence; +-shaped (left) and X-shaped (right) 28

3.2 Illustration of the inertial and rotating reference frames on the quadcopter 29

3.3 Simplified geometry of the collision-resilient quadcopter in MATLAB 35

3.4 Soft impact represented as visco-elastic deformation (Δ_d) in the compliant body (right), and visco-elastic penetration (Δ_p) into the colliding surface (left) 37

3.5 Demonstration of the impact forces 41

4.1 Cascaded PID controller scheme used for low-level and high-level control of the collision-resilient foldable quadcopter– in experiment and simulation 45

4.2 Normal sensor data distribution 46

4.3 Sensor reading results for 3DOF accelerometer and 3DOF gyroscope 47

4.4 Modeling MPU9250 IMU sensor output in Simulink 47

4.5 Use of saturation blocks to limit the final output of the controllers (control inputs) 52

4.6 3D view of the tracked $10m \times 10m$ square path in the simulation environment (MATLAB Simulink) 54

4.7 $[X, Y, Z]$ view of the tracked $10m \times 10m$ square path in the simulation environment (MATLAB Simulink) 55

4.8 Soft impact detection algorithm added as the obstacle block to the simulation environment 57

4.9 Collision scenario designed in Simulink (the size of the quadcopter is exaggerated) 58

4.10 The movement of the compliant foldable quadcopter during the impact - purple area shows the penetration in the direction of impact 59

4.11 Velocity change of the CoG of the quadcopter (X-direction) during the soft impact - shaded area indicates the impact duration 60

4.12 CoR for different impact velocities for the simulations in *Simulink* 61

4.13 Velocity change during the collision for *MiniCoRe* and its rigid equivalent [2] 61

4.14 Approching pitch angle before the impact for $v_{in} = 9\,m/s$ (right), and $v_{in} = 5\,m/s$ (left) 62

4.15 *SolidWorks* model for the collision-resilient foldable quadcopter . 64

4.16 Simplified model imported as a URDF file in Gazebo 65

4.17 COR values for different impact velocities in Gazebo 66

4.18 Soft impact (into a fixed, rigid block) simulation for the compliant foldable quadcopter in Gazebo 67

4.19 Simulation failure due to backward folding of the arms and central housing contact to the block at $v_{in} = 5\,m/s$ in *Gazebo* 67

4.20 Comparison between the simulation results in MATLAB *Simulink*, ROS Gazebo, and experiments for *MiniCoRe* 68

4.21 Single collision-resilient foldable quadcopter during a field inspection mission 69

4.22 Three collision-resilient foldable quadcopters during a field inspection mission 69

4.23 Damaged collision-resilient foldable body after a $20m$ high free fall 70

4.24 FPV cam streaming from three collision-resilient foldable drones during an inspection mission 71

A.1 Actuation mechanism in collision-resilient micro quadcopter . . . 84

A.2 Lift (thrust) and drag forces definition on propeller 84

A.3 Setup for static lift and drag force coefficients experiment 84

A.4 Thrust force (Lift) test results for all motor-propeller combinations mentioned in table A.1 87

D.1 Experiment on the result of the complementary sensor using the accelerometer and the gyroscope of MPU9250 (the blue side of the arrow represents the top of the sensor and the red side represents the bottom part) 93

D.2 Madgwick Filter's schematic mathematical representation 94

D.3 Experiment on the result of the Madgwick filter using the MPU9250 MARG (the blue side of the arrow represents the top of the sensor and the arrow shows the same attitude at start (left) and end of the motion (right)) 95

List of Tables

3.1 Parameter and Constant values for collision-resilient foldable micro quadcopter's dynamic model 32

4.1 Noise amplitude and delay time values used in the simulation 48

4.2 PID gains for altitude (Z) controller 51

4.3 PID gains for angular rate ($\dot{\phi}; \dot{\theta}; \dot{\psi}$) controllers 51

4.4 PID gains for angular position ($\phi; \theta; \psi$) controllers 52

4.5 PID gains for global $[X, Y]$ position controllers 53

4.6 Visco-elastic and friction coefficients used in the simulations 59

4.7 Joint and surface properties used to simulate the soft impact in Gazebo 66

A.1 Motor and Propeller types used in the thrust and drag coefficient experiment 86

Chapter 1

Introduction

1.1 Motivation and Objectives

The recent developments in electrical motors and power storage technologies in the past two decades have resulted in rapid growth in the utilization of mobile robots in different applications. Modern technologies in electrical motors such as Direct Drive motors, Brush-Less DC (BLDC) motors, and Coreless motors provide a higher Power-to-Weight ratio making it possible to supply higher output torques with smaller and lighter rotors. New versions of commercial high-density power storage technologies like Lithium-ion batteries (Li-ion) also supply a specific power of up to 250 Wh/kg with relatively high output currents (discharge rate) compared to their earlier versions.

As a result, nowadays, mobile robots are capable of performing complex tasks fully autonomously, untethered to a power/control source. Further enhancements in the production of the smaller elements, including sensors, actuators, and power supplies, facilitate the transition of mobile robots to smaller scales where they can perform the same tasks with lower energy consumption rates and higher agility.

Employing the new advancements in technology, *Bilkent Miniature Robotics Lab*

(MRL) have developed many mesoscale miniature robots with compliant bodies. The primary objective of these robots is to provide access to hazardous confined environments for rescue and inspection purposes where human/animal presence is dangerous or impossible.

The Miniature Collision-Resilient Foldable Quadcopter project started in 2016. The earlier version of this Vertical Take-Off and Landing (VTOL) Unmanned Aerial Vehicle (UAV) was developed as a 50g X-shaped quadcopter [1]. *Mini-CoRe* was utilizing a commercial flight controller and was being controlled by a Radio Control (RC) device. The indoor collision experiments for this drone demonstrated considerable damping effects during impacts in comparison to an equivalent rigid drone of the same size and shape [2].

In the continuation of this project, as objectives of the present thesis, a new version of the collision-resilient foldable micro quadcopter is developed for outdoor inspection and surveillance applications. Remaining under the predefined criteria of the project (maximum dimensions of $30cm * 30cm$ and maximum weight of 300g), the new version of the collision-resilient foldable micro quadcopter maintains a stable fully-autonomous flight during its surveillance mission outdoors.

As another objective of this project, the compliant interaction of the quadcopter with the physical environment is modeled and added to the quadcopter's equations of motion. Later, this set of dynamic equations is solved to simulate different scenarios of soft impact between the compliant drone and a rigid surface (simulations are conducted in Simulink and Gazebo).

1.2 Contributions

While many impact-resilient bodies and structures are designed for VTOLs, few have included compliance into the multicopter's arms. In other words, the quadcopter developed in this project is one of the first multi-rotor UAVs that uses the compliance of the main body to dampen impacts. This demonstrates itself by

varying place of the rotors during the impact, unlike the rigid frames in which the relative distance between the rotors is the same (and equal to each other) at all times. In addition to the compliant frame, the quadcopter utilizes soft bumpers and landing gears to prevent any possible damage to the propellers and the battery during impacts and landings. It is good to mention that lowcost manufacturing process of the quadcopter's body is also unrepeated.

In the modeling and simulation part, the integration of the soft impact dynamics with the quadcopter's equations of motion is investigated for the first time. The derived dynamics are then solved in a single simulation scenario while an active controller is controlling the attitude and position of the quadcopter (both in Simulink and Gazebo).

1.3 Structure of the Thesis

This thesis is organized in the following arrangement:

Chapter 2 begins with a short introduction to collision-resilience in multi-rotor UAV design in the literature. After this intro, the fabrication steps and considerations for the collision-resilient foldable quadcopter's design are detailed. Later, it is explained how different components such as motors, propellers, etc., are selected to fit the predefined features of the system. Eventually, the system design for the drone and how different parts and components are connected related to each other and communicate with each other is explained.

In chapter 3, developed dynamic models for multi-rotor vehicles in literature are presented. Then a mathematical model representing the dynamics of the collision-resilient foldable quadcopter is derived for an X-shaped quadcopter. Eventually, soft impact dynamics are derived in this chapter. This section describes how the contact forces are generated during an impact and how they can be calculated and added to the system dynamics.

Chapter 4 consists of four sections. After a brief literature review of the VTOL UAV control problem, the recommended control scheme for the quadcopter's attitude and global position control is discussed. After that, the implementation of this control scheme in MATLAB *Simulink* is explained. Later the results for the soft impact scenarios conducted in simulations are represented. Soft impact simulation for the foldable drone in *Gazebo* is explained as an alternative dynamical simulation environment. Ultimately, the performance of the collision-resilient foldable body in field experiments is described and analyzed.

Chapter 2

Design and Fabrication

This chapter introduces the development of the collision-resilient foldable micro quadcopter for outdoor inspection and surveillance purposes based on the earlier model developed in [1]. After reviewing the previous works in literature, the design and fabrication process of the origami-inspired foldable body and its components are discussed. Later, selection of the mechanical, electrical, and electronic components of the UAV is presented in this chapter.

2.1 Collision-Resilient Multi-Rotor UAV in Literature

Since the early stages of multi-rotor drones' emergence in the 2000s, development of micro-size multicopters have been getting attention. To achieve this purpose, the first step was to investigate the governing physical rules applied to VTOL multi-rotors and derive the mathematical equations for them [3, 4]. Further improvements in the dynamic modeling and control of micro drones on one side, and overcoming the existing limitations in the hardware (actuators, micro-controllers, power storage, etc.) on the other side, gave a more vivid perspective of how the micro-drone world will change in the future [5].

With the recent proliferation of Micro Aerial Vehicles (MAV) in industry and commercial applications, the interaction between these vehicles and the physical environment/human has drawn the attention of both industry and academia. The outcomes of these investigations suggest a wide range of solutions for various applications, including the employment of peripheral structures and cage-like platforms [6, 7, 8], utilizing innovative origami-inspired protective bumpers [9, 10, 11, 12, 13], and benefiting from variable stiffness/softness in the design as an impact-resilient mechanism [2, 14, 15, 16, 17]. Active/passive change in drone configuration is also represented as an effective way of protecting MAVs during their mission in the literature [18, 19, 20, 21].

2.2 Design of the Collision-Resilient Foldable Body

As outlined previously, the new version of the collision-resilient foldable quadcopter is designed based on a primary version (*MiniCoRe*) developed in *Bilkent Miniature Robotics Lab*. This section elaborates on the design, modifications, and improvements made to the foldable body to make it prepared for its outdoor missions.

2.2.1 Define Foldable

It appears that it is necessary to define what the term **Foldable** denotes in this thesis. In the literature, the majority of the papers have indicated this term as a capability of the drone's arms to fold by rotating around a revolute joint to form different morphologies like in [19]. This is also the case for commercial drones, where folding arms are a measure of easy portability.

In this thesis, the term **foldable** shows that the body is manufactured by folding a single 2D sheet into a 3D structure using origami techniques - similar to [11].

2.2.2 Material Selection and Fabrication Methods

One of the objectives of this project is to keep the manufacturing costs of the compliant robot low by using readily available materials in the market and conventional manufacturing methods. $250\mu m$ PET (**P**oly**e**thylene **T**erephthalate) sheets are chosen for the manufacturing of the foldable parts. These sheets are easily available in a wide range of colors and widths at stationer's and print shops. The mechanical properties of the material also make it a good candidate for origami structures fabrication. A desktop laser cutter machine is used to cut the PET sheets and engrave the folding lines and origami patterns on them.

It is worth mentioning that the use of Kapton® Polyimide films is investigated as an alternative for PET sheets, but because of its relatively high yield strength and low plastic deformation range, it cannot be folded properly, and it is not good for origami manufacturing.

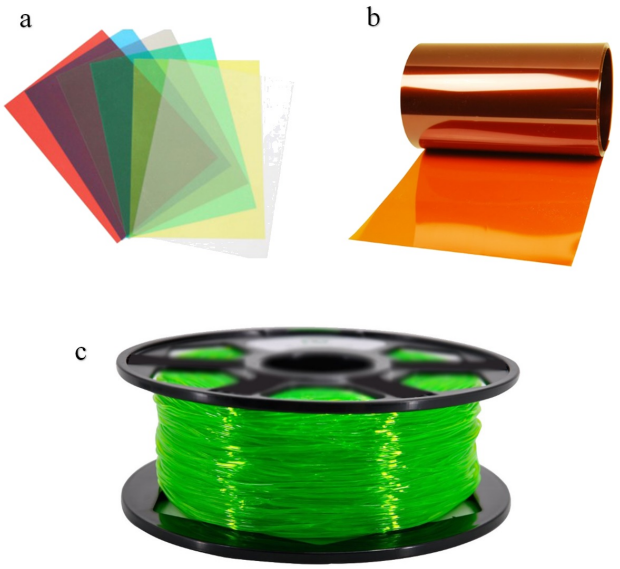

Figure 2.1: a) Transparent PET sheets in different colors. b) A roll of Kapton® sheet. c) A roll of TPU 3D-printing filament

An intermediate connector part, which holds to the main electronics of the quadcopter on one side and fastens to the drone arms on the other side, is fabricated

from TPU (**T**hermoplastic **P**oly**u**rethane) filaments using the 3D printing methods. During the test stages of the project, many rigid quadcopter bodies were also made of PLA (**P**oly**l**actic **A**cid) filaments using 3D printing.

2.2.3 Foldable Arm Design

The arms of the quadcopter are modified versions of the previous design for *MiniCoRe* [1]. The two-sided triangular structure of the arms provides a lower moment of inertia about the *z-axis*, making it possible for the arms to easily bend during the horizontal impacts, while the higher moment of inertia about the x and y axes prevents undesirable bending, twisting, and vibration. Figure 2.2 illustrates a pair of foldable arms and how they are cut from a single sheet.

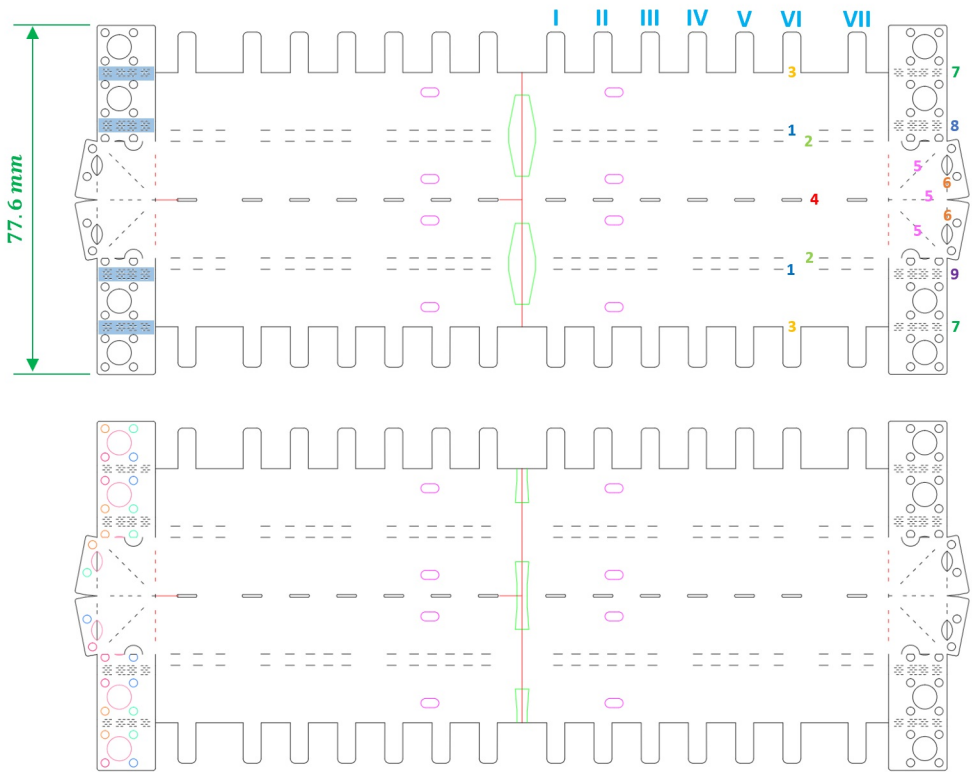

Figure 2.2: Design of the foldable arms with the illustration of folding sequence, mating holes and slots, and the folding buffer zones.

In the figure above, the numbers beside the dashed lines indicate the folding sequence for the origami structure. The mating slots are also demonstrated in the same colors in the middle of the arms. On the lower left part of the figure, the concentric holes after folding are also shown in the same colors.

On the top left part, the highlighted zones on the motor mounts depict the buffered folding zones. The reason for changing the whole design for the motor mounts in the new version and embedding a folding buffer zone instead of a single folding line is that by scaling up the arms, the previous motor mounts lose their rigidity and are not able to hold the motors. The new design, on the other hand, maintains the strength during the flight without any considerable vibrations or mechanical failures. The folding buffer zones also secure the alignment of the concentric screw holes.

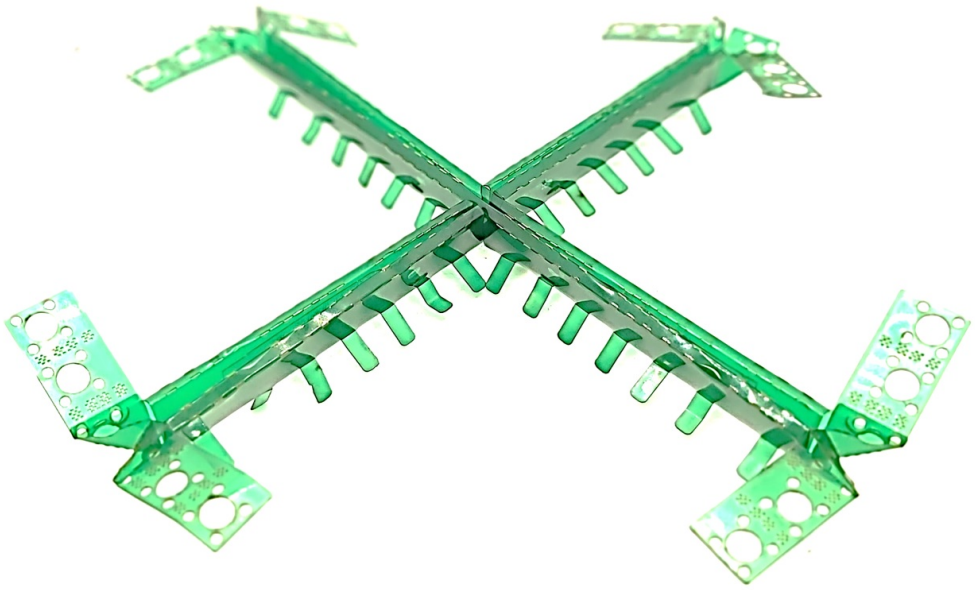

Figure 2.3: A pair of mating foldable arms

2.2.4 Central Housing Design

The central housing of the foldable quadcopter is responsible for maintaining the drone's structural integrity by holding the foldable arms together and providing space for the installation of drone electronics. Figure 2.4 illustrates the design of the central housing, its folding sequence, and the mating screw holes and slots (designated in the same colors).

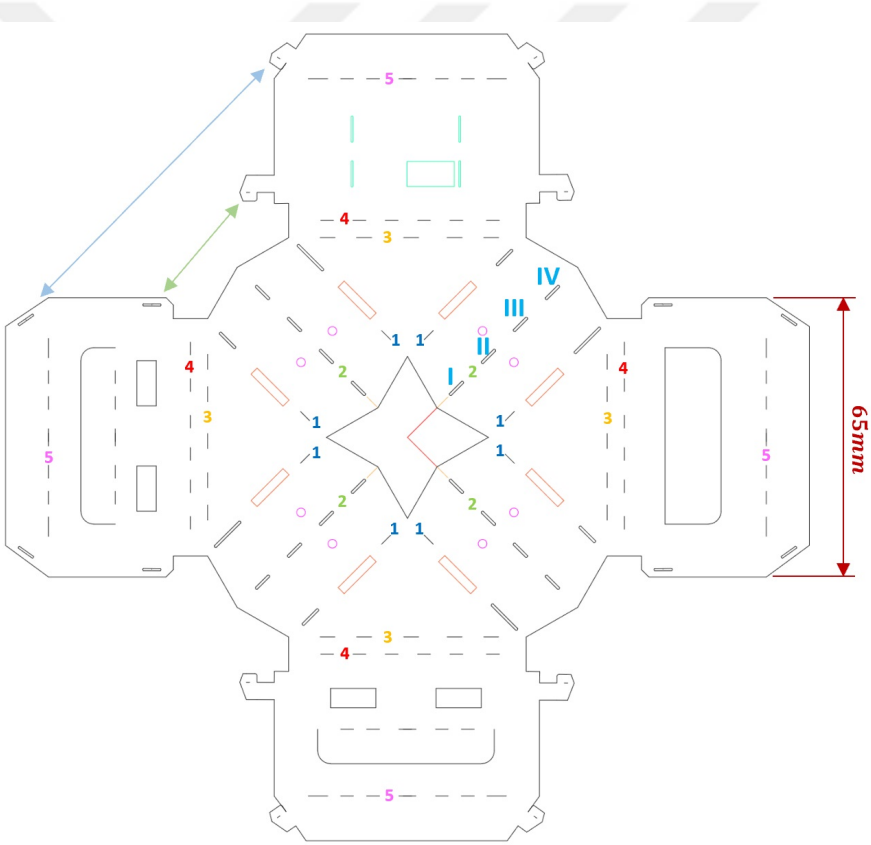

Figure 2.4: Design of the central housing - illustration of the folding sequence, mating screw holes, and mounting slots

As depicted in the figure above, the central housing is being installed on the locks **I** through **IV** (designated by Roman numbers in Figures 2.2 and 2.4) of all four arms.

2.2.5 Protective Bumper Design

The protective bumpers are initially installed on the motor mounts beneath the motor itself through the motor screws. After the installation of the motors, bumpers are tight-fastened to extension lock number **VII** of the arm they are installed on using four extension hands (as shown in Figure 2.5). The middle extension hands are added to increase the bumpers' stiffness and strengthen their weakest part.

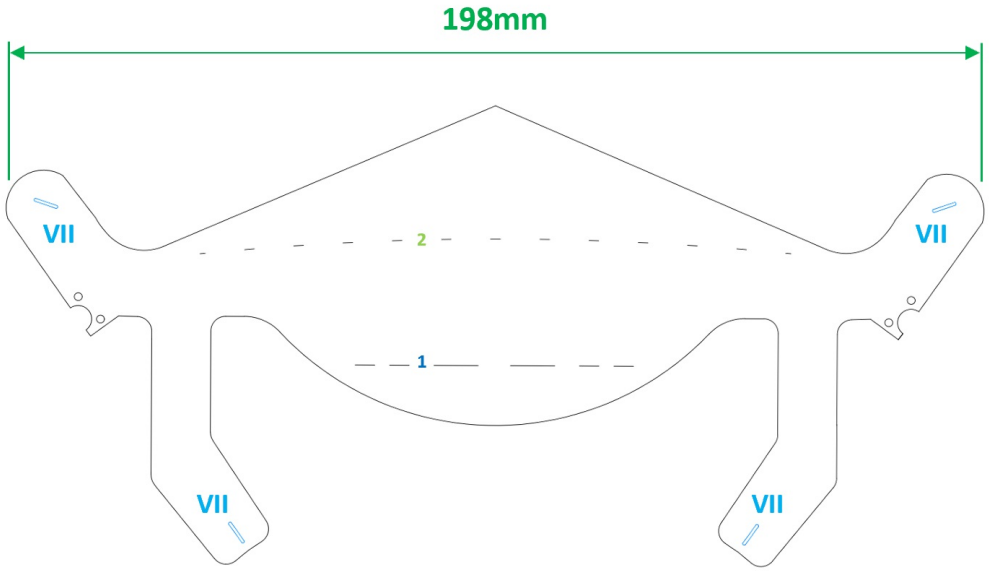

Figure 2.5: Design of the foldable protective bumpers - illustration of the folding sequence and mating slots

One of the fold lines of the protective bumper is curved with a curvature radius of 600mm. This curvature helps the bumper not to buckle during installation. Figure 2.6 demonstrates an installed bumper from different angles before the installation of the motor.

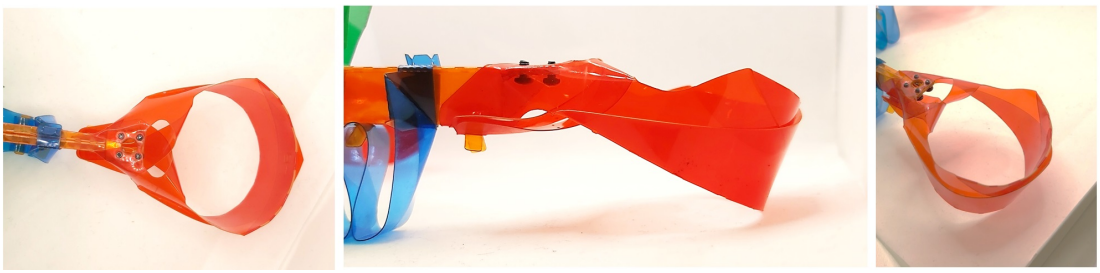

Figure 2.6: Protective foldable bumper's installation on the arms

2.2.6 Landing Gear Design

To protect the drone from the impacts during its landing, a compliant landing gear is designed and assembled under four sides of the drone. The length of the landing gear is determined such that the battery housing located under the drone does not touch the ground, even in relatively harsh landings. Figure 2.7 shows the design and assembly of this part.

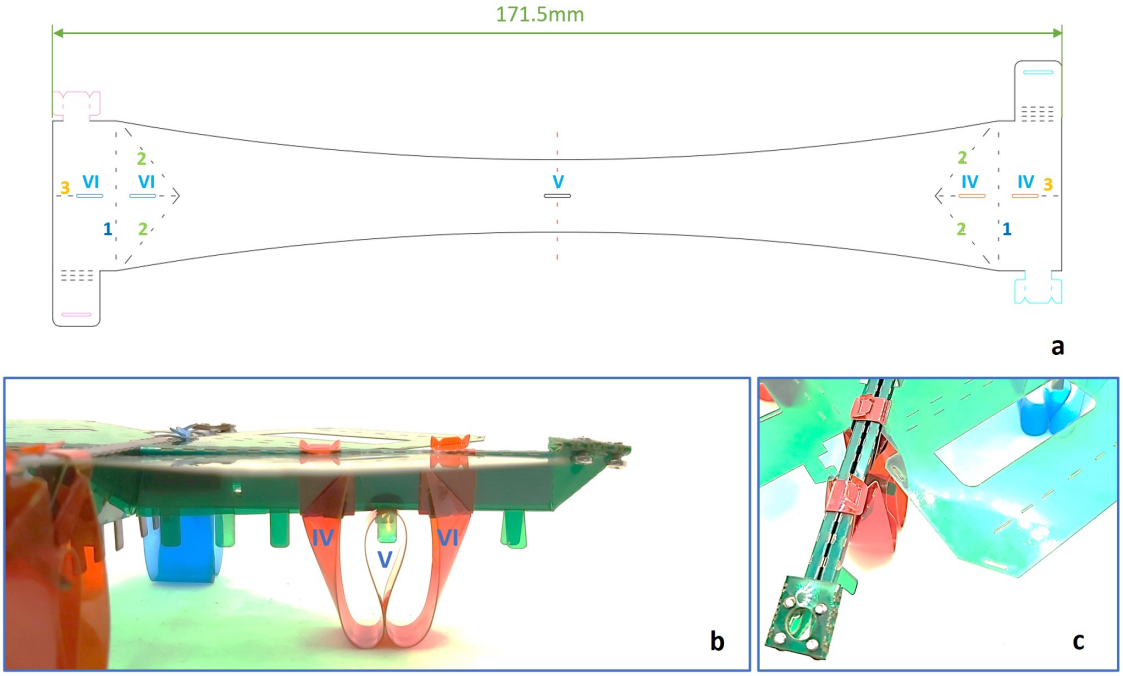

Figure 2.7: Landing gear design and assembly

As illustrated in the figure above, each landing gear uses three locks on the arm it is installed (locks **IV**, **V**, **VI**). On one side, the landing gear grips the foldable arm and locks around it after it is fastened to the lock extension number **VI**. On the other end, the landing gear fastens into the extension lock number **IV** and then grips both the arms and the central housing through the slots embedded on the housing. This helps improve the foldable body's rigidity in the required zones by limiting excessive bending and twisting along the foldable arm.

2.2.7 Battery Housing Design

The design for this part is somewhat simple. The battery housing essentially consists of a rectangular box in which sides are locked through the edges of the box. The battery can be easily placed in the box from one side. The opening side is fastened by two locks. Some edges of this box have extensions folded towards out that help increase the edge strength and also protect the battery housing from sudden fracture during harsh landings (shaded areas in Figure 2.8).

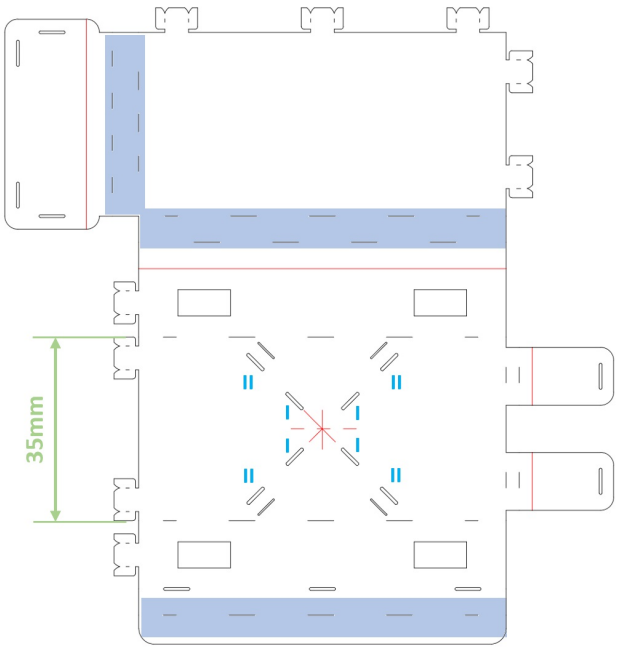

Figure 2.8: Battery housing design

The battery housing uses locks number **I**, **II** on all four sides of the arms to be installed under the foldable drone. Figure 2.9 illustrates the installation of the battery housing under the quadcopter.

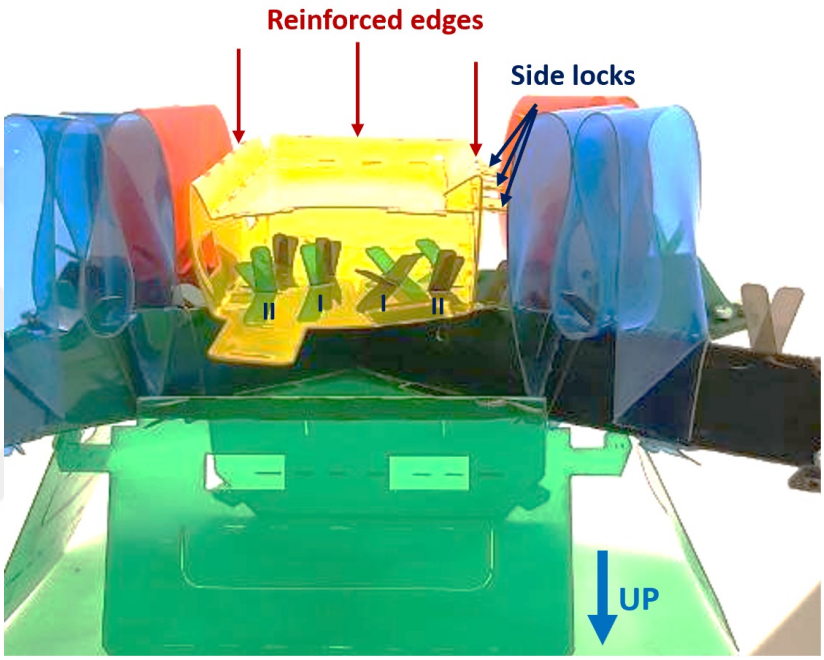

Figure 2.9: Battery housing assembly under the foldable drone

2.2.8 First-Person-View (FPV) Camera Case Design

As mentioned before, the new version of the collision-resilient foldable quadcopter is required to do outdoor inspection and surveillance missions. In order to do this, an FPV camera is installed on the front side of the quadcopter using a case made of folded PET sheet. The following figure demonstrates the design and installation of the FPV camera case on the front side of the central housing, where the camera is connected to the video transmission circuit through the cables passing through the slots cut on the front side of the central housing.

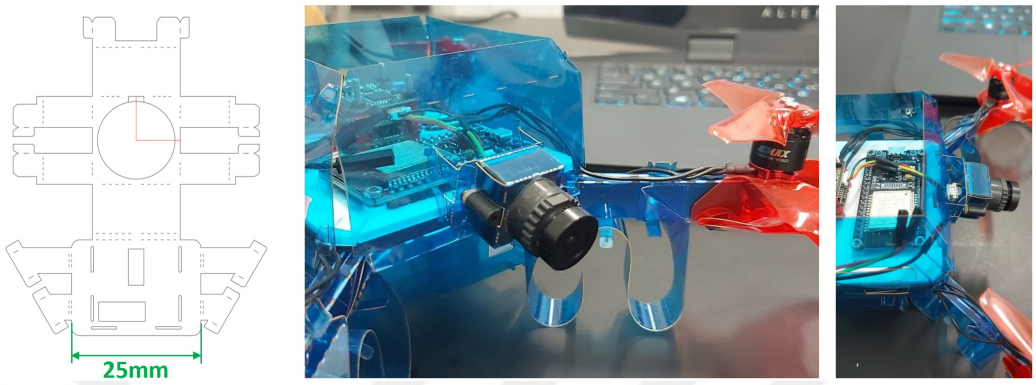

Figure 2.10: FPV camera case design and its assembly

2.2.9 Intermediate Electronics Holder Board Design

An intermediate board is designed to connect the electronics into the arms inside the central housing. This board is made of TPU using 3D printing. On the top side, the board is connected to the main electronic board of the drone, and on the lower side, it has fork-like extensions that go through the rectangular slots on the central housing (shown in orange in Figure 2.4). These extensions are then bolted through the purple holes and slots illustrated in Figures 2.2 and 2.4. The softness of the TPU material helps with the vibration canceling and maintaining the required compliance of the foldable drone. Figure 2.11 demonstrates this part and how the electronics are assembled on it.

2.2.10 MiniCore Vs. New Version

Figure 2.12 demonstrates *MiniCoRe* (right) and the new version of the Collision-Resilient Foldable Quadcopter in the same frame.

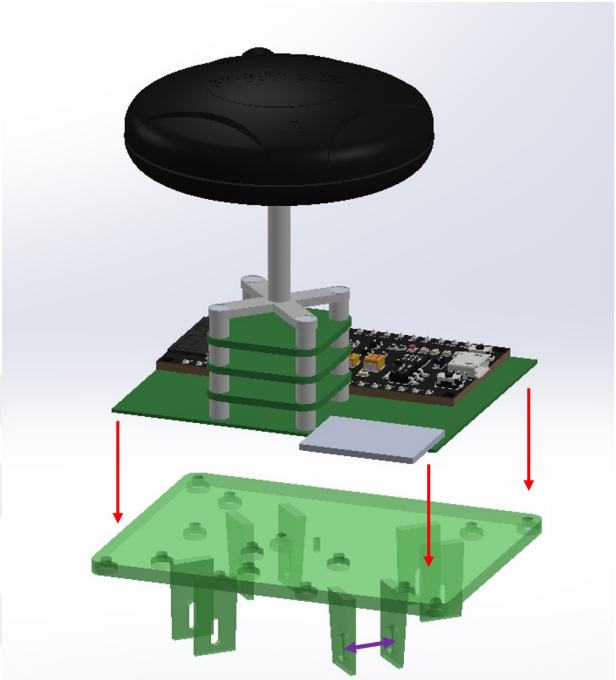

Figure 2.11: Assembly of the drone electronics on the intermediate holder board

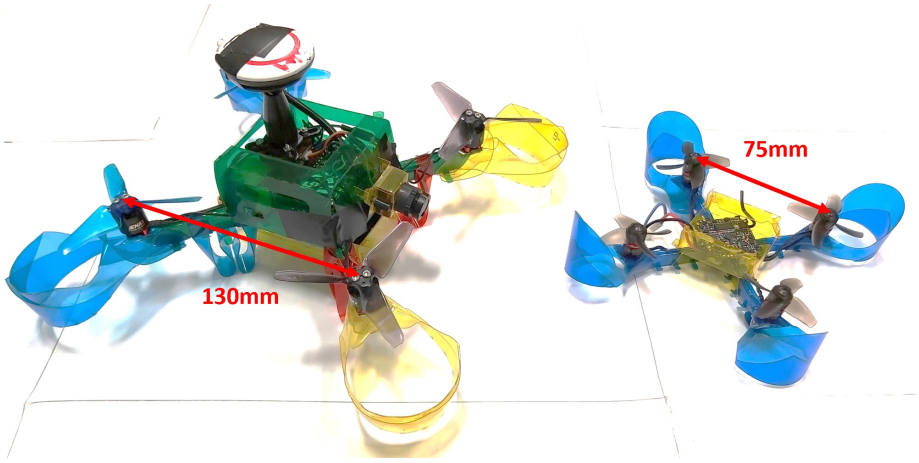

Figure 2.12: *MiniCoRe* (right) and the new version of the collision-resilient foldable micro quadcopter (left)

2.3 Component Selection

Regarding the design criteria in this project (maximum weight of 300gr and maximum dimension of $30cm * 30cm$), the following components are selected to ensure that the foldable quadcopter will fulfill its outdoor missions without failure.

2.3.1 Main Processor

As the main processing unit for the drone, *ESPRESSIF*'s ESP32-DevKitC-32E [22] is selected for the drone. Apart from its great computational capacity compared to other commercial microprocessors kits like Arduino series, ESP32 provides up to 26 **G**eneral **P**urpose **O**utput **I**nput (GPIO) pins. Furthermore, the above-mentioned kit has integrated WiFi and BlueTooth modules inside it, making no need to attach any peripheral modules/kits to secure communication. Figure 2.13 shows a ESP32-DevKit-V1 board.

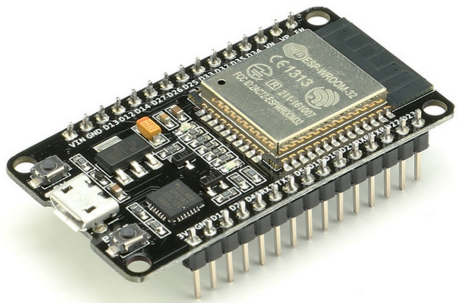

Figure 2.13: ESP32-DevKit-V1 micro-processing unit

2.3.2 Flight Controller

A commercial flight controller (GepRC STABLE F411) is selected as the low-level controller of the collision-resilient foldable quadcopter. This small-packed flight controller enables a perfect attitude (the Euler angles) control for the quadcopter. It includes a three-level circuitry; a 200mW VTX video transmission card, the

F4 flight controller, and a 4*1 **E**lectronic **S**peed **C**ontroller (ESC) unit to drive the BLDC motors [23].

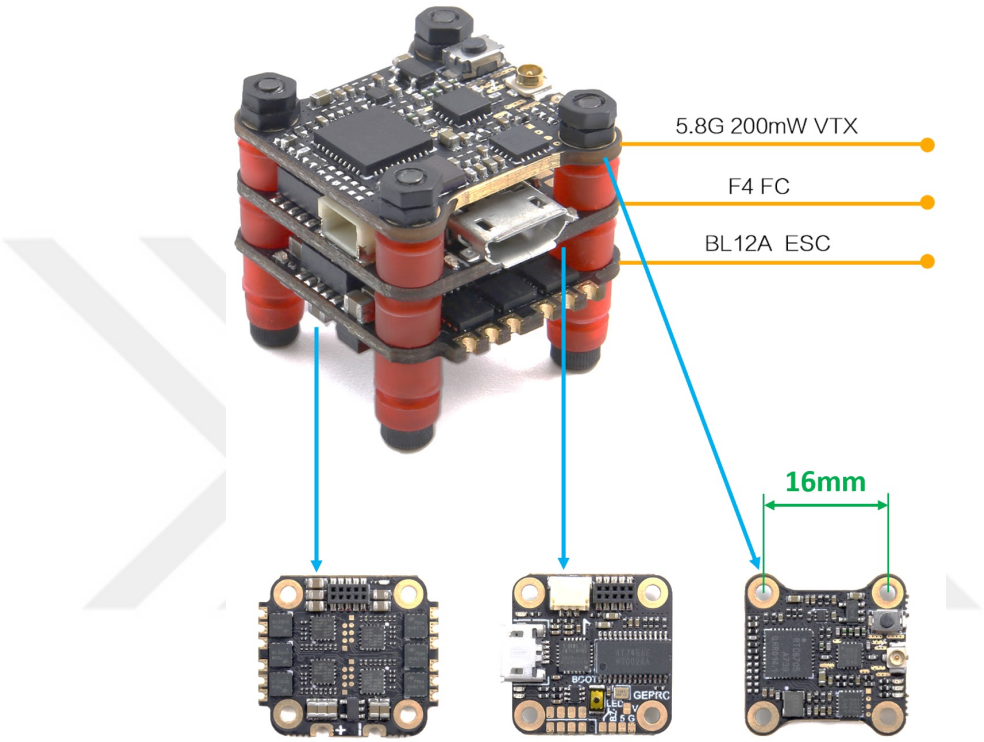

Figure 2.14: GepRC STABLE F411 flight controller

2.3.3 Inertial Measurement Unit (IMU)

IMUs are motion sensors, including MEMS (**M**icro **E**lectro-**M**echanical **S**ystems) Gyroscopes and Accelerometers (and sometimes MEMS Magnetometers). Gyroscopes measure the apparent angular velocity (angular velocity in the body frame of the vehicle) about body frame x-y-z axes, while MEMS accelerometers measure the absolute acceleration along the inertial X-Y-Z axes. Fusing the output measurements of these two sensors, the Euler angle rates of the body can be calculated. Adding a 3DOF MEMS magnetometer to these two sensors would improve the state estimation and prevent any drift in the data.

Although there is not direct use of IMUs in the quadcopter in this project, GepRC

STABLE F411 utilizes an MPU6000 (3DOF Gyroscope + 3DOF Accelerometer) for attitude estimation and control. It is good to mention here that during this research, a flight controller is developed in the MRL with MPU9250 IMU embedded in its PCB (**P**rinted **C**ircuit **B**oard). In this flight controller, MPU9250 is connected to ESP32 using I^2C serial communication protocol. Using GepRC is preferred to the mentioned flight controller because of more reliable and smoother attitude control in it.

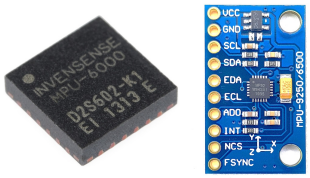

Figure 2.15: MPU6000 chipset (left), and MPU9250 IMU kit (right)

2.3.4 Altitude Sensor (Barometer)

To measure the altitude (elevation) of the quadcopter, a barometric pressure MEMS sensor MS5611 is embedded into the main PCB. The sensor is connected to the ESP32 on the drone through an I^2C serial connection and provides the absolute barometric pressure of the environment. Then this pressure can be converted to the sea-level elevation, which represents the height at which the drone is flying (with an accuracy of $\pm 15cm$).

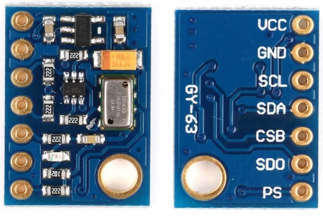

Figure 2.16: MS5611 barometric pressure sensor

2.3.5 Global Positioning System (GPS) and Compass (Magnetometer)

This sensor gives the location of the quadcopter on the Earth. For limited local use, its measurement can be used to determine the $X - Y$ position of the drone in the inertial frame. A NEO-M8 GPS sensor is selected for this purpose. This sensor connects to the ESP32 through the UART port (an asynchronous serial communication protocol) and updates the location with a frequency of 10Hz.

The NEO-M8 kit is also equipped with a compass (MEMS magnetometer sensor - IST8310), which estimates the heading angle of the drone with respect to the Earth's Magnetic North Pole. The accuracy of this sensor depends on the number of satellites it can connect to [24].

Figure 2.17: NEO-M8 GPS+Compass module

2.3.6 BLDC Motors and Propellers

Meeting the requirements defined in the project, the quadcopter is allowed to have a maximum weight of 300g (that is equal to 220g in the final version). To ensure that a quadcopter with this weight can do the expected maneuvers, it must be equipped with motor-propeller sets that secure a thrust force twice its weight. *EMAX* RS1106 6000KV BLDC motor and AVAN MINI 3" propeller are selected for this purpose. This motor-propeller set provides a nominal maximum of 150g (600g of overall thrust for four propellers). Figure 2.18 illustrates the motor-propeller set.

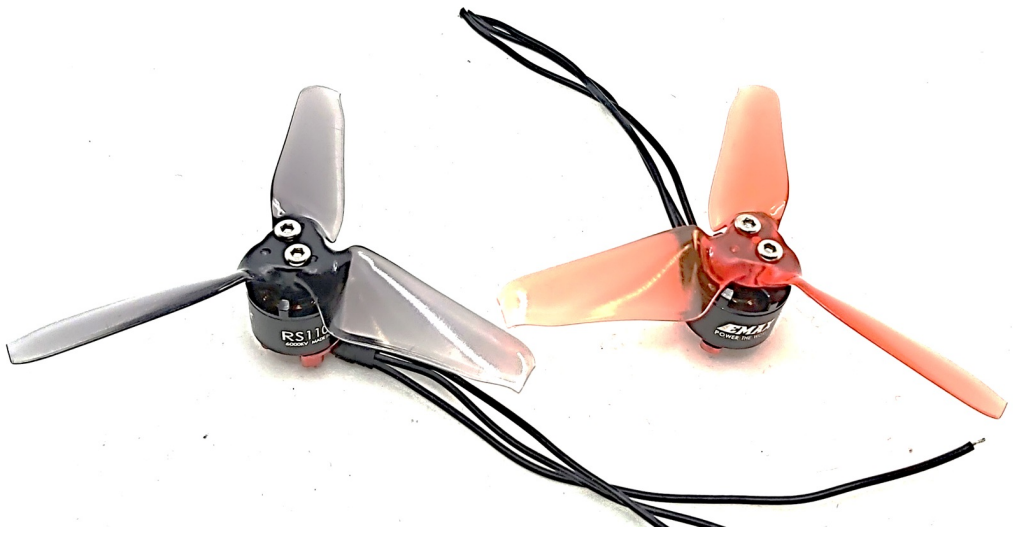

Figure 2.18: *EMAX* RS1106 6000KV and AVAN MINI 3" motor-propeller set

2.3.7 Power Storage (Battery)

Two 550mAh 2S (7.6V) high-discharge-rate (60C) DC **Li**thium-**Po**lymer (Li-Po) batteries supply the drone's power. The maximum flight duration depends on the quadcopter's mission and batteries' age, but regardless of the battery condition, they guarantee a 5-min flight for surveillance purposes.

Figure 2.19: Two parallel 2S Li-Po batteries used to power the collision-resilient foldable drone

2.3.8 FPV Camera and Video Transmission

A compatible analog FPV camera is chosen to be installed on the quadcopter. This camera is connected to the video transmission (VTX) board on GepRC, which sends the video to a 5.8GHz video receiver. The video receiver outputs the received video through an HDMI output which can be readily converted to a USB output to be received and processed in the ground station computer.

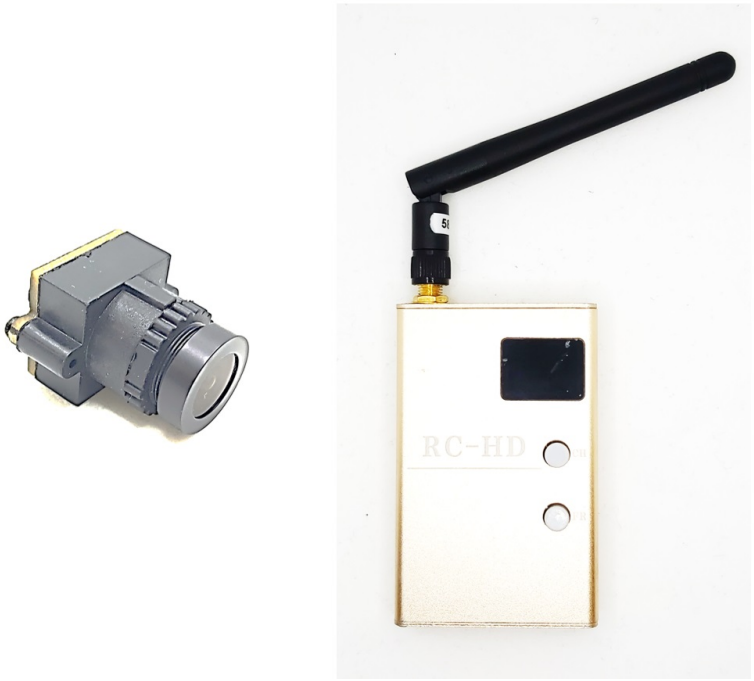

Figure 2.20: FPV camera (left), 5.8GHz video receiver (right)

2.4 System Design

Regarding the missions defined for the collision-resilient foldable quadcopter in the current project, two to five of these quadcopters should fly at the same time for an inspection and surveillance task and provide images from a targeted infrastructure from different angles. To achieve this goal, the drones are communicating with a ground station from which they receive start (take-off), terminate (landing), and abort (emergency landing) commands. The ground station is an ESP32 kit, connected to a portable computer (laptop). The ESP32, receives the commands from the user and sends them to the ESP32 boards on the other end (on drones) through ESP-NOW (a peer-to-peer WiFi protocol specific to ESP32). The ESP on the drones is responsible for acquiring data from the sensors (barometer, compass, and GPS), state estimation, and communication. The full map of the system components and the communication between them is illustrated in Figure 2.21. It is good to mention that this diagram demonstrates the connection between the ground station ESP32 and one of the drones. The connection is scalable for multiple drones.

The total mass of the quadcopter is 220g, and the foldable body parts only constitute 36.5g of it. The mass distribution of the different components of the quadcopter is illustrated in Figure 2.22.

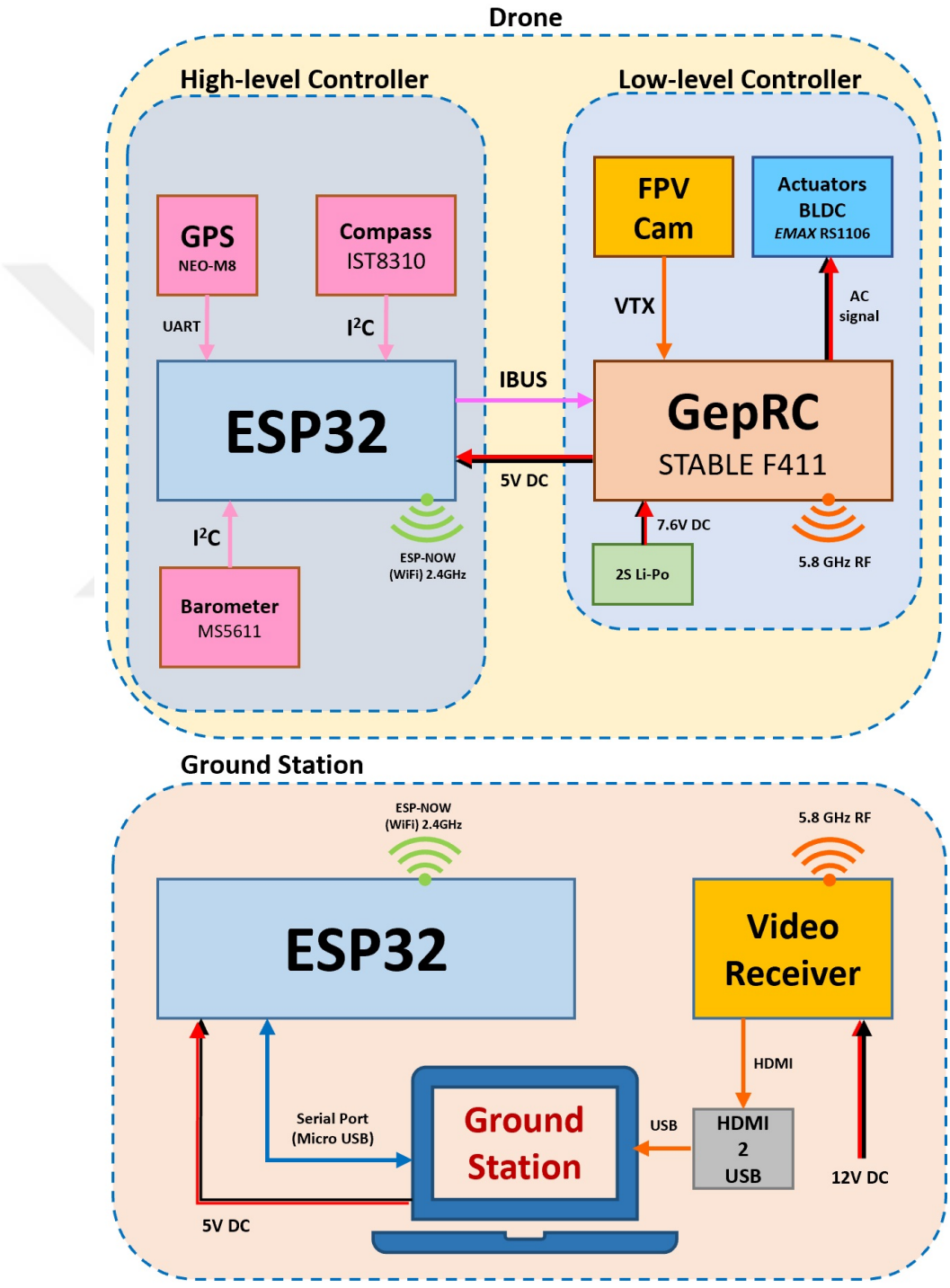

Figure 2.21: System demonstration for the collision-resilient foldable quadcopter

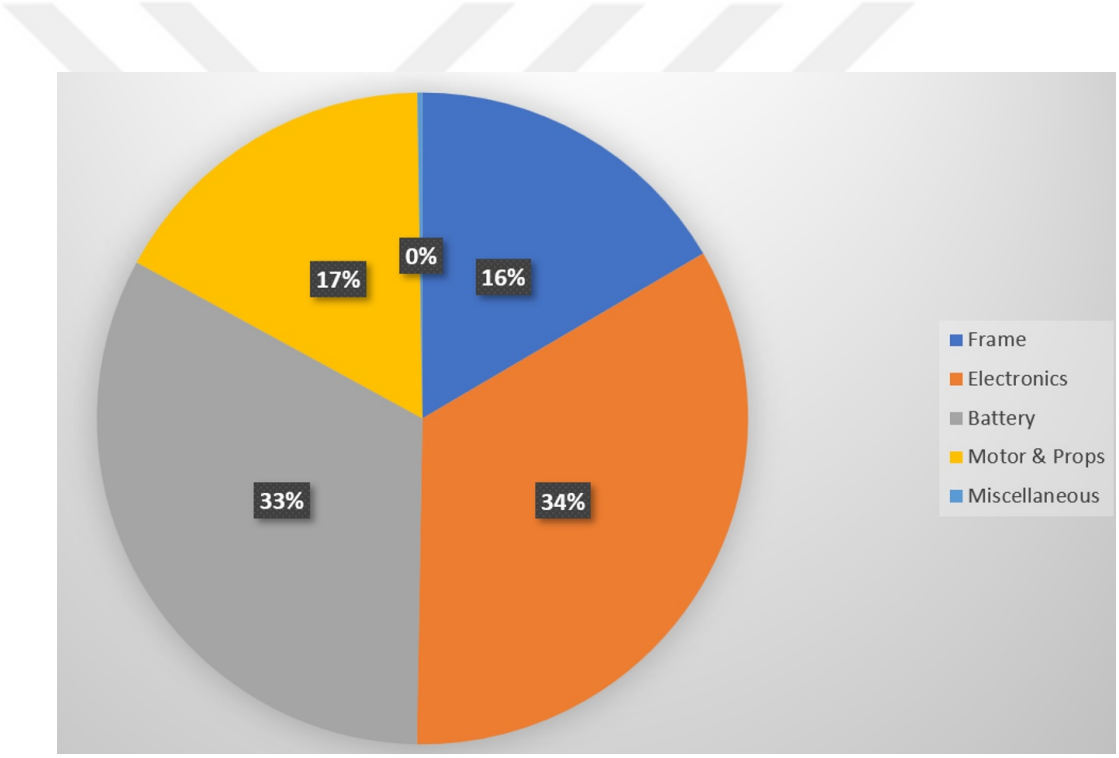

Figure 2.22: Mass distribution of different components of the collision-resilient foldable quadcopter

Chapter 3

Quadcopter and Soft Impact Dynamics

The governing mathematical equations representing the dynamics of the collision-resilient foldable quadcopter and the soft impact spatial dynamics are derived in the course of this chapter.

3.1 Multi-Rotor VTOL Dynamics in Literature

Anticipating the rapid growth of applications for multi-rotor drones in the mid-2000s, several researches and investigations started to identify the dynamics of multi-rotor VTOLs. The more accurate the dynamic model represents the system behavior, the easier it becomes to design a controller capable of controlling the system as desired.

While the early research was focused on investigating the effects of existing forces on the multi-rotor VTOLs [4, 25], recent studies mostly explore new potential dynamics like tilted-angle actuators [26, 27] and over-actuated redundant configurations [28, 29]. Also, with the growth in consumer drone production and

use, to ease the flight control for unskilled drone pilots, the effects of minor disturbing forces on drone dynamics are investigated, and many solutions have proposed (both in academia and industry) to solve these issues. The most significant addressed issues are *Ground Effect* and *Propwash*. Ground effect is the forces resulting from turbulent air flow affecting the multi-rotor VTOL during take-off and landing [30, 31, 32, 33]. Propwash is the lateral force caused by sudden changes in the propeller's speed in agile maneuvers causing vibration in the rotors [34].

3.2 Quadcopter Dynamics

The collision-resilient foldable quadcopter is designed in a way to show compliance during lateral impacts. However, the designed body is stiff enough that it does not bend or vibrate during flight. Therefore, the general dynamic model of the quadcopter is derived based on the existing models for rigid quadcopters in literature [4]. The assumptions made to derive this model are listed below:

- The quadcopter's structure is fully rigid in the absence of collision.
- The quadcopter's **C**enter **o**f **G**ravity (CoG) is placed in its geometrical center, where all four rotors have the same distance to it.
- The body-centered inertial coordinate system coincides with the CoG.
- The quadcopter is fully symmetric about the axes of the CoG-centered inertial frame.
- The existence of minor forces and moments (hub forces and rolling moments) are neglected due to the efficient design of the quadcopter and *Stable-Mode* controller design that always keeps it close to the hovering point.

3.2.1 Quadcopter Configuration

As mentioned in the previous chapter, quad-rotor multi-copters are introduced in various configurations. However, the most popular configurations are $+$-shaped and $\boldsymbol{X}$-shaped. The collision-resilient foldable quadcopter is developed as an X-shaped quadcopter. Figure 3.1 illustrates these two popular quadcopter configurations.

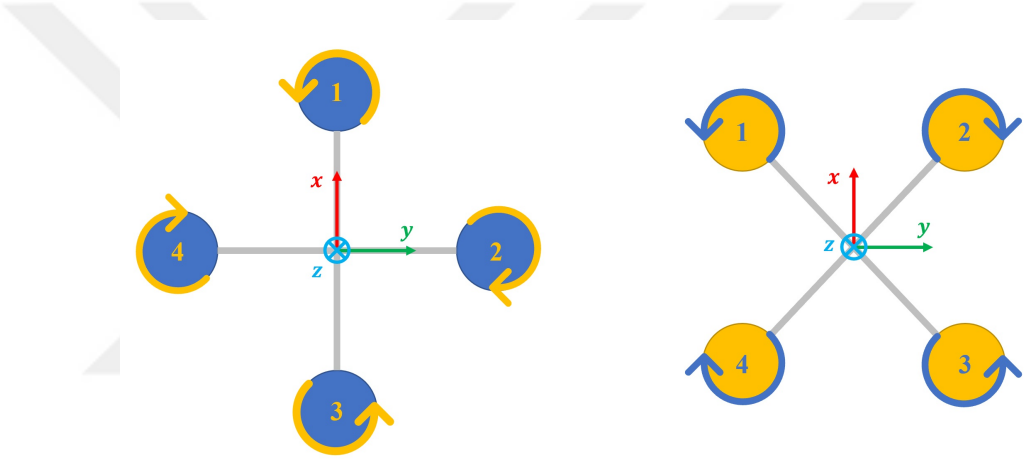

Figure 3.1: Quadcopter configurations, coordinate definition, and rotor sequence; $+$-shaped (left) and X-shaped (right)

3.2.2 Coordinate System Definition and Kinematics

The state of the quadcopter (position and orientation) is defined using the coordinate systems demonstrated in Figure 3.2. The coordinate system represented by $[\ {}^{e}X \quad {}^{e}Y \quad {}^{e}Z]$ shows the Earth-fixed frame (Inertial) with Z-axis toward the Earth center (direction of gravity), while $[\ {}^{b}X \quad {}^{b}Y \quad {}^{b}Z]$ is the Body-centered frame (Inertial)– always parallel to the Earth-fixed frame coordinates. The other coordinate system $[\ {}^{b}x \quad {}^{b}y \quad {}^{b}z]$ is the Body-fixed frame (rotating). The airframe orientation in the inertial frame can be represented by a rotation from the Body-centered inertial frame to the Body-fixed rotating frame.

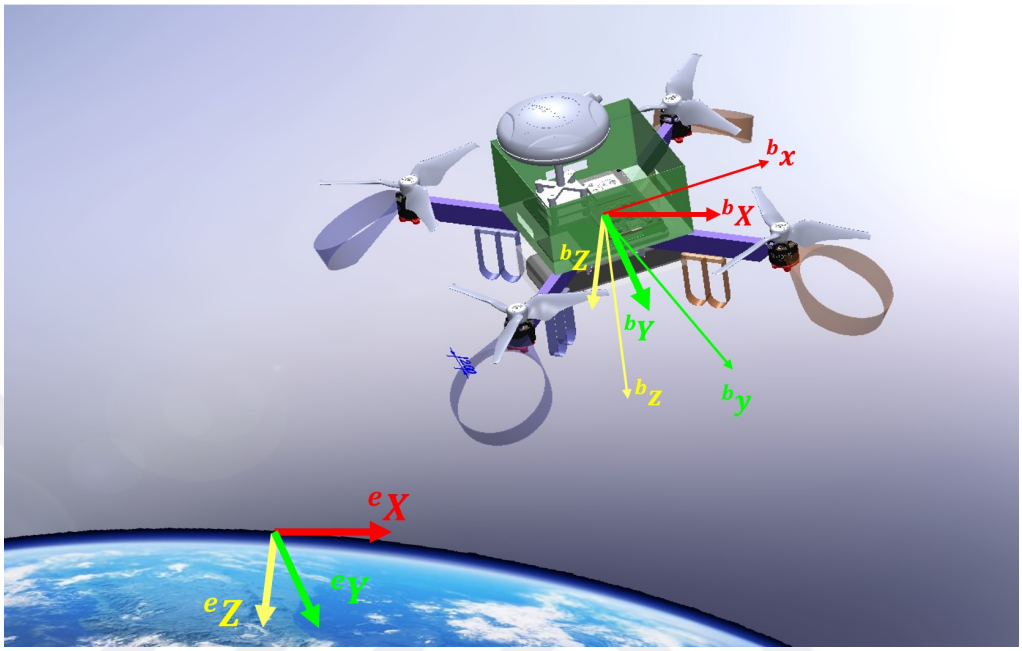

Figure 3.2: Illustration of the inertial and rotating reference frames on the quadcopter

$$^{b/e}\boldsymbol{r} = R\,{}^{b/b}\boldsymbol{r} \tag{3.1}$$

where R is the resultant rotation matrix that can be decomposed into three consecutive elemental rotations about $Z(\psi)$, $Y(\theta)$, and $X(\phi)$ axes.

$$R = Z(\psi)\,Y(\theta)\,X(\phi) \tag{3.2}$$

where ϕ, θ, and ψ are the Euler angles *Roll*, *Pitch*, and *Yaw*, respectively. Therefore, Equation 3.1 can be rewritten as

$$\begin{aligned}\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} &= \begin{bmatrix} c\psi & -s\psi & 0 \\ s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c\theta & 0 & s\theta \\ 0 & 1 & 0 \\ -s\theta & 0 & c\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\phi & -s\phi \\ 0 & s\phi & c\phi \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \\ &= \begin{bmatrix} c\theta c\psi & -c\phi s\psi + s\phi s\theta c\psi & s\phi s\psi + c\phi s\theta c\psi \\ c\theta s\psi & c\phi c\psi + s\phi s\theta s\psi & -s\phi c\psi + c\phi s\theta s\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}\end{aligned} \tag{3.3}$$

3.2.3 Equations of Motion (EoM)

It is assumed that the minor forces influencing the quadcopter can be neglected in this study because of the efficient design and stable-mode flight of the quadcopter. Therefore, considering the forces and moments resulting from propellers' thrust and drag force, the drone body's gyroscopic effect, and the propellers' gyroscopic effect as the only affecting forces on the drone, the quadcopter's dynamic model can be derived as follows.

Defining the state vector as

$$\bar{X} = [\phi\ \dot{\phi}\ \theta\ \dot{\theta}\ \psi\ \dot{\psi}\ X\ \dot{X}\ Y\ \dot{Y}\ Z\ \dot{Z}]^T \tag{3.4}$$

where $[X\ Y\ Z]^T$ is the CoG position vector of the quadcopter in the Earth-fixed inertial frame and $[\phi\ \theta\ \psi]^T$ is the drone body Euler angles represented in the CoG-centered body frame. The dynamic model is defined as

$$\dot{\hat{X}} = \begin{cases} \dot{\varphi} \\ \dot{\theta}\dot{\psi}a_1 + \dot{\theta}a_2\Omega_r + b_1U_2 \\ \dot{\theta} \\ \dot{\varphi}\dot{\psi}a_3 + \dot{\varphi}a_4\Omega_r + b_2U_3 \\ \dot{\psi} \\ \dot{\varphi}\dot{\theta}a_5 + b_3U_4 \\ \dot{x} \\ u_x\frac{1}{m}U_1 \\ \dot{y} \\ u_y\frac{1}{m}U_1 \\ \dot{z} \\ g - (\cos\varphi \cdot \cos\theta)\frac{1}{m}U_1 \end{cases} \tag{3.5}$$

where coefficients a_1 to a_5 and b_1 to b_3 are defined as

$$\left.\begin{array}{l} a_1 = (I_{yy} - I_{zz})/I_{xx} \\ a_2 = \frac{J_r}{I_{xx}} \\ a_3 = (I_{zz} - I_{xx})/I_{yy} \\ a_4 = \frac{J_r}{I_{yy}} \\ a_5 = (I_{xx} - I_{yy})/I_{zz} \end{array}\right| \begin{array}{l} \\ b_1 = l/I_{xx} \\ b_2 = l/I_{yy} \\ b_3 = 1/I_{zz} \\ \\ \end{array} \tag{3.6}$$

and

$$\begin{aligned} u_x &= (\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi) \\ u_y &= (\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi) \end{aligned} \tag{3.7}$$

in which J_r is the rotor inertia and I_{xx}, I_{yy}, and I_{zz} are moments of inertia about x, y, and z axes, respectively. l is the roll/pitch angle arm, and Ω_r is the residual angular speed of the quadcopter's four rotors defined as

$$\Omega_r = |\Omega_1 - \Omega_2 + \Omega_3 - \Omega_4| \tag{3.8}$$

where Ω_i is the angular speed of the i-th rotor (in $[rpm]$) in an X-shaped quadcopter (Figure 3.1). Parameters U_i are the input forces and moments to the system defined as

$$\begin{cases} U_1 = K_T\left(PWM_1 + PWM_2 + PWM_3 + PWM_4\right) \\ U_2 = K_T\left(PWM_1 - PWM_2 - PWM_3 + PWM_4\right) \\ U_3 = K_T\left(PWM_1 + PWM_2 - PWM_3 - PWM_4\right) \\ U_4 = K_D\left(PWM_1 - PWM_2 + PWM_3 - PWM_4\right) \end{cases} \tag{3.9}$$

where K_T and K_D are linear thrust and drag coefficients for variable-pitch propellers selected for the quadcopter (The complete process of obtaining the value for these variables is elaborated in Appendix A). The PWM_i is the PWM signal ratio sent to the i-th rotor in the interval [0 1]

The mass and inertia properties of the quadcopter, alongside the rotor inertia, are extracted from the *SolidWorks* model developed for the collision-resilient foldable micro quadcopter. Table 3.1 displays the values for the parameters and constants in the dynamic model.

Table 3.1: Parameter and Constant values for collision-resilient foldable micro quadcopter's dynamic model

Param./Const.	**Definition**	**Value [unit]**
I_{xx}	moment of inertia about the body-fixed x-axis	4.190758×10^{-4} $[kg.m^2]$
I_{yy}	moment of inertia about the body-fixed y-axis	4.523956×10^{-4} $[kg.m^2]$
I_{zz}	moment of inertia about the body-fixed z-axis	8.304871×10^{-4} $[kg.m^2]$
J_r	rotor inertia	1.4844×10^{-7} $[kg.m^2]$
l	rolling/pitching arm	8×10^{-2} $[m]$
k_T	linear thrust coefficient	1.589 $[\frac{N}{pwm\ ratio}]$
k_D	linear drag coefficient	0.09 $[\frac{N}{pwm\ ratio}]$

3.3 Soft Impact Dynamics

The collision-resilient foldable quadcopter is designed such that it is rigid enough during its normal hovering and flight. But while an impact happens, the quadcopter's structure shows compliance, corresponding to the intensity of the impact, to alleviate the effect of the collision. The reason for using this dual-stiffness is that a completely soft structure is prone to serious oscillations, and also deriving a dynamic model for a soft body is not an easy job, and it is not attainable without hiring computational methods like machine learning. On the other hand, using a fully rigid body gives no compliance and damping in the impacts and makes the drone subject to serious damage and crashes.

Although the quadcopter is assumed as a fully rigid body during flight, it is necessary to investigate its compliant nature during collisions and augment the dynamic model developed in the previous section.

3.3.1 Definition of Collision as Impact Force

Primary studies investigating the dynamic contact between rigid bodies suggest that the behavior of the bodies in contact is similar to the case when there is a mechanical impedance between them. As a result, the physical collision between two solids is modeled as a linear spring and a linear damper in the contact point (or surface) in these studies [35]. While the linear model (Kelvin-Voigt model) justifies many simple cases of viscoelastic solid impact with acceptable accuracy, it does not adequately explain the impact behavior for softer materials and high-velocity contacts/collisions [36]. To solve this problem, a nonlinear loading/unloading behavior is suggested by Hunt and Crossly [37]. They present their solution as a viscoelastic nonlinear spring-damper element generating a normal force to the contact surface during the high-velocity impacts (vibro-impacts) represented by

$$N = K\delta^n + D\dot{\delta} \tag{3.10}$$

where, N is the normal impact force, δ is the deformation (or penetration) depth, and n is the nonlinear spring exponent. K and D are spring and damping ratios, respectively.

More established impact theories also consider the effect of existing friction forces on the contact surface [38, 39, 40].

3.3.2 Impact Detection

To determine whether the quadcopter is in contact with a surface or not, in the simulation environment, a simple geometry is introduced to represent its morphology. The bumpers installed on the collision-resilient foldable quadcopter are the first components touching the colliding surface. In the model, the bumpers are considered to be spheres with a radius of $\rho = 4cm$. Therefore, if any rotor center approaching an obstacle has a distance $\delta \leq 4cm$ to the closest point of that obstacle, its protective bumper has a contact/impact with that surface/obstacle. Figure 3.3 shows an illustration of the simplified geometry representing the collision-resilient quadcopter in MATLAB approaching a wall in $X = 10m$.

Since the detection of the contact/impact between the bumpers and an arbitrary obstacle depends on the geometry of the colliding obstacle (it can be a plane, curved surface, column, prism, or any irregular shape or surface), the simple case of impact to a vertical wall is addressed in this section.

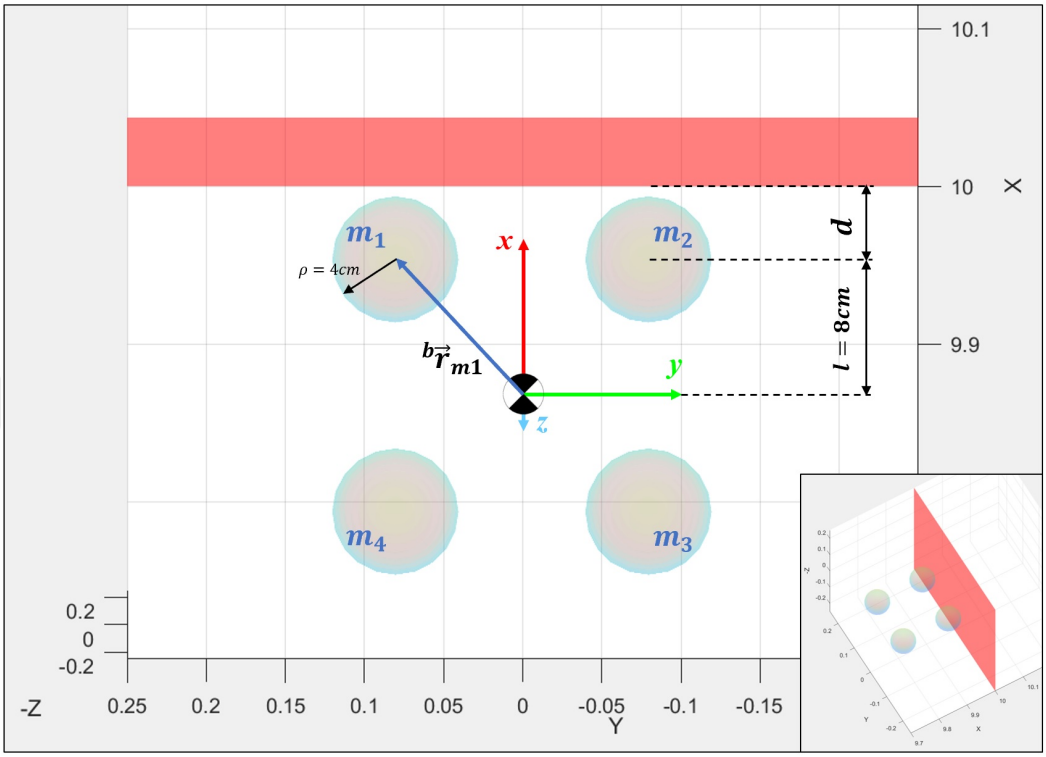

Figure 3.3: Simplified geometry of the collision-resilient quadcopter in MATLAB

For each rotor in the drone, the rotor center's position in the body-fixed rotating frame is always fixed and can be represented as

$$\begin{cases} {}^{b/b}\vec{r}_{m1} = [+0.08,\ -0.08,\ 0]^T \\ {}^{b/b}\vec{r}_{m2} = [+0.08,\ +0.08,\ 0]^T \\ {}^{b/b}\vec{r}_{m3} = [-0.08,\ +0.08,\ 0]^T \\ {}^{b/b}\vec{r}_{m4} = [-0.08,\ -0.08,\ 0]^T \end{cases} \tag{3.11}$$

Using the rotation matrix derived in Equation 3.3, the position vector for the i-th rotor center in the body-fixed inertial frame can be calculated as

$${}^{b/e}\vec{r}_{mi} = R\ {}^{b/b}\vec{r}_{mi} \tag{3.12}$$

Adding the position of the CoG of the drone in the global frame ($\vec{r}_{CoG}$) to these vectors, the position of the rotor centers in the Earth-fixed inertial frame is

$$^{e/e}\vec{r}_{mi} = \vec{r}_{CoG} + {}^{b/e}\vec{r}_{mi} = \vec{r}_{CoG} + R \ {}^{b/b}\vec{r}_{mi} \tag{3.13}$$

Checking the distance between each rotor center and a general plane (wall) defined by $aX + bY + cZ = -d$ in the global frame assuming $^{e/e}\vec{r}_{mi} = [X_i \ Y_i \ Z_i]^T$

$$\delta_i = \frac{|aX_i + bY_i + cZ_i + d|}{\sqrt{a^2 + b^2 + c^2}} \tag{3.14}$$

If $\delta_i \leq \rho$, it can be concluded that the i-th bumper is colliding with the plane.

To extract the effecting forces during the impact, two scenarios can be considered. In the first one, the arms and bumpers of the collision-resilient quadcopter bend when it collides with the plane (wall), and it is the amount of deformation (Δ_d) and its rate that defines how much force is being applied to the quadcopter due to stiffness and damping. This almost represents what happens during the soft impact in reality. The other case is to assume the quadcopter's body as a rigid frame colliding with a soft obstacle where the amount of penetration (Δ_p) and its rate can be used to determine the stiffness and damping forces generated in the contact [41]. The difference between the deformation approach and the penetration approach is illustrated in Figure 3.4.

Since the first approach requires involving the solid mechanics and vibration analyses of the soft bending membrane in the dynamics, it is not recommended to be used in the simulation. Using the second approach, on the other hand, incredibly simplifies the impact dynamic model and allows us to see the collision-resilient foldable quadcopter as a compliant bulk colliding with another body.

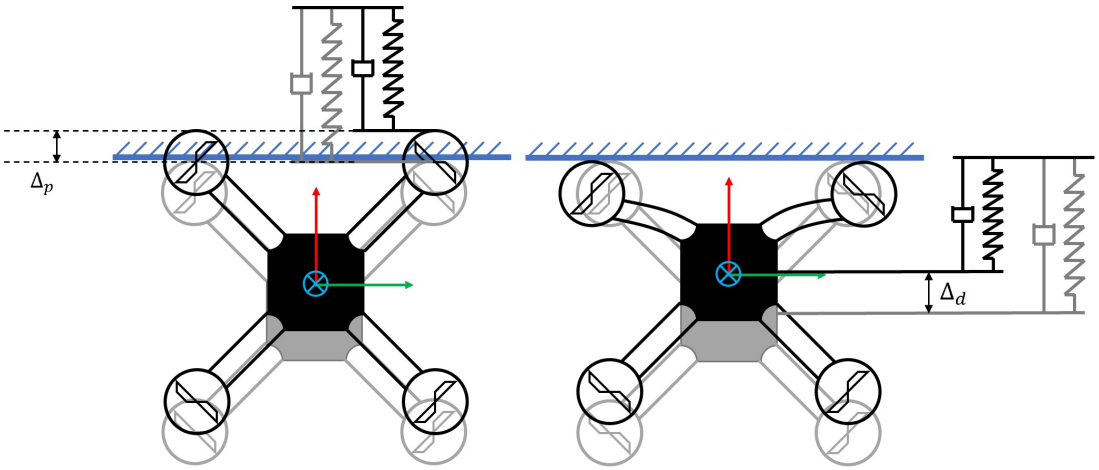

Figure 3.4: Soft impact represented as visco-elastic deformation (Δ_d) in the compliant body (right), and visco-elastic penetration (Δ_p) into the colliding surface (left)

3.3.3 Determining the Penetration Vector

Assuming that the colliding plane can be approached from both sides, to define the penetration vector for each motor bumper, first it is essential to define some parameters.

Definition: The distance between the CoG of the drone and the collision plane $\delta_{CP} = \delta_{CoG \to P}$

$$\delta_{CP} = \frac{|aX_{CoG} + bY_{CoG} + cZ_{CoG} + d|}{\sqrt{a^2 + b^2 + c^2}} \tag{3.15}$$

Definition: The distance between the CoG of the drone and the plane passing from the i-th rotor center parallel to the collision plane $\delta_{CMi} = \delta_{CoG \to Mi}$

$$\delta_{CMi} = \frac{\left|aX_{CoG} + bY_{CoG} + cZ_{CoG} + d_i{}'\right|}{\sqrt{a^2 + b^2 + c^2}} \tag{3.16}$$

where $d_i^{'} = |aX_i + bY_i + cZ_i|$

Definition: The angle between ${}^{b/e}\vec{r}_{mi}$ and the collision plane normal vector $\vec{n}$; γ_i

$$\gamma_i = \text{acos}\left(\frac{{}^{b/e}\vec{r}_{mi} \cdot \vec{n}}{|{}^{b/e}\vec{r}_{mi}||\vec{n}\,|}\right) \tag{3.17}$$

If any collisions have been detected for any of the bumpers (Equation 3.13), the penetration vector can be determined as follows (defining $\vec{p_i}$ as the penetration vector for the i-th bumper starting at point P_{ai} and ending at point P_{bi})

- if $\delta_{CP} > \delta_{CMi}$ & $\gamma_i \leq \pi/2$

$$\begin{cases} \vec{P}_{ai} = {}^{e/e}\vec{r}_{mi} + \delta_i\ \vec{n} \\ \vec{P}_{bi} = {}^{e/e}\vec{r}_{mi} + \rho\ \ \vec{n} \\ \vec{p}_i\ = \vec{P}_{bi} - \vec{P}_{ai} \end{cases} \tag{3.18}$$

- if $\delta_{CP} \leq \delta_{CMi}$ & $\gamma_i \leq \pi/2$

$$\begin{cases} \vec{P}_{ai} = {}^{e/e}\vec{r}_{mi} - \delta_i\ \vec{n} \\ \vec{P}_{bi} = {}^{e/e}\vec{r}_{mi} + \rho\ \ \vec{n} \\ \vec{p}_i\ = \vec{P}_{bi} - \vec{P}_{ai} \end{cases} \tag{3.19}$$

- if $\delta_{CP} \leq \delta_{CMi}$ & $\pi/2 < \gamma_i \leq \pi$

$$\begin{cases} \vec{P}_{ai} = {}^{e/e}\vec{r}_{mi} + \delta_i\ \vec{n} \\ \vec{P}_{bi} = {}^{e/e}\vec{r}_{mi} - \rho\ \ \vec{n} \\ \vec{p}_i\ = \vec{P}_{bi} - \vec{P}_{ai} \end{cases} \tag{3.20}$$

- if $\delta_{CP} > \delta_{CMi}$ & $\pi/2 < \gamma_i \leq \pi$

$$\begin{cases} \vec{P}_{ai} = {}^{e/e}\vec{r}_{mi} - \delta_i\ \vec{n} \\ \vec{P}_{bi} = {}^{e/e}\vec{r}_{mi} - \rho\ \ \vec{n} \\ \vec{p}_i\ = \vec{P}_{bi} - \vec{P}_{ai} \end{cases} \tag{3.21}$$

3.3.4 Deriving the Impact Forces from the Penetration Vector

From Equation 3.9, the normal visco-elastic reaction of the presumed spring-damper element in the contact surface can be calculated as

$$\vec{N}_i = -\left[K\left|\vec{p}_i\right|^n \hat{n}_{p_i} + D\left|\dot{\vec{p}}_i\right| \hat{n}_{\dot{p}_i}\right] \tag{3.22}$$

where $\hat{n}_{p_i}$ and $\hat{n}_{\dot{p}_i}$ are the normal vectors representing $\vec{p}_i$ and $\dot{\vec{p}}_i$ directions. The minus sign shows that the normal visco-elastic forces act in the opposite direction of the penetration vector and its growth.

The friction occurs at the contact surface between two colliding objects and its magnitude is equal to

$$F_{ti} = \mu_d \left|\vec{N}_i\right| \tag{3.23}$$

where μ_d is the dynamic coefficient of friction. The direction of the dynamic friction force always opposes the direction of the tangential velocity on the surface. This direction can be determined by projecting the $\dot{\vec{p}}_i$ vector on the contact plane which is basically parallel to $\dot{\vec{P}}_{ai}$

$$\mathrm{Proj}_S\left(\dot{\vec{p}}_i\right) \parallel \dot{\vec{P}}_{ai} \tag{3.24}$$

where, S represents the contact plane defined by equation $aX + bY + cZ = -d$. Therefore,

$$\vec{F}_{ti} = -\mu_d \left|\vec{N}_i\right| \hat{n}_{\dot{\vec{P}}_{ai}} \tag{3.25}$$

It should be mentioned that the existence of static friction is neglected due to the continuous movement of the quadcopter during collisions and ignoring the side movements of the bumpers caused by arm deflections. This assumption does not affect the nature of the impact because, in the case of drones, the friction force is not considerable, and it is not a driving force for the robot– unlike the legged robots [41].

The resultant force at each bumper is the sum of the normal and tangential forces exerted on that bumper

$$\vec{F}_i = \vec{N}_i + \vec{F}_{ti} \tag{3.26}$$

The resulting moment from this force about the CoG is equal to

$$\vec{M}_i = \vec{r}_{CoG \rightarrow P_{ai}} \times \vec{F}_i \tag{3.27}$$

The resultant force and moment applied by the impact on the drone's CoG can be easily determined as

$$\begin{cases} \vec{F}_{\text{impact}} = \sum \vec{F}_i \\ \vec{M}_{\text{impact}} = \sum \vec{M}_i \end{cases} \tag{3.28}$$

Figure 3.5 illustrates the acting forces on the contact surface as the bumpers impact into it. The penetration vector, the normal spring force, the normal damping force, and the tangential friction force are depicted in black, blue, red, and cyan, respectively. (proportions are not right)

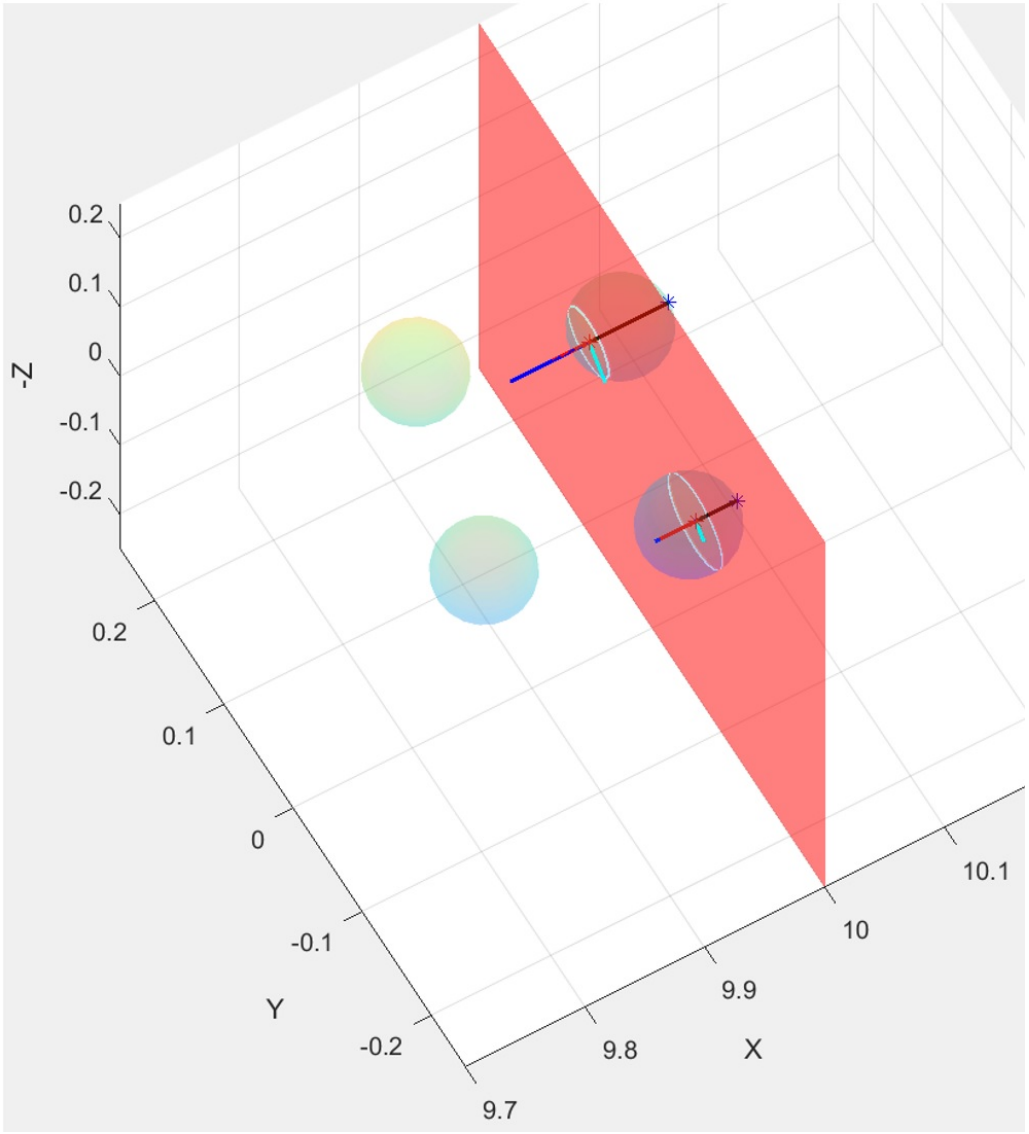

Figure 3.5: Demonstration of the impact forces

Chapter 4

Control, Simulation, and Experiments

In this chapter, a control scheme identical to the real controller used in to control the collision-resilient foldable micro quadcopter is developed to navigate the drone in the simulation environment in *Simulink*. The same scheme is used to control the quadcopter's model in *Gazebo*– the physical simulation environment in **R**obotic **O**peration **S**ystem (ROS). Ultimately, the soft impact dynamics are added to the simulations, and various impact scenarios are simulated, and the results are discussed.

The performance of the designed collision-resilient foldable body in the field experiments is also discussed.

4.1 Compliant Multi-rotor MAV Control in Literature

Since the emergence of multi-rotor UAVs in the early 2000s, their control problem has been appealing to researchers in academia and industry. Due to the complex

nonlinear nature of multi-rotor VTOL dynamics, the use of nonlinear control methods was investigated for attitude and position control of these UAVs [42, 43]. But at that time, because of the hardware limitations like limited computational capacity, especially for MAVs, implementation of these algorithms on small-scale drones was not feasible. Hence, in practice, simpler controllers were used [44, 45].

Due to their simplicity and therefore popularity in the industry, **P**roportional-**I**ntegral-**D**erivative (PID) controllers were utilized as a primary choice for attitude and altitude control for consumer drones [46, 47]. This was possible by linearization of the drone dynamics around its hovering point and decomposition of the Multi-Input-Multi-Output (MIMO) representation of the multi-rotor dynamics into multiple Single-Input-Single-Output (SISO) representations. Use of cascaded PID controllers in different configurations (P-PID, PI-PI, PI-PID) is preferable to achieve a fine control of drone's attitude/altitude [48, 49].

With recent advancements in microprocessors, flight controllers, and communication technologies [50, 51], the implementation of computationally expensive control algorithms on MAVs is no longer an issue. As a result, complex estimation algorithms [52, 53, 54, 55] and advanced controller schemes [56, 57, 58, 59] are being used on MAVs. Currently, investigations in a variety of areas of micro VTOL UAVs are going on, such as trajectory planning and optimization [60, 61, 62, 63], swarm and formation [64, 65, 66, 67], and Simultaneous Localization And Mapping (SLAM) [68, 69, 70].

It is good to mention here that, in collaboration with the current project, *Eraslan* and *Yıldız* at *Bilkent Systems Laboratory* investigated the modeling, flight control, and human-in-the-loop stability of a flexible quadcopter [71, 72].

4.2 Flight Control Strategy and Controller Design in Simulink

A cascaded PID control scheme is selected for low-level (attitude) and high-level (altitude and position) control of the collision-resilient foldable quadcopter. The reason for this is the fast implementation, easy tuning, and relatively low computational load of the PID controllers. The drawback of this method is the low reliability of the controller during aggressive, agile maneuvers far from the drone's hovering point. However, this would not be the case for the real drone in outdoor missions because of smooth path definition and tracking in stable-mode (not acro-mode) during its inspection and surveillance missions.

A similar scheme is used here to control the model of the collision-resilient foldable quadcopter in the simulation environment in *Simulink*. The full control loop in the simulation is illustrated in Figure 4.1.

The modeling of each block in the control loop designed in *Simulink* is elaborated in the following subsections.

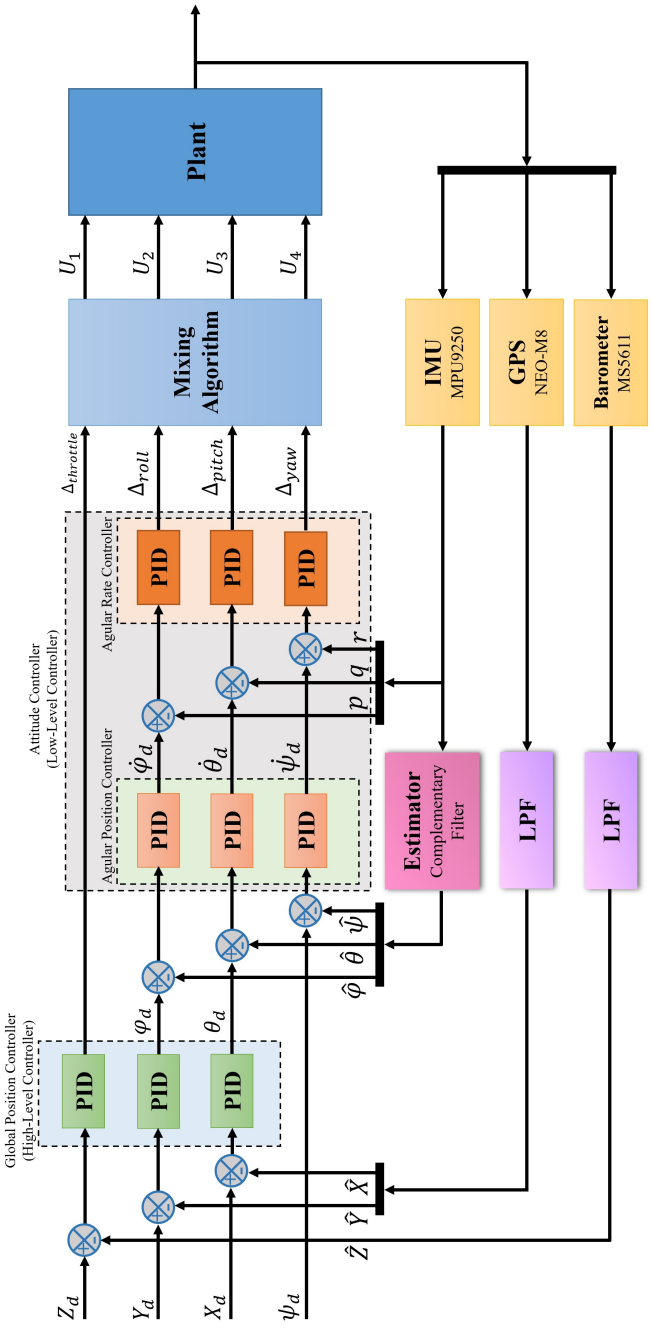

Figure 4.1: Cascaded PID controller scheme used for low-level and high-level control of the collision-resilient foldable quadcopter– in experiment and simulation

4.2.1 Plant Design

Primarily, the plant is the dynamic model developed in Section 3.2 and added as a MATLAB function block in Simulink. The MATLAB function code for the plant is attached in Appendix B.

4.2.2 Sensor Modeling

Sensor measurements are naturally noisy, and the structural and electronic elements of the sensor cause inevitable delays in data transmission. Therefore, to model the utilized sensors, it is rational to add noise and delay to the dynamic plant output in the simulations. To do so, first, the magnitude of noise for different sensors should be approximately measured.

Sensor noises can be modeled as a normal distribution function (Gaussian distribution) with a specific standard deviation σ [73]. An average value for standard deviation can be determined by recording the sensor output for limited numbers of data and averaging them. Then, by choosing the noise amplitude equal to 2σ in the simulation, $> 95\%$ confidence in sensor data distribution is guaranteed (Figure 4.2).

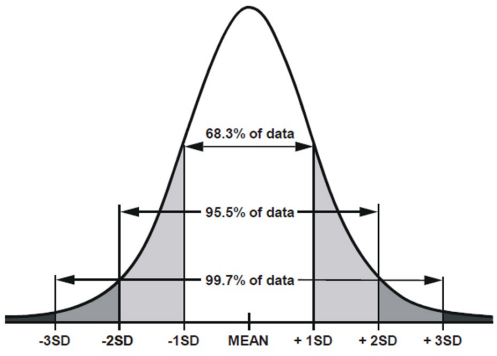

Figure 4.2: Normal sensor data distribution

In an experiment, 1200 data are read from each DOF of the IMU sensor (MPU9250). Then, averaging the resulting data vector and finding the standard deviation for each DOF, a noise magnitude is estimated for the sensors. Figure 8 illustrates the result of this experiment for the 3-DOF accelerometer and gyroscope.

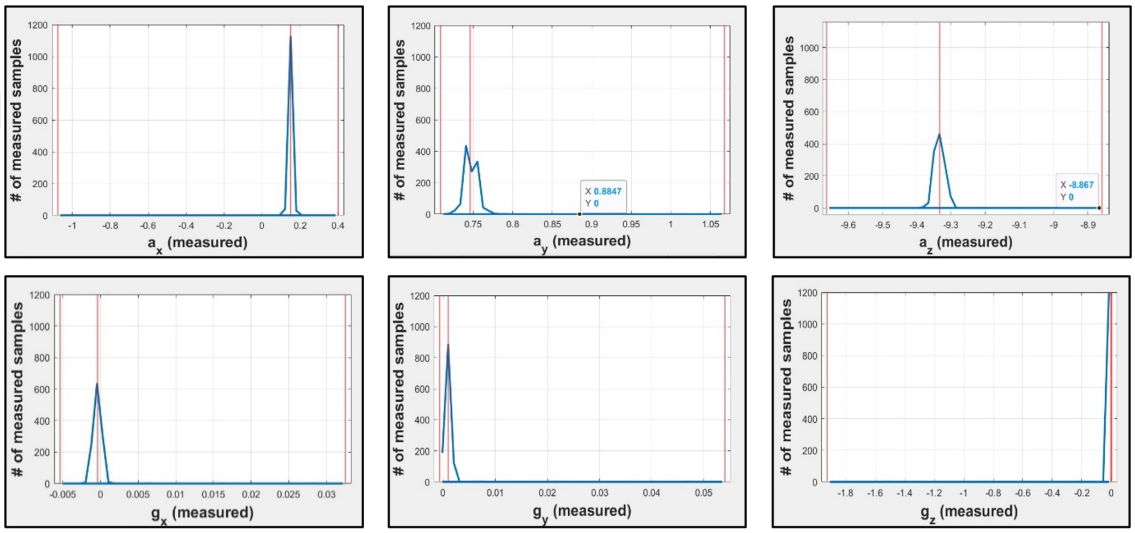

Figure 4.3: Sensor reading results for 3DOF accelerometer and 3DOF gyroscope

The sensor delay value is available in the sensor package datasheet [74]. For MPU9250, the delay for the 3DOF gyroscope is specified as $2.9ms$, where this value for the 3DOF accelerometer is equal to $5.9ms$. Figure 4.4 demonstrates the sensor modeling for the accel and gyro sensors by adding noise and delay to the measured data in Simulink.

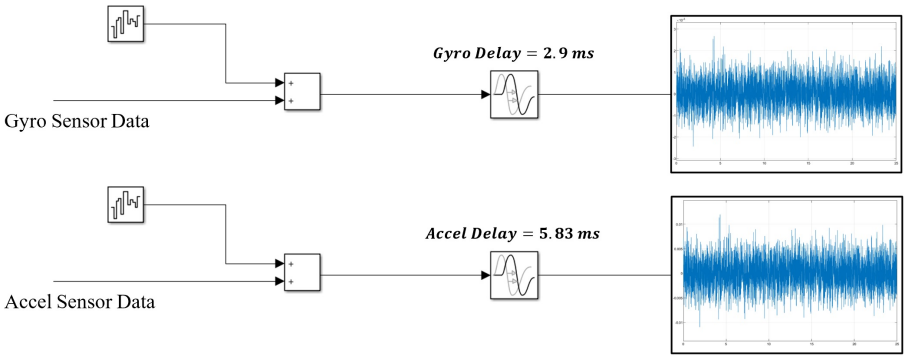

Figure 4.4: Modeling MPU9250 IMU sensor output in Simulink

Table 4.1 lists the noise amplitudes and delay values used for the different sensors in the simulation in Simulink.

Table 4.1: Noise amplitude and delay time values used in the simulation

Sensor	**Model**	**Noise Amplitude [unit]**	**Delay** $[ms]$
Accelerometer	MPU9250	5×10^{-2} $[m/s^2]$	6
Gyroscope	MPU9250	2.5×10^{-3} $[rad/s]$	3
Barometer	MS5611	1.5×10^{-1} $[m]$	10
GPS	NEO-M8	1.5×10^{-1} $[m]$	100

4.2.3 Estimator Block

As a popular and computationally low-cost method, the *Complementary filter* is chosen to estimate the attitude of the system in both experiment and simulation. A complementary filter is basically a **L**ow-**P**ass **F**ilter (LPF) and a **H**igh-**P**ass **F**ilter (HPF) in parallel [75]. For a measured quantity of u by accelerometer and gyroscope sensors, the estimated value by the complementary filter $\hat{u}$ can be calculated as

$$\hat{u} = F(s)u_g + \bar{F}(s)u_a \tag{4.1}$$

where $F(s)$ represents the low-pass filter applied to the gyroscope reading and $\bar{F}(s)$ is the complementary high-pass filter applied to the accelerometer data such that

$$F(s) + \bar{F}(s) = 1 \tag{4.2}$$

Assigning a constant value $\alpha \in [0,\ 1]$ to $F(s)$, Equation 4.1 can be rewritten as

$$\hat{u} = \alpha\ u_g + (1 - \alpha)u_a \tag{4.3}$$

For MPU9250, the value of this constant is selected as $\alpha = 0.98$ in the simulation.

For the barometer and GPS sensors, a simple LPF is applied to reduce the noise and simulate the sluggish nature of these sensors. The code for the MATLAB function used as an estimator block in the simulation is attached in Appendix C.

It is good to mention that the use of Madgwick Filter [53] as an alternative for state estimation algorithm is investigated in this study. Although Madgwick filter demonstrates a more accurate and reliable state estimation, basic complementary filter is still preferred to be used in the real system due to its simplicity and easy implementation. Appendix D compares these two methods.

4.2.4 Cascaded PID Controllers

As mentioned earlier, the cascaded PID scheme is chosen as the attitude, altitude, and global position controller for the collision-resilient foldable quadcopter. On the real drone, the PID constants are determined using trial-and-error. However, in the simulation, PID controller blocks are used, and they are tuned step-by-step using the PID Tuner App in Simulink. The controllers are tuned in specific order shaping the overall control structure in the simulation environment.

4.2.4.1 Altitude (Elevation) PID Controller Tuning

The throttle command ($\Delta_{throttle}$), which controls the elevation (Z) of the quadcopter, is the main command for its flight and hovering because it determines how fast all rotors must turn in combined to exert the required thrust force for a specific maneuver. The other control commands (attitude control commands) are somehow added/subtracted to/from the throttle value (this will be discussed ahead in the mixing algorithm section).

Therefore, the altitude (Z) PID controller is the first controller to be tuned. This is done by assigning Δ_{roll}, Δ_{pitch}, and Δ_{yaw} values equal to $\mathbf{0}$ (around the

hovering point) and using the PID tuner app to specify the desired transient response characteristics of the Z-controller (The assumptions of section 3.2 are valid here). The following table shows the transient response characteristics and the PID constant values for the altitude controller.

Table 4.2: PID gains for altitude (Z) controller

PID Controller	**Transient Response Char.**		**PID Gains**		
	Rise Time (t_r)	Max. Overshoot	K_p	K_i	K_d
Altitude (Z)	1.0 [s]	10%	0.03526	0.004146	0.07363

4.2.4.2 Angular Rate Controller Tuning

After tuning the PID controller for altitude, the angular rate controllers for roll, pitch, and yaw angles are added to the control loop and tuned. These controllers are also tuned around the equilibrium point $[\dot{\phi}; \dot{\theta}; \dot{\psi}] = [0; 0; 0]$. The error signal input to these three rate controllers is the apparent angular rates $[p; q; r]$ measured directly from the gyro sensor subtracted from the desired angular rate, which is the output from the outer loop angular position controller. The output of these three PID controllers are the differential PWM values (Δ_{roll}, Δ_{pitch}, Δ_{yaw}) added/subtracted to/from the throttle value to control the quadcopter's attitude (the Euler angles).

The transient response characteristics used to tune the PID controllers and the resulting PID gains are shown in table 4.3.

Table 4.3: PID gains for angular rate ($\dot{\phi}; \dot{\theta}; \dot{\psi}$) controllers

PID Controller	**Transient Response Char.**		**PID Gains**		
	Rise Time (t_r)	Max. Overshoot	K_p	K_i	K_d
Roll rate $\dot{\phi}$	0.2 [s]	10%	0.007877	0.024234	0
Pitch rate $\dot{\theta}$	0.2 [s]	10%	0.008786	0.013915	0
Yaw rate $\dot{\psi}$	0.2 [s]	10%	0.022785	0.036088	0

It should be noted that a saturation block is used to limit the output signal of angular rate controllers in the simulation. Because, in reality, the control input to the system is limited. This also helps with keeping the system close to the equilibrium point around which the PID controllers are tuned.

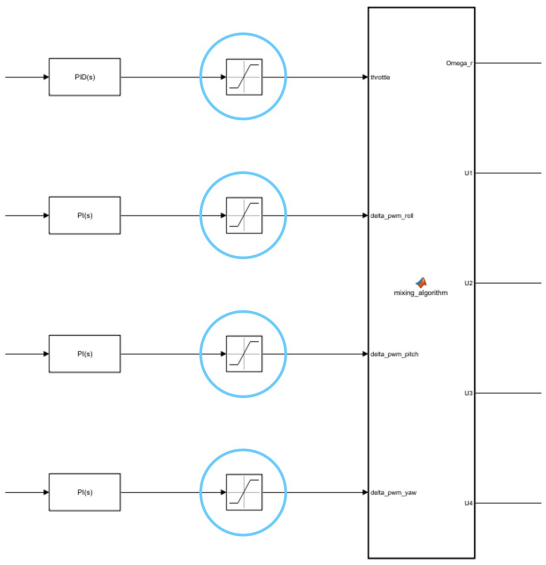

Figure 4.5: Use of saturation blocks to limit the final output of the controllers (control inputs)

4.2.4.3 Angular Position Controller Tuning

Having the angular rate PID controllers tuned, the outer loop PID controllers that are responsible for the Euler angle control of the quadcopter can be tuned now. The controllers are tuned around the equilibrium point $[\phi; \theta; \psi] = [0; 0; 0]$. The input error signals to these controllers are the subtraction of the estimated drone Euler angles (output of the complementary filter) from the desired Euler angles from the position controller (or a path-generation algorithm). Table 4.4 shows the transient response characteristics and the PID gains for the angular position controllers.

Table 4.4: PID gains for angular position $(\phi; \theta; \psi)$ controllers

PID Controller	**Transient Response Char.**		**PID Gains**		
	Rise Time (t_r)	Max. Overshoot	K_p	K_i	K_d
Roll rate $\dot{\phi}$	0.5 [s]	10%	5.2665	0.53648	0
Pitch rate $\dot{\theta}$	0.5 [s]	10%	5.0797	0.4539	0
Yaw rate $\dot{\psi}$	0.5 [s]	10%	5.1967	0.48043	0

4.2.4.4 Global Position $[X, Y]$ PID Controller Tuning

From the quadcopter's dynamic equations (Equation 3.5), it can be concluded that an isolated deviation in pitch angle while the quadcopter is hovering makes it move in the body-fixed rotating x-direction (a negative pitch results in positive x movement). Therefore, for small pitch angles, if the drone's heading (yaw ψ) is equal to zero, it moves along the global inertial X-axis. The same relationship is true for the roll deviation and Y-axis movement if the heading is $\psi = 0$ (a positive deviation in roll angle results in positive Y movement).

Therefore, keeping the heading equal to zero, the quadcopter can reach any $[X, Y]$ position in the global coordinate by controlling only the roll and pitch angles. Using this to keep the simulations simpler, two PID controllers are placed in the outer loop to perform the high-level position control for the drone.

The input error signals are the subtraction of the estimated position of the quadcopter in the global frame, $\hat{X}$ and $\hat{Y}$, from their desired value, X_d and Y_d, provided by the defined path for the quadcopter. The output of the controllers are the desired pitch (θ_d) and roll (ϕ_d) angles supposed to take the drone along the defined X_d and Y_d, respectively. The transient response characteristics and the PID gain values for the global $[X, Y]$ position controllers are arranged in Table 4.5.

Table 4.5: PID gains for global $[X, Y]$ position controllers

PID Controller	**Transient Response Char.**		**PID Gains**		
	Rise Time (t_r)	Max. Overshoot	K_p	K_i	K_d
Y **position**	0.5 [s]	10%	0.022556	0	0.0732
X **position**	0.5 [s]	10%	0.016	0	0.06549

4.2.5 Mixing Algorithm Block

What this algorithm fundamentally does, is that it relates the control inputs from the altitude and attitude controllers to the input forces and moments to the quadcopter system (Equation 3.9). The code for this MATLAB function block is

mentioned in Appendix E.

4.2.6 Flight Simulation

Having the low-level (attitude) and high-level (global position) controllers tuned, the following scenario is designed to test the modeled quadcopter's performance in simulation. In this simulation, starting from rest at point $[0; 0; 0]$, the quadcopter goes up to the height of $-2m$ (Z-axis is downwards) and goes through a $10m \times 10m$ square path (with a change in altitude up to $-4m$ at point $[X, Y] = [10, 10]$). Figures 4.6 and 4.7 demonstrate the path tracking of the developed control loop in Simulink.

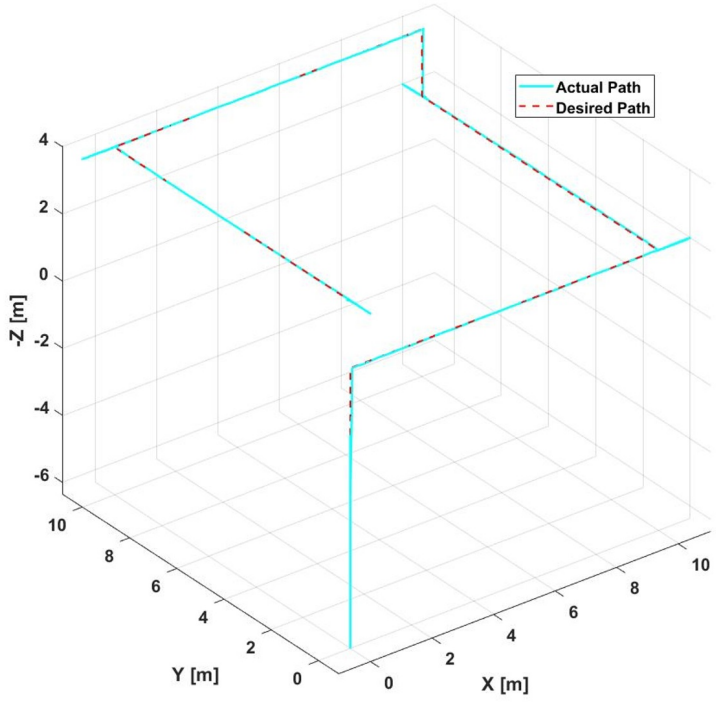

Figure 4.6: 3D view of the tracked $10m \times 10m$ square path in the simulation environment (MATLAB Simulink)

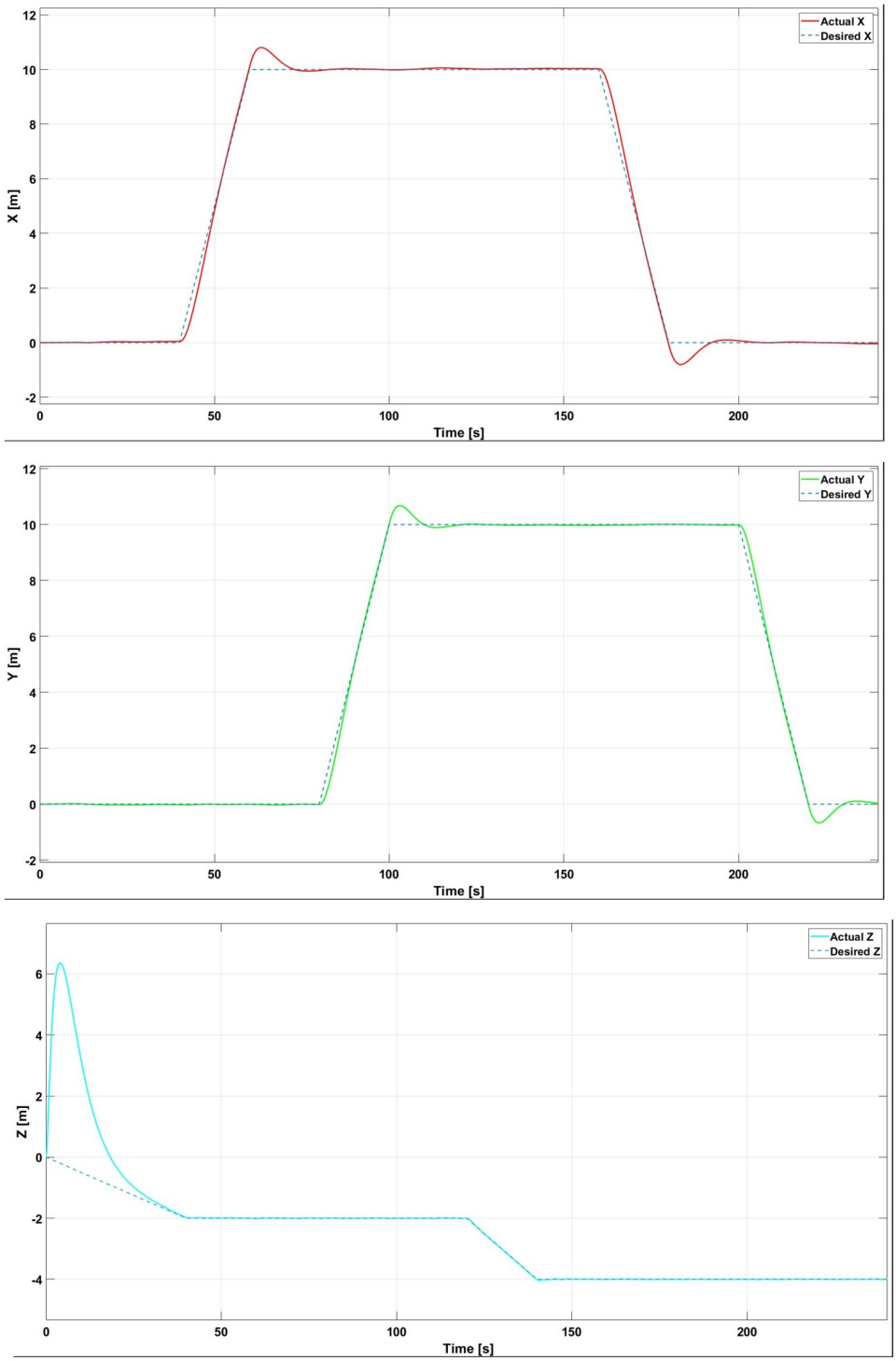

Figure 4.7: $[X, Y, Z]$ view of the tracked $10m \times 10m$ square path in the simulation environment (MATLAB Simulink)

There is a considerable deviation in the Z direction at the beginning. The result for this deviation is that no grounds are defined in the simulation environment, and as soon as the simulation begins, the drone falls due to the gravitational force. It takes about $40sec$ for the quadcopter to compensate for this initial fall and to catch up with the defined path. Obviously, to this point, the overall tracking error is significant. After this point ($t = 40s$), the average tracking error is about $error = 0.2314m$. Considering the $\pm 15cm$ noise amplitude defined for the GPS and Barometer sensors and the dimensions of the defined path, this much tracking error is acceptable.

It is good to mention that in another simulation, in order to observe the effect of noise on the path tracking accuracy of the designed cascaded PID controllers, the amplitude of the noises are doubled and the drone's movement is simulated along the same path defined above. The results of this simulation shows that the tracking error after $t = 40s$ is equal to $0.2949m$, which is a 27% increase.

4.2.7 Adding the Soft Impact Block

This block, augmented to the control loop in Simulink as a MATLAB function block, receives the robots state $\bar{X}$ and outputs the visco-elastic contact forces and moments (Equations 3.15-3.28) exerted by the obstacle (a vertical wall in $X = 10m$ in the case of this simulation) to the foldable drone (plant block) as disturbance forces and moments. Figure 4.8 depicts how the impact block is added to the control loop. The code for this block is attached in Appendix F.

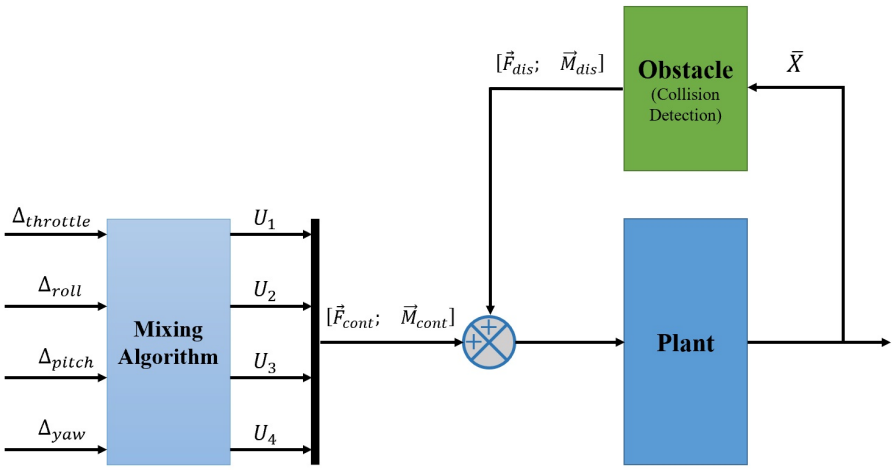

Figure 4.8: Soft impact detection algorithm added as the obstacle block to the simulation environment

4.2.8 Soft Impact Simulation

The scenario designed to simulate the soft impact in Simulink starts with a stabilizing period in the beginning. During this phase, which takes long about $30sec$, the drone starts at the origin $[X, Y, Z] = [0, 0, 0]$ at rest and stabilizes at an altitude of $-2m$ ($\downarrow +Z$). Then, maintaining the altitude, the foldable quadcopter approaches a vertical wall at $X = 10m$ with a constant linear velocity controlled by the cascaded PID controllers. Figure 4.9 illustrates the tracked path by the quadcopter in the simulation. As mentioned before, the initial drop is due to the sudden effect of gravity as the simulation starts.

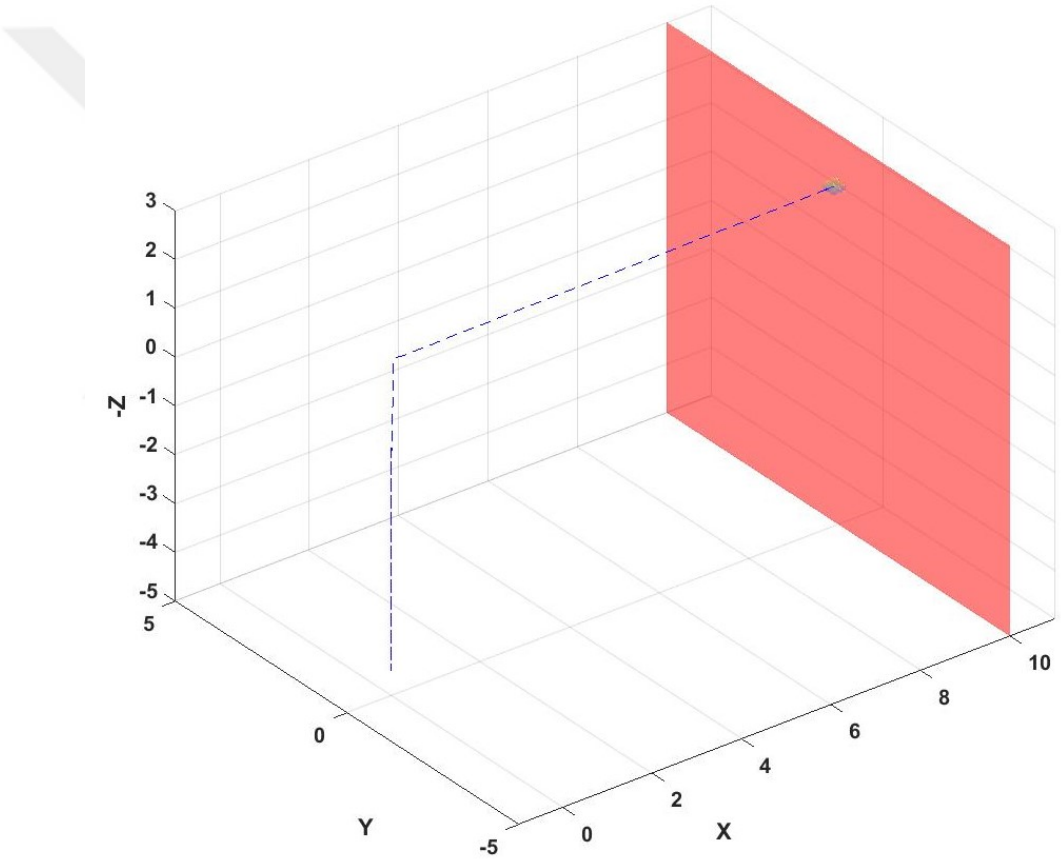

Figure 4.9: Collision scenario designed in Simulink (the size of the quadcopter is exaggerated)

Table 4.6: Visco-elastic and friction coefficients used in the simulations

Parameter	Definition	Value [unit]
K	Spring Ratio	1.8×10^4
D	Damping Ratio	8
n	Nonlinear Spring Exponent	2
μ_d	Coefficient of Dynamic Friction	0.3

The values for the constants in Equations 3.10 and 3.23 are assigned as shown in Table 4.6. The values are found with trial and error in the simulations such that the obtained results imitate the actual collision experiments in [2].

Figures 4.10 and 4.11 display the position and velocity of the quadcopter during a collision to the above-mentioned wall along X-axis direction (direction of the collision). The line along $X = 9.88m$ indicated the point where the bumpers start contacting the obstacle.

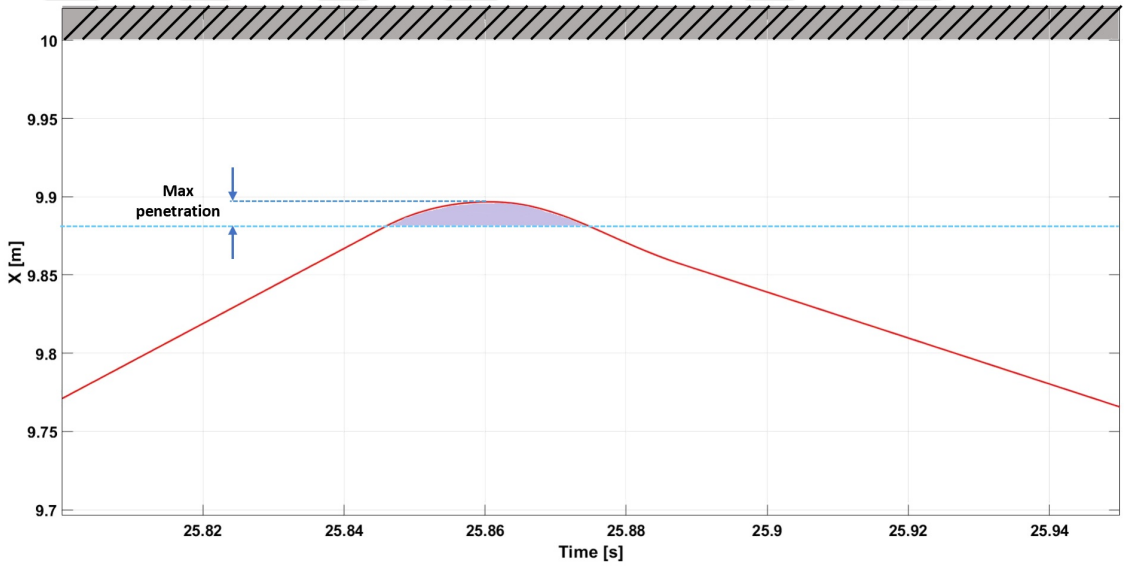

Figure 4.10: The movement of the compliant foldable quadcopter during the impact - purple area shows the penetration in the direction of impact

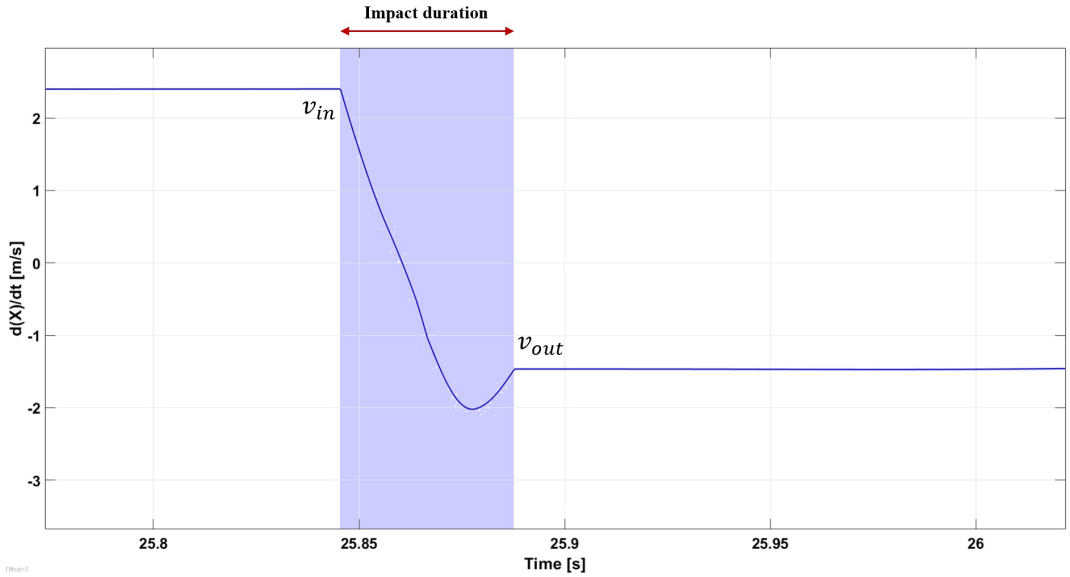

Figure 4.11: Velocity change of the CoG of the quadcopter (X-direction) during the soft impact - shaded area indicates the impact duration

Coefficient **o**f **R**estitution (CoR) is a measure indicating the amount of elasticity of a collision that is the ratio of the magnitude of the colliding object's velocity an instant before the collision to the magnitude of the velocity an instant after the impact ends ($|v_{out}|/|v_{in}|$). In other words, CoR shows how much of the colliding object's kinetic energy is dissipated during the impact. The more elastic (or let's say, visco-elastic) the impact, the more energy dissipates during the impact, therefore, the lesser the CoR [37].

The simulation scenario above is repeated for different velocities, and the CoR is calculated for each case. The diagram in Figure 4.12 depicts the CoR for different collision velocities. It can be concluded that for $v_{in} > 2m/s$, the result is close to the experiments in [2], where the CoR of the collision for *MiniCoRe* is equal to 0.556 (Figure 4.13)

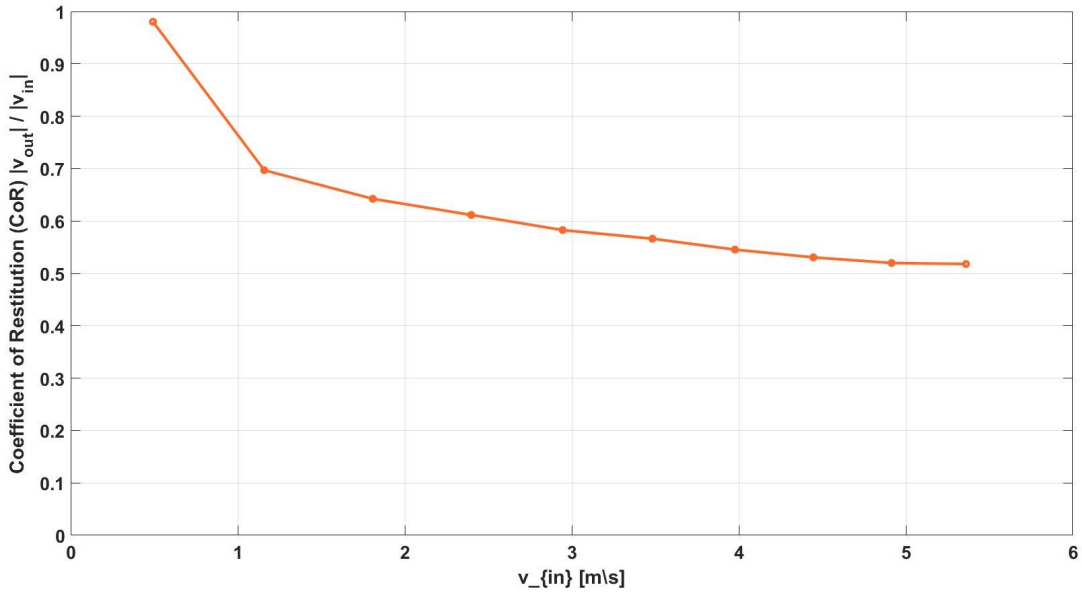

Figure 4.12: CoR for different impact velocities for the simulations in *Simulink*

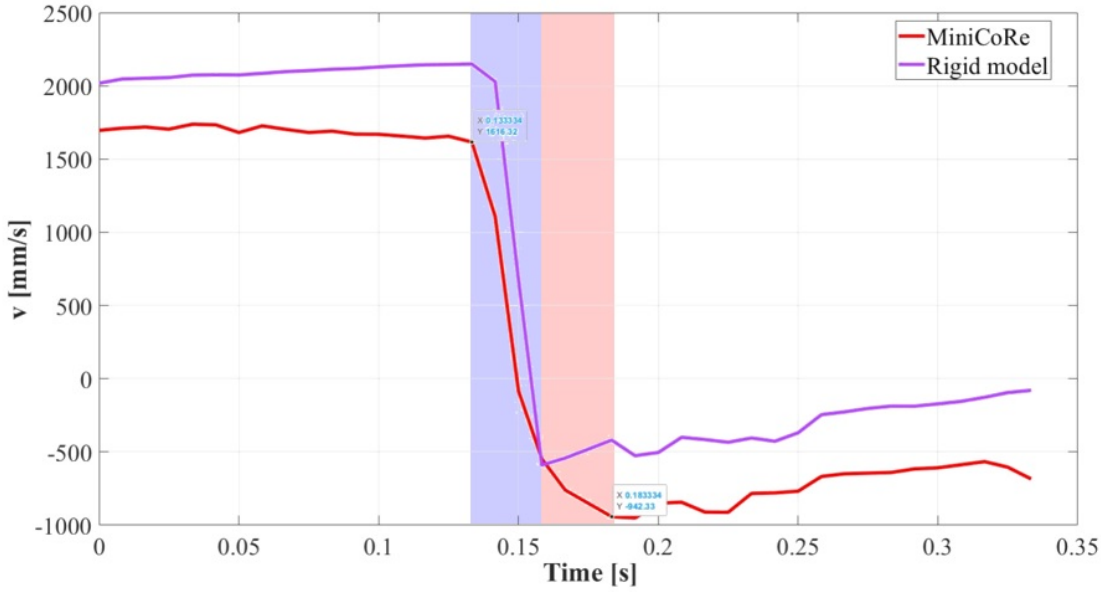

Figure 4.13: Velocity change during the collision for *MiniCoRe* and its rigid equivalent [2]

It should be noted that impact simulations are also conducted for higher collision velocities, but since the quadcopter exceeds its linearized range about the hovering point to provide such velocities, the controller cannot compensate for the quadcopter's attitude (the controller is not even capable of compensation for the drone's altitude at $Z = -2\,m$ before the impact). This results in an immediate loss of control and crash in the simulations after the impact. For an approaching velocity of $9\,m/s$ the approaching pitch angle is equal to $\theta = -0.63\,rad = -36°$ which is far from the linearized hovering range of $\pm 5°$ (this constraint is also used in the real drone), while the approach pitch angle for a $5\,m/s$ impact velocity is equal to $\theta = -0.07\,rad = -4°$ which is in the maximum linearized working range.

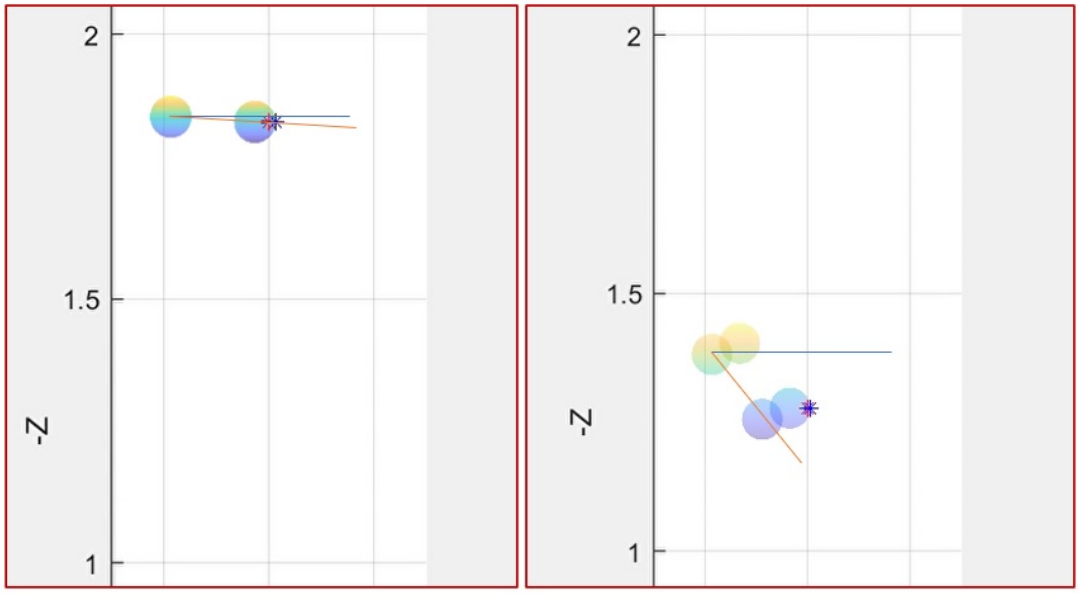

Figure 4.14: Approching pitch angle before the impact for $v_{in} = 9\,m/s$ (right), and $v_{in} = 5\,m/s$ (left)

4.2.9 Discussion on the Ordinary Differential Equation (ODE) Solving

Equation 3.5 introduces a set of **O**rdinary **D**ifferential **E**quations (ODEs) as the EoMs of the system. To solve this set of ODEs for the system in simulations, *Simulink* uses various types of nonstiff and stiff solvers. The most popular solver used in Simulink is `ode45`, a nonstiff solver based on the fourth-order and fifth-order Runge-Kutta method.

Due to the stiff nature of the ODEs in Equation 3.5, especially during the impacts where large impact forces are generated during a small time span, `ode45` is not a suitable choice for solving them. As an alternative, `ode15s`, which is a variable-order variable-step stiff ODE solver, is recommended to solve this set of ODEs. `ode15s` uses first-order to fifth-order Runge-Kutta ODE solvers with variable time steps during the solution interval to overcome the stiffness of the EoMs. The fact that Simulink itself selects `ode15s` as the default solver for this simulation, when the solver type and time step settings are set to "*Automatic*", approves that use of `ode15s` is a rational choice in the simulations. It is obvious that using `ode15s` instead of `ode45` considerably increases the simulation time.

It is recommended to set the maximum time step of the stiff solver to $10^{-3}[s]$. This helps the simulation be closer to the real drone in which the fastest loop on the microprocessor runs at $1kHz$.

4.3 Soft Impact Simulation in Gazebo

In parallel to the development of a dynamic model for the collision-resilient foldable quadcopter in MATLAB, another model is developed in this project to use *Gazebo*, the popular robotic simulation environment in ROS, to simulate the impact of the compliant drone to a solid object.

The ROS libraries developed in this investigation are based on an earlier library developed by *Kumar Robotics Lab* at the University of Pennsylvania [76]. The cascaded PID controller structure, used in both the real quadcopter and the MATLAB simulations, is added to this library. In addition to the control scheme, a **U**nified **R**obotics **D**escription **F**ormat (URDF) model is developed based on the foldable quadcopter's real model.

To build a .xacro URDF model, an importable file format to ROS, the first step is to design the quadcopter in a CAD software like *SolidWorks*. Figure 4.15 illustrates the designed model in SolidWorks based on the real quadcopter design and its components.

Figure 4.15: *SolidWorks* model for the collision-resilient foldable quadcopter

Although the developed model provides an accurate estimate of the mass and inertia properties of the system, it is an extremely heavy model to be added to the Gazebo simulation environment due to the excessive number of components and details, resulting in a disproportionate number of meshes introduced by the dynamic solver. It is proved that a simplified equivalent model, consisting of fewer components that each represent the mass and inertia properties of many parts, gives an almost accurate representation of the system in finite element solvers [77]. Therefore, using the mass and inertia properties extracted from the model above, an equivalent model is developed with fewer components to be imported as a URDF file (Figure 4.15).

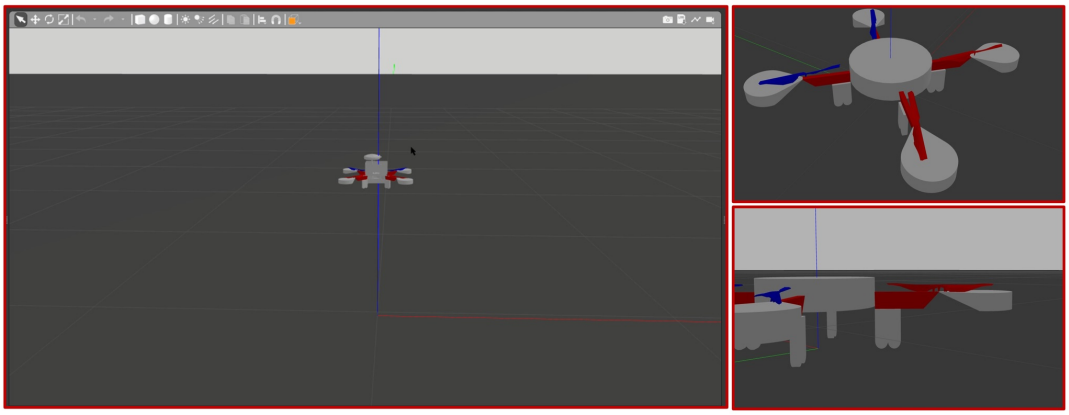

Figure 4.16: Simplified model imported as a URDF file in Gazebo

Gazebo libraries do not allow to add compliance to the bodies, and all bodies are defined as rigid bodies in the simulation. However, these rigid bodies relate to each other by *fixes* and *joints* that can exhibit the required compliance for a soft impact. Defining the mating relation between the quadcopter arms and its main body by joints with torsional stiffness, damping, and friction properties, the visco-elastic impact forces can be generated in the joints. Therefore, by using an equivalent and close-to-reality model in Gazebo, the soft impact of a compliant quadcopter can be modeled. Using the values for the joint properties and the surface friction tabulated below, some collision scenarios are simulated in Gazebo.

Table 4.7: Joint and surface properties used to simulate the soft impact in Gazebo

Parameters		**Value**
Joint Properties	Torsional Spring Stiffness	1 [N.m/rad]
	Friction	0.0001 [N.m]
	Damping	0.0005 [N.m.s/rad]
Surface Properties	Friction	0.01

Employing the above values for the joint and surface parameters, a set of experiments are conducted in Gazebo where the developed model collides with a rigid wall while it is being navigated directly toward it with different velocities ranging from $0.5m/s$ to $5m/s$. The COR values calculated for these impacts are close to the results of the experiments for *MiniCoRe* in [2].

It should be mentioned that the motor-propeller properties required for the *RotorX* plugin in Gazebo are acquired from UIUC Propeller Database [78].

Figure 4.17 illustrates the "COR vs. v_{in}" diagram for conducted impact experiments in Gazebo. Figure 4.18 shows the folded arms of the quadcopter towards the body in a collision simulation.

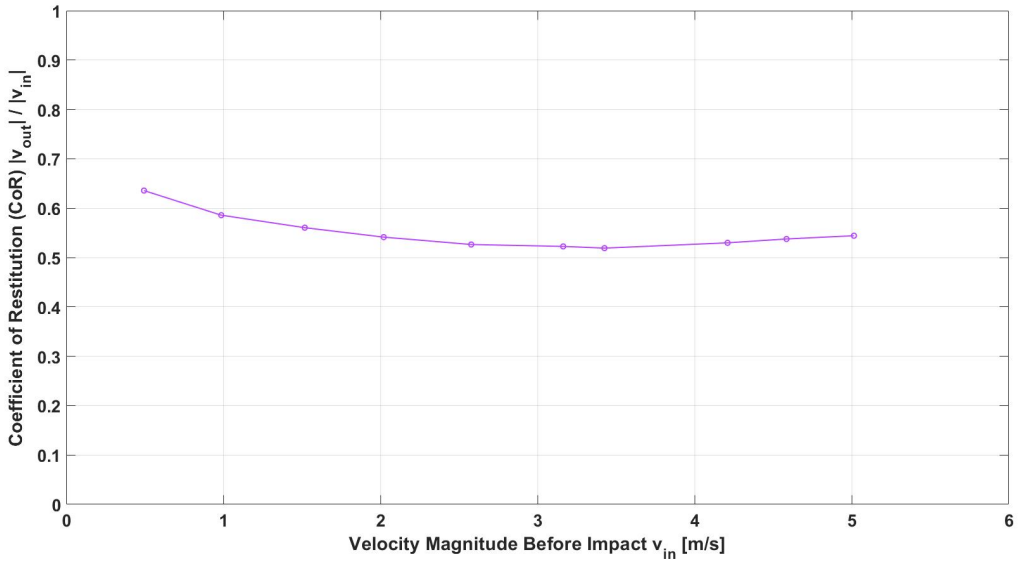

Figure 4.17: COR values for different impact velocities in Gazebo

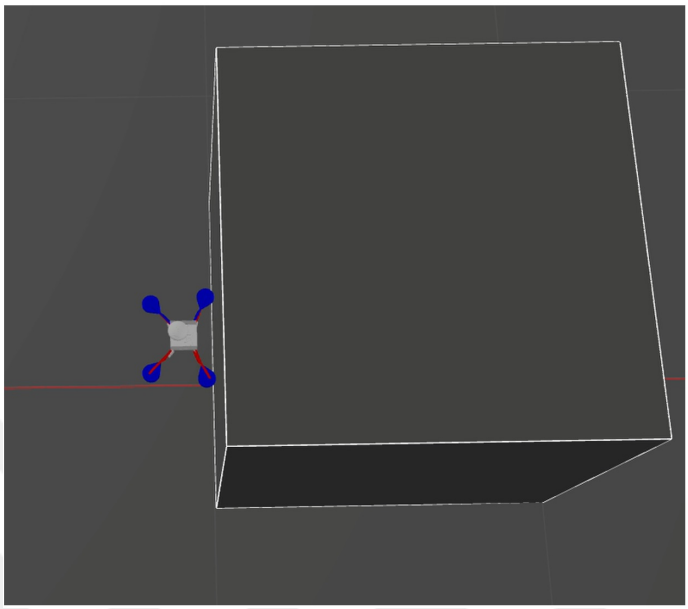

Figure 4.18: Soft impact (into a fixed, rigid block) simulation for the compliant foldable quadcopter in Gazebo

Simulations in *Gazebo* are not continued for higher velocities, since at $v_{in} = 5m/s$ the arms are being folded backwards and the central housing touches the block.

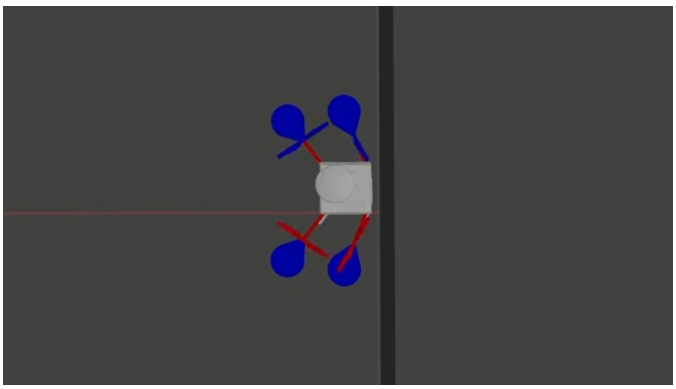

Figure 4.19: Simulation failure due to backward folding of the arms and central housing contact to the block at $v_{in} = 5\,m/s$ in *Gazebo*

4.4 CoR Comparison for the Simulations and Experiments

The following diagram plots the CoR for the simulations in MATLAB *Simulink* and ROS *Gazebo* alongside the scattered data from 15 impact experiments for MiniCoRe. The average CoR for the experiments is illustrated by the dotted horizontal blue line of $CoR = 0.556$.

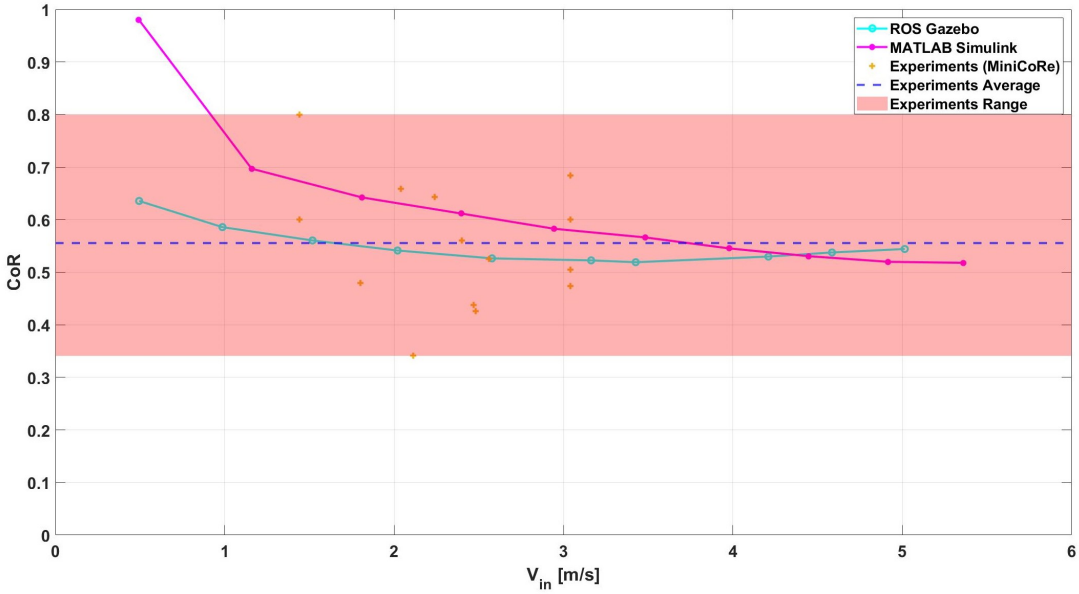

Figure 4.20: Comparison between the simulation results in MATLAB *Simulink*, ROS Gazebo, and experiments for *MiniCoRe*

4.5 Collision-Resilient Foldable Body in Outdoor Field Experiments

The final mission defined for the collision-resilient foldable quadcopter was to make multiple of these drones fly at the same time and inspect a predefined target by providing images from different angles. The following figures show the quadcopter on its outdoor missions.

Figure 4.21: Single collision-resilient foldable quadcopter during a field inspection mission

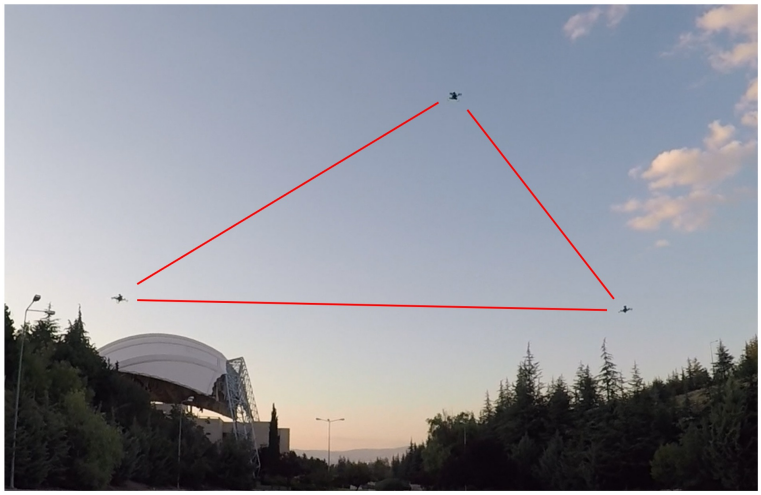

Figure 4.22: Three collision-resilient foldable quadcopters during a field inspection mission

4.5.1 Protective Bumpers and Landing Gear

In many outdoor experiments during the design and test process of the quadcopter, the drone encountered numerous failures and crashes due to communication loss or sudden disarming. While some of these crashes were significant, and sometimes the drone fell off from an altitude around $50m$ high, the damage to the drone body was minor and repairable (by changing the bumpers, landing gear, or one or two propeller(s)) in most cases. But it is for certain that in none of the crashes, the main flight controller or the motors damaged.

Figure 4.23 demonstrates a collision-resilient foldable quadcopter that has crashed and fallen off from an approximate altitude of $20m$ with minor fractures in landing gears as all four protective bumpers are intact. It should be mentioned that no damage to the body is recorded in a normal landing sequence.

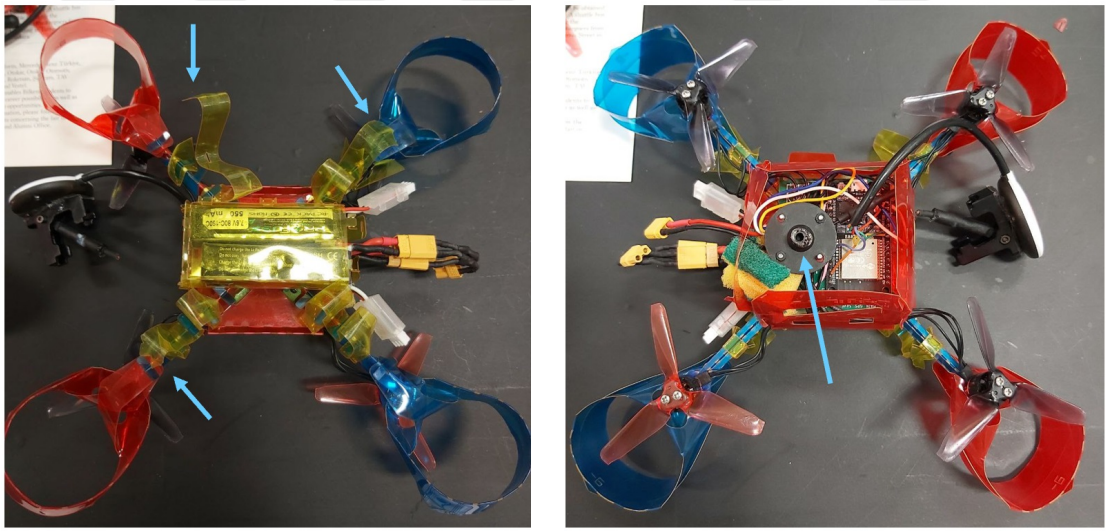

Figure 4.23: Damaged collision-resilient foldable body after a $20m$ high free fall

4.5.2 FPV Camera Streaming During the Inspection Flight

The FPV camera video streaming is elaborated in Section 2.3.8, and it is illustrated in the schematic in Figure 2.21. The following image shows a snapshot of the video streaming for three FPV cams on the collision-resilient foldable drones during the inspection mission.

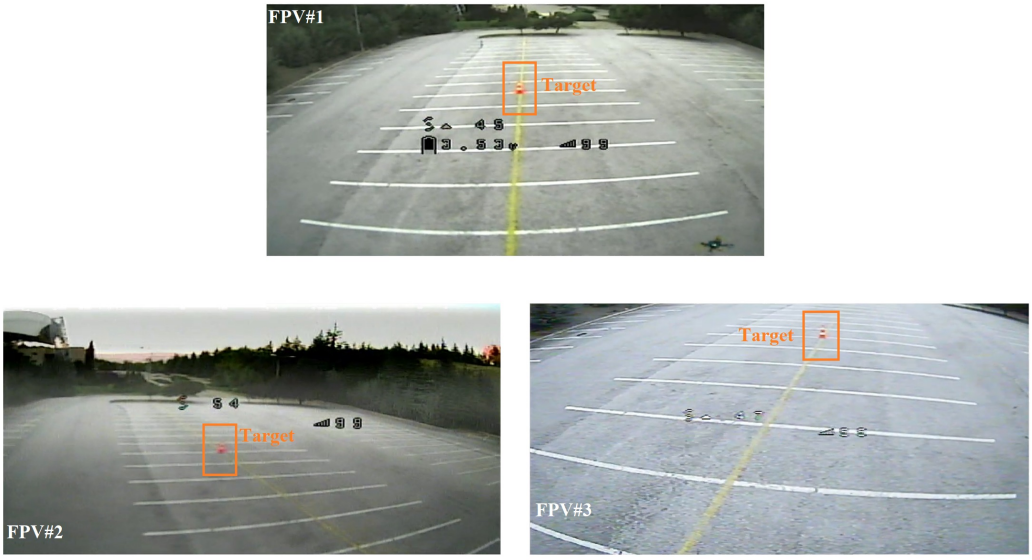

Figure 4.24: FPV cam streaming from three collision-resilient foldable drones during an inspection mission

Chapter 5

Conclusion and Future Works

The Collision-Resilient Foldable Micro Quadcopter is designed, so it meets the predefined criteria for the project. With a maximum weight of $220g$ and a maximum dimension of $23cm \times 23cm$, the drone remains below the $300g$ and $30cm \times 30cm$ threshold. The protective bumpers and the landing gear cushion the quadcopter during physical contact with surrounding obstacles and faces. The foldable body frame constitutes only 16% of the total drone weight that demonstrates the superiority of the design in comparison to the existing MAV models (the inexpensive fabrication and compliance of the body frame should be taken into account making any comparison with other drones of the kind).

The field experiments indicate incredible aerodynamic stability of the foldable body during its inspection missions. The compliance of the body protects the drone even in serious crashes. The protective parts are designed and fastened to the body in a way that makes it easy for the user to replace them in the field as they fail. Further improvements can be made in body design by strengthening the weak parts prone to failure in impacts. The performance of different origami structures and inclination angles for the bumpers can also be investigated.

The designed cascaded PID control scheme provides a reasonable low-level and high-level control for the quadcopter, both in simulation and experiment. Easy

implementation and tuning, low computational cost, and acceptable error bounds for the defined missions are advantages of using PID controllers in this project. Integrating additional sensors (like optical flow sensors) and employment of more accurate sensor fusion algorithms (advanced complementary filters like Mahony and Magdwick, or EKF) will further improve the state estimation in the drone and, consequently, the control performance. With an improved controller, the post-impact trajectory planning for the soft drone can be investigated.

The simulation results demonstrate accurate and close-to-reality modeling of the soft impact, both in MATLAB Simulink and ROS Gazebo. The dynamic model can be enhanced by adding a reliable Pseudo-Coulomb dry friction model to the surface dynamics instead of using a coefficient of dynamic friction at all times. Repeating the collision simulations in the vertical direction can also be utilized to simulate the soft landing of the quadcopter on its compliant landing gears. These simulations can be later utilized to optimize the landing sequence of the quadcopter. Also, to extract an exact dynamic model of the compliant body during collisions, a combination of **F**inite **E**lement **M**ethods (FEM) and **M**achine **L**earning (ML) can be hired.

Bibliography

[1] L. Dilaveroğlu, *Collision Resilient Foldable Micro Aerial Robot*. PhD thesis, Bilkent Universitesi (Turkey), 2019.

[2] L. Dilaveroğlu and O. Özcan, "Minicore: A miniature, foldable, collision resilient quadcopter," in *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft)*, pp. 176–181, IEEE, 2020.

[3] S. Bouabdallah, P. Murrieri, and R. Siegwart, "Towards autonomous indoor micro vtol," *Autonomous robots*, vol. 18, no. 2, pp. 171–183, 2005.

[4] S. Bouabdallah, "Design and control of quadrotors with application to autonomous flying," tech. rep., Epfl, 2007.

[5] D. Floreano and R. J. Wood, "Science, technology and the future of small autonomous drones," *nature*, vol. 521, no. 7553, pp. 460–466, 2015.

[6] P. M. Kornatowski, M. Feroskhan, W. J. Stewart, and D. Floreano, "A morphing cargo drone for safe flight in proximity of humans," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4233–4240, 2020.

[7] A. Mahmud, *Development of Collision Resilient Drone for Flying in Cluttered Environment*. West Virginia University, 2021.

[8] Z. Liu and K. Karydis, "Toward impact-resilient quadrotor design, collision characterization and recovery control to sustain flight after collisions," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 183–189, IEEE, 2021.

[9] P. Sareh, P. Chermprayong, M. Emmanuelli, H. Nadeem, and M. Kovac, "Rotorigami: A rotary origami protective system for robotic rotorcraft," *Science Robotics*, vol. 3, no. 22, p. eaah5228, 2018.

[10] P. M. Kornatowski, S. Mintchev, and D. Floreano, "An origami-inspired cargo drone," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6855–6862, IEEE, 2017.

[11] A. M. Mehta, D. Rus, K. Mohta, Y. Mulgaonkar, M. Piccoli, and V. Kumar, "A scripted printable quadrotor: Rapid design and fabrication of a folded mav," in *Robotics Research*, pp. 203–219, Springer, 2016.

[12] J. Shu and P. Chirarattananon, "A quadrotor with an origami-inspired protective mechanism," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3820–3827, 2019.

[13] D. Yang, S. Mishra, D. M. Aukes, and W. Zhang, "Design, planning, and control of an origami-inspired foldable quadrotor," in *2019 American Control Conference (ACC)*, pp. 2551–2556, IEEE, 2019.

[14] S. Mintchev, S. de Rivaz, and D. Floreano, "Insect-inspired mechanical resilience for multicopters," *IEEE Robotics and automation letters*, vol. 2, no. 3, pp. 1248–1255, 2017.

[15] R. de Azambuja, H. Fouad, Y. Bouteiller, C. Sol, and G. Beltrame, "When being soft makes you tough: A collision-resilient quadcopter inspired by arthropods' exoskeletons," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 7854–7860, IEEE, 2022.

[16] S. T. Bui, Q. K. Luu, D. Q. Nguyen, N. D. M. Le, G. Loianno, and V. A. Ho, "Tombo propeller: Bio-inspired deformable structure toward collision-accommodated control for drones," *arXiv preprint arXiv:2202.07177*, 2022.

[17] F. Ruiz, B. Arrue, and A. Ollero, "Sophie: Soft and flexible aerial vehicle for physical interaction with the environment," *arXiv preprint arXiv:2205.12883*, 2022.

[18] S. Mintchev, L. Daler, G. L'Eplattenier, L. Saint-Raymond, and D. Floreano, "Foldable and self-deployable pocket sized quadrotor," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2190–2195, IEEE, 2015.

[19] D. Falanga, K. Kleber, S. Mintchev, D. Floreano, and D. Scaramuzza, "The foldable drone: A morphing quadrotor that can squeeze and fly," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 209–216, 2018.

[20] N. Zhao, Y. Luo, H. Deng, and Y. Shen, "The deformable quad-rotor: Design, kinematics and dynamics characterization, and flight performance validation," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2391–2396, IEEE, 2017.

[21] S. H. Derrouaoui, Y. Bouzid, M. Guiatni, and I. Dib, "A comprehensive review on reconfigurable drones: Classification, characteristics, design and control technologies," *Unmanned Systems*, vol. 10, no. 01, pp. 3–29, 2022.

[22] `https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32e_esp32-wroom-32ue_datasheet_en.pdf`. Accessed: 2022-08-17.

[23] `https://geprc.com/product/stable-f411/`. Accessed: 2022-08-17.

[24] `https://content.u-blox.com/sites/default/files/NEO-M8-FW3_DataSheet_UBX-15031086.pdf`. Accessed: 2022-08-17.

[25] S. Bouabdallah and R. Siegwart, "Full control of a quadrotor," in *2007 IEEE/RSJ international conference on intelligent robots and systems*, pp. 153–158, Ieee, 2007.

[26] A. Nemati and M. Kumar, "Modeling and control of a single axis tilting quadcopter," in *2014 American Control Conference*, pp. 3077–3082, IEEE, 2014.

[27] R. Ji, J. Ma, and S. S. Ge, "Modeling and control of a tilting quadcopter," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 4, pp. 2823–2834, 2019.

[28] D. Brescianini and R. D'Andrea, "Design, modeling and control of an omni-directional aerial vehicle," in *2016 IEEE international conference on robotics and automation (ICRA)*, pp. 3261–3266, IEEE, 2016.

[29] Y. Long and D. J. Cappelleri, "Omnicopter: A novel overactuated micro aerial vehicle," in *Advances in Mechanisms, Robotics and Design Education and Research*, pp. 215–226, Springer, 2013.

[30] `https://ardupilot.org/copter/docs/ground-effect-compensation.html`. Accessed: 2022-08-18.

[31] A. Matus-Vargas, G. Rodriguez-Gomez, and J. Martinez-Carranza, "Ground effect on rotorcraft unmanned aerial vehicles: A review," *Intelligent Service Robotics*, vol. 14, no. 1, pp. 99–118, 2021.

[32] P. Wei, S. N. Chan, S. Lee, and Z. Kong, "Mitigating ground effect on mini quadcopters with model reference adaptive control," *International Journal of Intelligent Robotics and Applications*, vol. 3, no. 3, pp. 283–297, 2019.

[33] X. He, G. Kou, M. Calaf, and K. K. Leang, "In-ground-effect modeling and nonlinear-disturbance observer for multirotor unmanned aerial vehicle control," *Journal of Dynamic Systems, Measurement, and Control*, vol. 141, no. 7, 2019.

[34] `https://ardupilot.org/copter/docs/common-throttle-based-notch.html`. Accessed: 2022-08-18.

[35] W. Goldsmith, *The theory and physical behaviour of colliding solids*. Dover Publ., 1999.

[36] P. Flores and H. M. Lankarani, *Contact force models for multibody dynamics*, vol. 226. Springer, 2016.

[37] K. H. Hunt and F. R. E. Crossley, "Coefficient of restitution interpreted as damping in vibroimpact," 1975.

[38] W. J. Stronge, "Rigid body collisions with friction," *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, vol. 431, no. 1881, pp. 169–181, 1990.

[39] J. Keller, "Impact with friction," 1986.

[40] W. Stronge, "Smooth dynamics of oblique impact with friction," *International journal of impact engineering*, vol. 51, pp. 36–49, 2013.

[41] M. Askari, *Design, control, modeling, and gait analysis in miniature foldable robotics*. PhD thesis, Bilkent Universitesi (Turkey), 2018.

[42] S. Bouabdallah and R. Siegwart, "Backstepping and sliding-mode techniques applied to an indoor micro quadrotor," in *Proceedings of the 2005 IEEE international conference on robotics and automation*, pp. 2247–2252, IEEE, 2005.

[43] R. Mahony, V. Kumar, and P. Corke, "Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor," *IEEE Robotics and Automation magazine*, vol. 19, no. 3, pp. 20–32, 2012.

[44] S. Bouabdallah, A. Noth, and R. 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)*, vol. 3, pp. 2451–2456, IEEE, 2004.

[45] L. M. Argentim, W. C. Rezende, P. E. Santos, and R. A. Aguiar, "Pid, lqr and lqr-pid on a quadcopter platform," in *2013 International Conference on Informatics, Electronics and Vision (ICIEV)*, pp. 1–6, IEEE, 2013.

[46] `https://ardupilot.org/copter/docs/tuning.html#overview`. Accessed: 2022-08-22.

[47] `https://github.com/betaflight/betaflight/wiki/PID-Tuning-Guide`. Accessed: 2022-08-22.

[48] N. Bao, X. Ran, Z. Wu, Y. Xue, and K. Wang, "Research on attitude controller of quadcopter based on cascade pid control algorithm," in *2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, pp. 1493–1497, IEEE, 2017.

[49] E. A. Paiva, J. C. Soto, J. A. Salinas, and W. Ipanaqué, "Modeling and pid cascade control of a quadcopter for trajectory tracking," in *2015 CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*, pp. 809–815, IEEE, 2015.

[50] M. A. Hsieh, A. Cowley, V. Kumar, and C. J. Taylor, "Maintaining network connectivity and performance in robot teams," *Journal of field robotics*, vol. 25, no. 1-2, pp. 111–131, 2008.

[51] E. Stump, A. Jadbabaie, and V. Kumar, "Connectivity management in mobile robot teams," in *2008 IEEE international conference on robotics and automation*, pp. 1525–1530, IEEE, 2008.

[52] R. Mahony, T. Hamel, and J.-M. Pflimlin, "Complementary filter design on the special orthogonal group so (3)," in *Proceedings of the 44th IEEE Conference on Decision and Control*, pp. 1477–1484, IEEE, 2005.

[53] S. Madgwick *et al.*, "An efficient orientation filter for inertial and inertial/-magnetic sensor arrays," *Report x-io and University of Bristol (UK)*, vol. 25, pp. 113–118, 2010.

[54] K. D. Sebesta and N. Boizot, "A real-time adaptive high-gain ekf, applied to a quadcopter inertial navigation system," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 1, pp. 495–503, 2013.

[55] S. A. Ludwig and K. D. Burnham, "Comparison of euler estimate using extended kalman filter, madgwick and mahony on quadcopter flight data," in *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 1236–1241, IEEE, 2018.

[56] T. Lee, M. Leok, and N. H. McClamroch, "Control of complex maneuvers for a quadrotor uav using geometric methods on se (3)," *arXiv preprint arXiv:1003.2005*, 2010.

[57] T. Lee, M. Leok, and N. H. McClamroch, "Geometric tracking control of a quadrotor uav on se (3)," in *49th IEEE conference on decision and control (CDC)*, pp. 5420–5425, IEEE, 2010.

[58] A. Noormohammadi-Asl, O. Esrafilian, M. A. Arzati, and H. D. Taghirad, "System identification and h-based control of quadrotor attitude," *Mechanical Systems and Signal Processing*, vol. 135, p. 106358, 2020.

[59] F. Ruggiero, M. A. Trujillo, R. Cano, H. Ascorbe, A. Viguria, C. Peréz, V. Lippiello, A. Ollero, and B. Siciliano, "A multilayer control for multirotor uavs equipped with a servo robot arm," in *2015 IEEE international conference on robotics and automation (ICRA)*, pp. 4014–4020, IEEE, 2015.

[60] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE international conference on robotics and automation*, pp. 2520–2525, IEEE, 2011.

[61] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," *The International Journal of Robotics Research*, vol. 31, no. 5, pp. 664–674, 2012.

[62] M. Hehn and R. D'Andrea, "Quadrocopter trajectory generation and control," *IFAC proceedings Volumes*, vol. 44, no. 1, pp. 1485–1491, 2011.

[63] M. Jun and R. D'Andrea, "Path planning for unmanned aerial vehicles in uncertain and adversarial environments," in *Cooperative control: models, applications and algorithms*, pp. 95–110, Springer, 2003.

[64] D. Mellinger, A. Kushleyev, and V. Kumar, "Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams," in *2012 IEEE international conference on robotics and automation*, pp. 477–483, IEEE, 2012.

[65] F. Caballero, L. Merino, J. Ferruz, and A. Ollero, "Vision-based odometry and slam for medium and high altitude flying uavs," *Journal of Intelligent and Robotic Systems*, vol. 54, no. 1, pp. 137–161, 2009.

[66] M. Müller, S. Lupashin, and R. D'Andrea, "Quadrocopter ball juggling," in *2011 IEEE/RSJ international conference on Intelligent Robots and Systems*, pp. 5113–5120, IEEE, 2011.

[67] R. Ritz, M. W. Müller, M. Hehn, and R. D'Andrea, "Cooperative quadrocopter ball throwing and catching," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4972–4978, IEEE, 2012.

[68] P. Schmuck and M. Chli, "Multi-uav collaborative monocular slam," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3863–3870, IEEE, 2017.

[69] M. Blösch, S. Weiss, D. Scaramuzza, and R. Siegwart, "Vision based mav navigation in unknown and unstructured environments," in *2010 IEEE International Conference on Robotics and Automation*, pp. 21–28, IEEE, 2010.

[70] S. Weiss, D. Scaramuzza, and R. Siegwart, "Monocular-slam–based navigation for autonomous micro helicopters in gps-denied environments," *Journal of Field Robotics*, vol. 28, no. 6, pp. 854–874, 2011.

[71] E. Eraslan and Y. Yildiz, "Modeling and adaptive control of flexible quadrotor uavs," in *2021 60th IEEE Conference on Decision and Control (CDC)*, pp. 1783–1788, IEEE, 2021.

[72] E. Eraslan and Y. Yildiz, "Modeling, control and human-in-the-loop stability analysis of an elastic quadrotor," *arXiv preprint arXiv:2012.05747*, 2020.

[73] J. P. Bentley, *Principles of measurement systems*. Pearson education, 2005.

[74] `https://invensense.tdk.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf`. Accessed: 2022-08-22.

[75] P. Narkhede, S. Poddar, R. Walambe, G. Ghinea, and K. Kotecha, "Cascaded complementary filter architecture for sensor fusion in attitude estimation," *Sensors*, vol. 21, no. 6, p. 1937, 2021.

[76] https://github.com/KumarRobotics.

[77] Y. Zhang, Y. Huang, Z. Li, K. Liang, K. Cao, and Y. Guo, "A simplified fe modeling strategy for the drop process simulation analysis of light and small drone," *Aerospace*, vol. 8, no. 12, p. 387, 2021.

[78] J.B. Brandt, R.W. Deters, G.K. Ananda, O.D. Dantsker, and M.S. Selig, UIUC Propeller Database, Vols 1-4, University of Illinois at Urbana-Champaign, retrieved from https://m-selig.ae.illinois.edu/props/propDB.html.

Appendix A

Obtaining the Linear Coefficients of Thrust and Drag

The motors and propellers used in micro-scale drone projects are mainly developed in the hobby industry and lack detailed datasheets. Hence, the motor and propeller coefficients required for simulation and controller design purposes are not available. In this section, an experiment is designed to investigate the static relationship between the input PWM voltage to the BrushLess DC (BLDC) motor drivers (known as ESC: Electronic Speed Controller) and the output lift and drag forces.

It should be noticed that the actuation system in this project consists of three components working in series and feeding each other to generate the lift (perpendicular to the propeller surface) and drag (parallel to the propeller surface) forces. Quadcopter's main controller generates PWM signals that are being fed to ESC. ESC converts this DC signal into a proper 3-phase AC and feeds it to the BLDC motor. The output rotational speed of the loaded motor determines the magnitude of the lift and drag forces generated in the attached propeller (knowing the aerodynamic characteristics of the propeller).

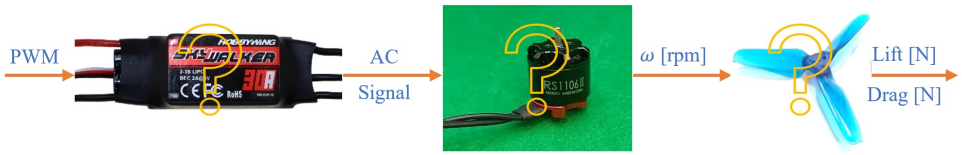

Figure A.1: Actuation mechanism in collision-resilient micro quadcopter

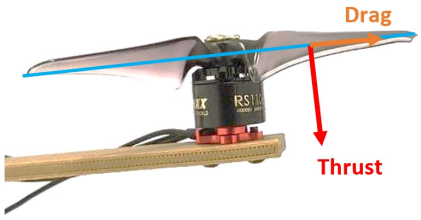

Figure A.2: Lift (thrust) and drag forces definition on propeller

To measure the lift and drag forces generated by the actuation system, a lever mechanism is designed to hold the BLDC motor on one side in a horizontal direction and exert the same force on a scale in a vertical direction on the other side. Figure A.3 illustrates the test setup for this experiment.

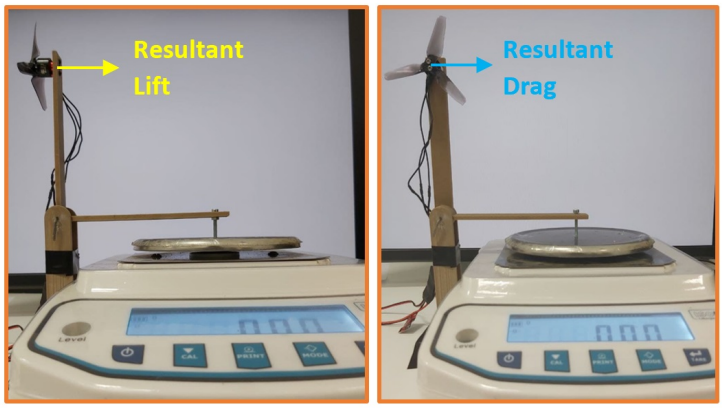

Figure A.3: Setup for static lift and drag force coefficients experiment

In general, the lift and drag forces generated by a constant-pitch (a propeller that has a constant angle of attack (AoA) along its blade) relates to its rotational speed as

$$L = C_T \cdot \omega^2 \tag{A.1}$$

$$D = C_D \cdot \omega^2 \tag{A.2}$$

where, C_T and C_D are lift and drag force coefficients, respectively. The above equation may change for a variable-pitch blade. For a BLDC motor, the relation between the input voltage (or PWM signal duty-cycle) and the output rotational speed is represented as

$$\omega = (KV) \cdot (pwm[\%]) \tag{A.3}$$

Therefore, (A.1) and (A.2) can be rewritten as

$$L = C_T \cdot \omega^2 = C_T \cdot (KV \cdot pwm)^2 = (C_T \cdot KV)^2 \cdot (pwm)^2 = k_T \cdot (pwm)^2 \tag{A.4}$$

$$D = C_D \cdot \omega^2 = C_D \cdot (KV \cdot pwm)^2 = (C_D \cdot KV)^2 \cdot (pwm)^2 = k_D \cdot (pwm)^2 \tag{A.5}$$

Equations (A.4) and (A.5) can be linear instead of quadratic for variable-pitch blades.

Experiments are done on two different motors and seven different propellers to find the k_T and k_D values for them and decide which motor-propeller set produces more thrust.

Table A.1: Motor and Propeller types used in the thrust and drag coefficient experiment

Motor #	**Model**
1	*Emax* RS1106ii 6000KV
2	*Emax* RS1106ii 7500KV
Propeller #	**Model – [Diameter[inches]*Pitch*Blade]**
1	*Emax* AVAN Mini – 3.0*2.4*3
2	*Emax* AVAN Rush – 2.5*X*3
3	*Emax* AVAN Blur – 2.0*X*3
4	*Emax* AVAN - 2.0*X*4
5	*PCK* HBRC - 2.5*X*5
6	*GEMFAN* (rotorX) RX2535 – 2.5*3.5*2
7	*GEMFAN* (rotorX) RX2035 – 2.0*3.5*4

Results of the experiments are illustrated in plots in figure A.4. It can be concluded from the diagrams that propeller#1 produces more lift in both motors. Although the 7500KV BLDC motor produces a slightly higher lift than the 6000KV version, the 6000KV version works at a lower temperature, making it a better candidate to serve as an actuator in our system. Also, it can be seen that the 3-inch 3-blade propeller *Emax* AVAN Mini produces the highest lift for a specific PWM input amongst all.

From the plots in figure A.4, it can be easily concluded that a linear function $L = k_T.(pwm)$ can be fitted on the experimental data with high confidence. Doing such for the selected BLDC motor and propeller set, the following equations are derived for lift and drag forces with respect to the input PWM ratio

$$\begin{cases} L = k_T \cdot (pwm) = \mathbf{1.586} \times (pwm) \\ D = k_D \cdot (pwm) = \mathbf{0.0915} \times (pwm) \end{cases} \tag{A.6}$$

Where (pwm) is in the interval [0,1] (0 means 0% duty-cycle and 1 means 100% duty-cycle for a 2S battery providing a continuous DC voltage of $7.9v$.

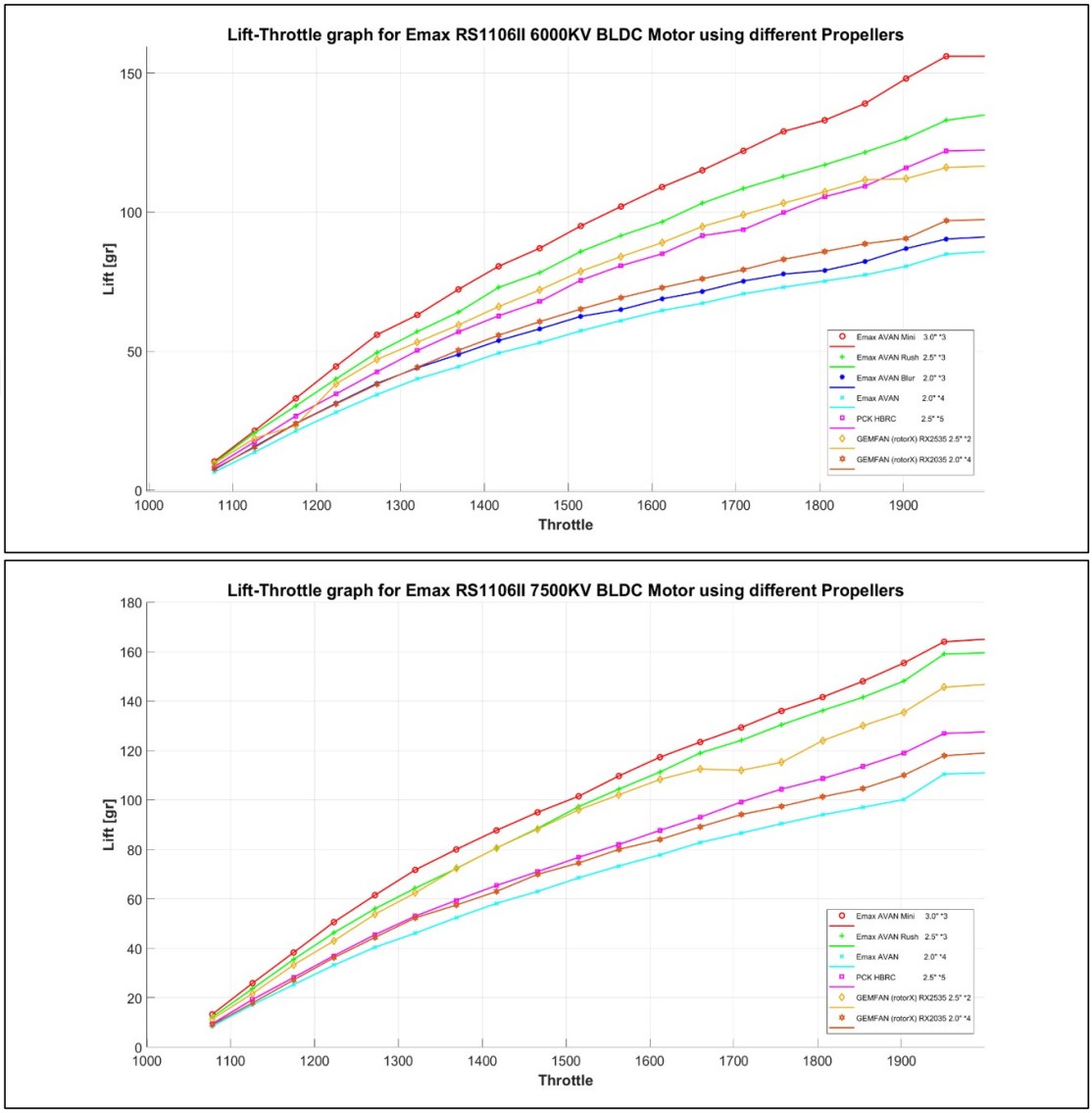

Figure A.4: Thrust force (Lift) test results for all motor-propeller combinations mentioned in table A.1

Appendix B

Plant Model - MATLAB Function in Simulink

```
1  function [ddX1, ddX2, ddX3, ddX4, ddX5, ddX6] = DDX(t, ...
       Omega_r, fx, fy, fz, mx, my, mz, U1, U2, U3, U4, X1, X2, ...
       X3, X4, X5, X6,dX1, dX2, dX3, dX4, dX5, dX6)
2
3  % this MATLAB function defines the Equations of Motion (EOMs ...
       for the test quadcopter regid model)
4  %
5  % the state vector is defined as
6  %
7  %   X = [phi theta psi z x y]    dX = d/dt(X)    ddX = ...
       d^2/dt^2(X)
8
9
10 % defining the mass, inertia, and geometric properties of the ...
       quadcopter
11 g    = 9.81;
12 m    = 191.3*10^-3;          % [kg]
13 I_xx = 419075.83*10^-9;      % [kg.m^2]
14 I_yy = 452305.61*10^-9;      % [kg.m^2]
15 I_zz = 830487.08*10^-9;      % [kg.m^2]
16 arm  = 80*10^-3;             % [m]
```

```
17 J_r  = 148.44*10^-9;        % [kg.m^2]
18
19 % defining intermediate variables
20 a1 = (I_yy - I_zz) / I_xx;
21 a2 = J_r / I_xx;
22 a3 = (I_zz - I_xx) / I_yy;
23 a4 = J_r / I_yy;
24 a5 = (I_xx - I_yy) / I_zz;
25 b1 = arm / I_xx;
26 b2 = arm / I_yy;
27 b3 =   1 / I_zz;
28 u_x = cos(X1)*sin(X2)*cos(X3) + sin(X1)*sin(X3);
29 u_y = cos(X1)*sin(X2)*sin(X3) - sin(X1)*cos(X3);
30
31 % % disarming point definition (if needed in simulation)
32 % t_disarm = 37.5; % [sec]
33 % if t < t_disarm
34 %     u1 = U1;
35 %     u2 = U2;
36 %     u3 = U3;
37 %     u4 = U4;
38 % else
39 %     u1 = 0;
40 %     u2 = 0;
41 %     u3 = 0;
42 %     u4 = 0;
43 % end
44
45 % system dynamics
46 % ([fx; fy; fz; mx; my; mz] is the resultant collision force ...
       and moment vector added to the EoMs as disturbance)
47 % the minus sign in x, y directions is to keep consistency ...
       for the downward z-axis direction
48 ddX1 = b1*(u2+mx);
49 ddX2 = b2*(u3+my);
50 ddX3 = b3*(u4+mz);
51 ddX4 = g - cos(X1)*cos(X2)*(1/m)*u1 - (1/m)*fz;
52 ddX5 = -u_x * (1/m)*u1 + (1/m)*fx;
53 ddX6 = -u_y * (1/m)*u1 + (1/m)*fy;
```

Appendix C

Complementary Filter Code - Estimator Block in Simulink

```
function [yaw, pitch, roll] = ...
    Complementary_Filter(gyroAngleX, gyroAngleY, gyroAngleZ, ...
    ax , ay , az)

accAngleX = atan(ay / sqrt(ax^2 + az^2));          % roll ...
    angle from accelerometer   [rad]
accAngleY = atan(-1 * ax / sqrt(ay^2 + az^2));     % pitch ...
    angle from accelerometer  [rad]

% Complementary Filter Logic
roll  = 0.98 * gyroAngleX + 0.02 * accAngleX;
pitch = 0.98 * gyroAngleY + 0.02 * accAngleY;
yaw   = gyroAngleZ;
```

Appendix D

State Estimation Methods

D.1 Sensor Fusion using Complementary Filter

a complementary filter is a combination of two low-pass and high-pass filters. The high-pass filter is used to remove the low-frequency noise of one sensor, and the low-pass filter is used to cancel out the high-frequency noise in the other. Then, the combination of the outputs of these sensors is represented as the estimated value for the measured quantity. In practice, for the case of a 6-DOF IMU (accelerometer + gyroscope), the simplest utilization of the complementary filter can be represented as the following linear weighted summation of the separately calculated angles from the accelerometer and the gyroscope.

$$\begin{cases} \text{Roll Angle } (\phi) = w_a \times \phi_{\text{accel}} + w_g \times \phi_{\text{gyro}} \\ \text{Pitch Angle } (\theta) = w_a \times \theta_{\text{accel}} + w_g \times \theta_{\text{gyro}} \end{cases} \tag{D.1}$$

where, w_a and w_g are the complementary weights and

$$
\begin{cases} \phi_{\text{accel}} = \tan^{-1}\left(+a_y/\sqrt{a_{x^2}+a_z^2}\right) \\ \theta_{\text{accel}} = \tan^{-1}\left(-a_x/\sqrt{a_y^2+a_z^2}\right) \end{cases}
$$
$$
\begin{cases} \phi_{\text{gyro}} = \int_0^t g_x dt \\ \theta_{\text{gyro}} = \int_0^t g_y dt \\ \psi_{\text{gyro}} = \int_0^t g_z dt \end{cases} \tag{D.2}
$$

$[a_x \ a_y \ a_z]^T$ and $[g_x \ g_y \ g_z]^T$ are the apparent acceleration $[m/s^2]$ and apparent angular velocity $[rad/s]$ vectors measured in the body frame, respectively.

As it is seen from the equations above, using a 6-DOF IMU, only roll and pitch angles can be determined by the complementary filter because the yaw motion cannot be detected and measured by the accelerometer. Therefore, using the simple complementary filter, there would be no sufficient yaw angle estimation and this is one of the drawbacks of this method.

During this project, an experiment is conducted to see the performance of the simple complementary filter using the accelerometer and gyroscope sensors of MPU9250. The values for the weights are chosen as $w_g = 0.98$ and $w_a = 0.02$ by observing the oscillations of the calculated values for the roll and pitch angles.

The results show that other than the drift in yaw angle, there is also significant drifts in roll and pitch angle. For example, as it is illustrated in Figure D.1, after some **harsh movements** of the sensor and positioning it in the same position as the start point, the complementary shows more than 90° turn in roll angle (the arrow is turned over and the bottom side (red side) is facing up).

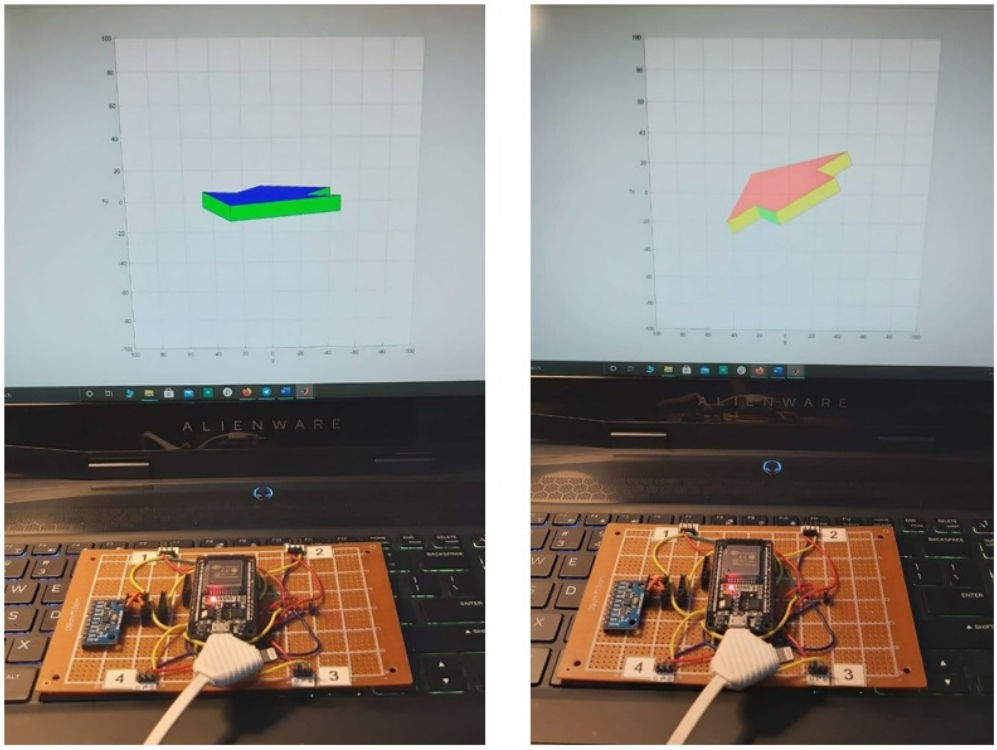

Figure D.1: Experiment on the result of the complementary sensor using the accelerometer and the gyroscope of MPU9250 (the blue side of the arrow represents the top of the sensor and the red side represents the bottom part)

There are some advanced and developed versions of the complementary filter in the literature that utilize data from all 9-DOF of the IMU sensor to cancel out the dynamic error in all angles during the estimation process. Madgwick Filter is one of these state estimation algorithms that its performance is investigated in the following section.

D.2 Sensor Fusion Madgwick Filter:

This estimator uses the Gradient Descent algorithm, which is an optimization method, to minimize the measurement errors of the sensors and obtain an exact attitude estimation of UAVs from Magnetic, Angular Rate, and Gyroscope (MARG) sensors (9-DOF IMUs like MPU9250). Figure D.2 illustrates the mathematical representation of the Madgwick IMU sensor fusion algorithm.

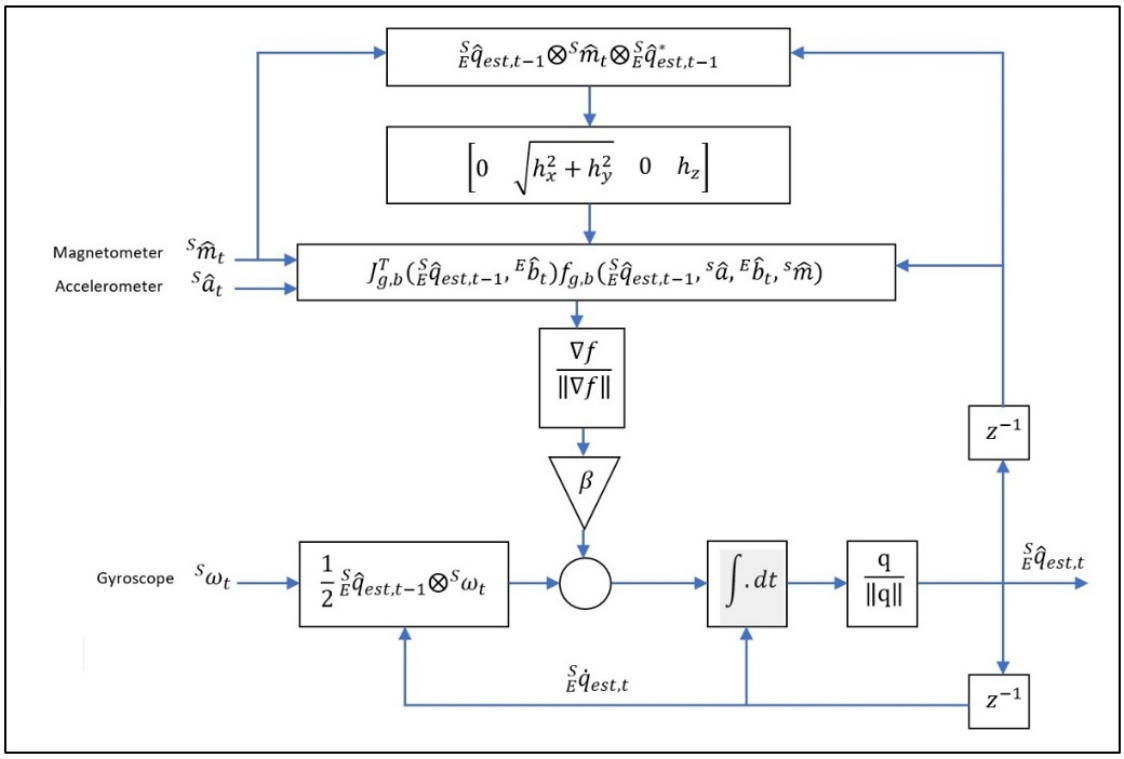

Figure D.2: Madgwick Filter's schematic mathematical representation

Apart from the use of the gradient descent algorithm to minimize the measurement errors in Madgwick Filter, another superiority of this algorithm is that it utilizes quaternions instead of the Euler angles to represent the rotations. In fact, the Madgwick filter tries to use the estimation of the quaternion rotation of the previous step and the normalized gradient of this estimation to calculate the current step quaternion vector. Using quaternions to represent the attitude of the system helps to avoid the singularities known as Gimbal Lock in the calculations.

It should be noticed that the Madgwick filter starts the estimation from an initial guess which is always equal to $\begin{bmatrix} \hat{\phi}_0 & \hat{\theta}_0 & \hat{\psi}_0 \end{bmatrix}^T = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$ in the case of UAV control. Therefore, it takes time for the algorithm to converge to the real attitude values measured by the sensor when it has just started to work. The speed of this convergence can be adjusted by setting the β coefficient in the algorithm. In this study, the value of this coefficient is set equal to $\sqrt{\frac{3}{4}}\left(40\left[\frac{\text{deg}}{\text{s}}\right] \times \frac{\pi}{180}\right)$. Assigning this value for , the Madgwick algorithm converges to the final value in about 2 seconds.

Figure D.3 shows the experimental results for the Madgwick filter at the start position at rest (after stabilization and convergence of the algorithm), and going back to the initial rest position after some harsh movements of the sensor.

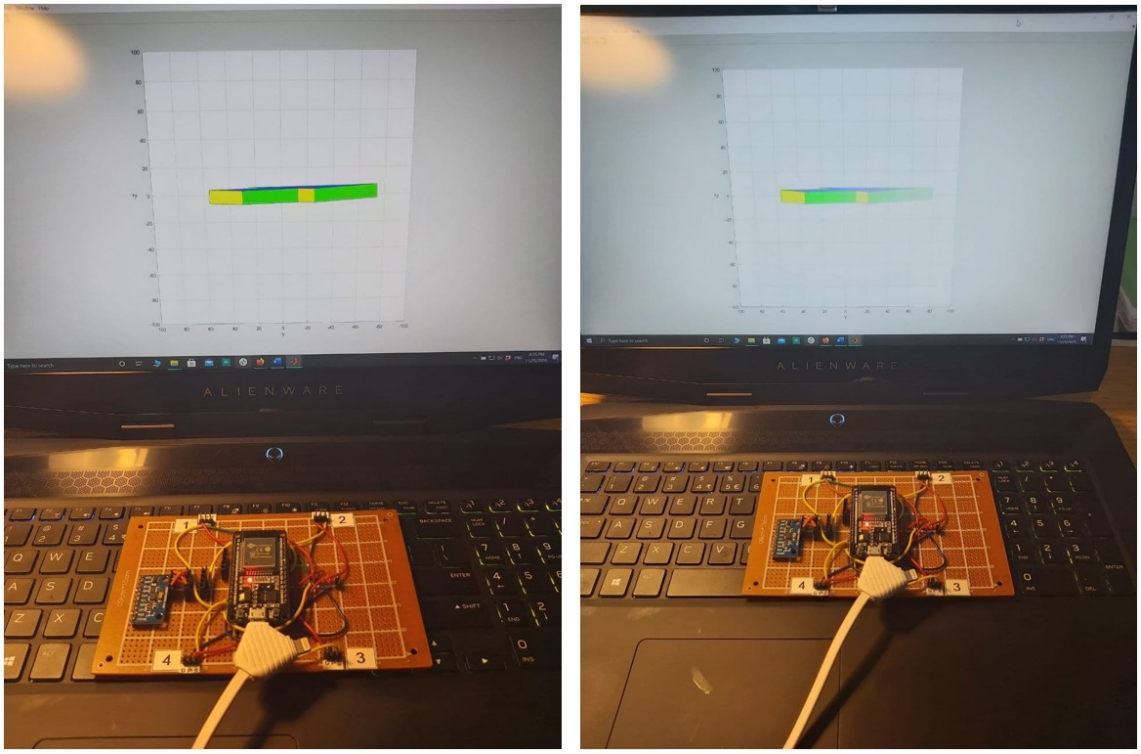

Figure D.3: Experiment on the result of the Madgwick filter using the MPU9250 MARG (the blue side of the arrow represents the top of the sensor and the arrow shows the same attitude at start (left) and end of the motion (right))

As it is seen in the figures, the orientation of the arrow, which illustrates the estimated attitude of the sensor using the Madgwick filter, is the same at the beginning of the test (when the sensor is at rest), and by the end of the test (when the sensor is again left at the rest in the same position and orientation) after some harsh maneuvers.

Appendix E

Mixing Algorithm MATLAB Function Block Code

```
function [Omega_r, U1, U2, U3, U4] = ...
    mixing_algorithm(throttle, Δ_pwm_roll, Δ_pwm_pitch, Δ_pwm_yaw)

cT = 1.589;    % [N / pwm_ratio]
cD = 0.09;     % [N / pwm_ratio]
pwm_to_rpm = 30000 * (2*pi) * (1/60);   % [(rad/s) / pwm_ratio]

motor1 = throttle + Δ_pwm_roll + Δ_pwm_pitch + Δ_pwm_yaw;
motor2 = throttle - Δ_pwm_roll + Δ_pwm_pitch - Δ_pwm_yaw;
motor3 = throttle - Δ_pwm_roll - Δ_pwm_pitch + Δ_pwm_yaw;
motor4 = throttle + Δ_pwm_roll - Δ_pwm_pitch - Δ_pwm_yaw;

U1 = cT * (motor1 + motor2 + motor3 + motor4);
U2 = cT * (motor1 - motor2 - motor3 + motor4);
U3 = cT * (motor1 + motor2 - motor3 - motor4);
U4 = cD * (motor1 - motor2 + motor3 - motor4);

Omega_r = abs( pwm_to_rpm * (motor1 - motor2 + motor3 - ...
    motor4) );
```

Appendix F

Collision Detection Block MATLAB Function Code

```
function [pen, d_pen, p_a, FX, FY, FZ, MX, MY, MZ] = ...
    Impact_Obj_1(pen_old, d_pen_old, p_a_old, dt, Phi, Theta, ...
    Psi, Z, X, Y, dPhi, dTheta, dPsi, dX, dY, dZ)

% this Simulink MATLAB function receives the quadcopter's CoG ...
    position and
% attitude as inputs and gives the resultant forces and ...
    moments on the
% drone if it has any Collision with the Object#1 (defined in ...
    the code)


%%% initializing the quadcopter's attitude (body-frame Euler ...
    angles) and
%%% CoG position
cog_pos = [X; Y; Z];    % [m]
rpy = [Phi; Theta; Psi];   % [rad]

%%% defining the linear and angular speed vectors ...
    (translational speed
%%% for the CoG and rotational speed for body  (Euler rates))
```

```
15 v_cog = [dX; dY; dZ];
16 omega = [dPhi; dTheta; dPsi];
17
18 %%% defining motor position vectors in body frame (cog -> mi) ...
       [m]
19 m1_b = [ 0.08; -0.08; 0];
20 m2_b = [ 0.08;  0.08; 0];
21 m3_b = [-0.08;  0.08; 0];
22 m4_b = [-0.08; -0.08; 0];
23
24 %%% defining the initial motor center positions in the ...
     inertial frame (I)
25 roll  = Phi;
26 pitch = Theta;
27 yaw   = Psi;
28
29 C_roll  = [1 0 0                    ;  0 cos(roll) sin(roll) ...
       ;  0 -sin(roll) cos(roll) ];
30 C_pitch = [cos(pitch) 0 -sin(pitch) ;  0 1 0                  ...
     ;  sin(pitch) 0 cos(pitch)];
31 C_yaw   = [cos(yaw) sin(yaw) 0      ;  -sin(yaw) cos(yaw) 0   ...
     ;  0 0 1                  ];
32
33 C_nb    = C_yaw * C_pitch * C_roll;
34
35 % calculating the CoG to motor center vector in Inertial ...
     frame (I)
36
37 cog_2_m_inI = zeros(3,4);
38
39 cog_2_m_inI(:,1) = C_nb * m1_b;
40 cog_2_m_inI(:,2) = C_nb * m2_b;
41 cog_2_m_inI(:,3) = C_nb * m3_b;
42 cog_2_m_inI(:,4) = C_nb * m4_b;
43
44 m = zeros(3,4);
45
46 m(:,1) = cog_pos + C_nb * m1_b;
47 m(:,2) = cog_pos + C_nb * m2_b;
48 m(:,3) = cog_pos + C_nb * m3_b;
```

```
49 m(:,4) = cog_pos + C_nb * m4_b;
50
51 %%% defining quadcopter's bumper radius
52 r  = 0.04;  % [m]
53
54 %%% defining the stiffness coefficient of the quadcopter arm ...
       contact point
55 C_k = 1.8e4;
56
57 %%% defining the damping coefficient of the quadcopter arm ...
       contact point
58 C_b = 8e0;
59
60 %%% defining dynamic friction coefficient of the colliding ...
       surface
61 mu_d = 0.3;
62
63 % defining the coefficients for Surface#1 - the general ...
       equation for a
64 % plane surface in inertial frame is considered to be like " ...
       aX + bY + cZ = -d
65 a1 = 1;
66 b1 = 0;
67 c1 = 0;
68 d1 = 10;
69 norm1 = sqrt(a1^2 + b1^2 + c1^2);           % calculating the ...
       normal vector 2nd norm
70 norm_vec = [a1/norm1; b1/norm1; c1/norm1];  % normalized ...
       normal vector of the plane
71
72 %%% checking for possible collision between the bumper ...
       spheres and a
73 %%% generally defined plane - Surface#1 (an obstacle like a wall)
74
75 d = zeros(1,4);
76
77 d(1) = abs(a1*m(1,1) + b1*m(2,1) + c1*m(3,1) - d1) / ...
       sqrt(a1^2 + b1^2 + c1^2);
78 d(2) = abs(a1*m(1,2) + b1*m(2,2) + c1*m(3,2) - d1) / ...
       sqrt(a1^2 + b1^2 + c1^2);
```

```
79 d(3) = abs(a1*m(1,3) + b1*m(2,3) + c1*m(3,3) - d1) / ...
       sqrt(a1^2 + b1^2 + c1^2);
80 d(4) = abs(a1*m(1,4) + b1*m(2,4) + c1*m(3,4) - d1) / ...
       sqrt(a1^2 + b1^2 + c1^2);
81
82 % d_1 = d(1);
83 % d_2 = d(2);
84 % d_3 = d(3);
85 % d_4 = d(4);
86
87 % checking if the bumper spheres are penetrated into the ...
       plane surface more
88 % or less than a radius long ( pen(i)>r or pen(i)<r )
89
90 % - finding the plane that is parallel to the colliding ...
       surface and
91 % passes from the colliding motor/propeller's center and ...
       calculating the
92 % distance between the CoG and the plane and the contact ...
       plane (if clause)
93
94 % - finding the direction of collision (checking if the arm ...
       direction is in
95 % the direction of the colliding surface's norm - ≤90 deg or ...
       > 90 deg)
96
97 % parameter initialization
98 cog_2_contact_plane = zeros(1,4);
99 cog_2_motor_center_plane = zeros(1,4);
100 arm_plane_normal_angle = zeros(1,4);
101 arm_angle_flag = zeros(1,4);
102
103 for j= 1:4
104
105     if d(j) ≤ r
106         a1_prime = a1;
107         b1_prime = b1;
108         c1_prime = c1;
109         d1_prime = a1_prime*m(1,j) + b1_prime*m(2,j) + ...
               c1_prime*m(3,j);
```

```
110         cog_2_contact_plane(j) = abs(a1*cog_pos(1) + ...
               b1*cog_pos(2) + c1*cog_pos(3) - d1) / sqrt(a1^2 + ...
               b1^2 + c1^2);
111         cog_2_motor_center_plane(j) = abs(a1_prime*cog_pos(1) ...
               + b1_prime*cog_pos(2) + c1_prime*cog_pos(3) - ...
               d1_prime) / sqrt(a1_prime^2 + b1_prime^2 + ...
               c1_prime^2);
112     end
113 arm_plane_normal_angle(j) = ...
       acos(dot(cog_2_m_inI(:,j),norm_vec)/(norm(cog_2_m_inI(:,j))*norm(norm_vec)));
114
115     if arm_plane_normal_angle(j) ≤ pi/2
116         arm_angle_flag(j) = 1;
117     else
118         arm_angle_flag(j) = -1;
119     end
120
121 end
122
123 % parameter initialization
124 p_a = zeros(3,4);   % contact point (projection of the ...
       penetrating edge on the contact surface)
125 p_b = zeros(3,4);   % penetration vector end-point ...
       (penetrating edge)
126 pen = zeros(3,4);   % penetration vector for all bumpers
127 pen_normal = zeros(3,4);
128 d_pen = zeros(3,4);
129 k_a = zeros(3,4);
130 k_b = zeros(3,4);
131 b_a = zeros(3,4);
132 b_b = zeros(3,4);
133 tang = zeros(3,4);
134 tang_direction_normalized = zeros(3,4);
135 t_a = zeros(3,4);
136 t_b = zeros(3,4);
137 resultant = zeros(3,4);
138 p_r = zeros(3,4);
139 arm_inI = zeros(3,4);
140 moment = zeros(3,4);
141 res_force_cog = zeros(3,1);
```

```
142 res_moment_cog = zeros(3,1);
143 p_f = zeros(3,1);
144 p_m = zeros(3,1);
145
146 for k = 1:4
147
148     if d(k) ≤ r
149
150         if cog_2_contact_plane(k) ≥ ...
               cog_2_motor_center_plane(k) && arm_angle_flag(k) ...
               == 1
151
152             p_a(:,k) = m(:,k) + d(k) * norm_vec;
153             p_b(:,k) = m(:,k) + r * norm_vec;
154             pen(:,k) = p_b(:,k) - p_a(:,k);
155
156         elseif cog_2_contact_plane(k) < ...
               cog_2_motor_center_plane(k) && arm_angle_flag(k) ...
               == 1
157
158             p_a(:,k) = m(:,k) - d(k) * norm_vec;
159             p_b(:,k) = m(:,k) + r * norm_vec;
160             pen(:,k) = p_b(:,k) - p_a(:,k);
161
162         elseif cog_2_contact_plane(k) ≤ ...
               cog_2_motor_center_plane(k) && arm_angle_flag(k) ...
               == -1
163
164             p_a(:,k) = m(:,k) + d(k) * norm_vec;
165             p_b(:,k) = m(:,k) - r * norm_vec;
166             pen(:,k) = p_b(:,k) - p_a(:,k);
167
168         elseif cog_2_contact_plane(k) > ...
               cog_2_motor_center_plane(k) && arm_angle_flag(k) ...
               == -1
169
170             p_a(:,k) = m(:,k) - d(k) * norm_vec;
171             p_b(:,k) = m(:,k) - r * norm_vec;
172             pen(:,k) = p_b(:,k) - p_a(:,k);
173
```

```
174         end
175
176     else
177
178         p_a(:,k) = zeros(3,1);
179         p_b(:,k) = zeros(3,1);
180         pen(:,k) = zeros(3,1);
181
182     end
183
184     % spring (stiffness) force vector start and end point ...
            calculation
185     if norm(pen(:,k)) ~= 0
186         pen_normal(:,k) = pen(:,k) / norm(pen(:,k));
187     else
188         pen_normal(:,k) = [0;0;0];
189     end
190     k_a(:,k) = p_a(:,k);
191     %k_b(:,k) = p_a(:,k) - C_k * pen(:,k);
192     k_b(:,k) = p_a(:,k) - C_k * (norm(pen(:,k))^2) * ...
            pen_normal(:,k);
193
194     % damping force vector start and end point calculation
195     if arm_angle_flag(k) == 1
196         d_pen(:,k) = (pen(:,k) - pen_old(:,k)) / dt;
197     elseif arm_angle_flag(k) == -1
198         d_pen(:,k) = (pen_old(:,k) - pen(:,k)) / dt;
199     end
200     if norm(d_pen(:,k)) > norm(v_cog)
201         d_pen(:,k) = d_pen_old(:,k);
202     end
203     b_a(:,k) = p_a(:,k);
204     b_b(:,k) = p_a(:,k) - C_b * d_pen(:,k);
205
206     % Coulomb friction force (tangent to the colliding ...
            surface) calculation
207     tang(:,k) = p_a(:,k) - p_a_old(:,k);
208     if norm(tang(:,k)) >  norm(v_cog)
209         tang(:,k) = [0;0;0];
210     end
```

```
211     if norm(tang(:,k)) == 0
212         tang_direction_normalized(:,k) = [0;0;0];
213     else
214         tang_direction_normalized(:,k) = tang(:,k) / ...
                norm(tang(:,k));
215     end
216     t_a(:,k) = p_a(:,k);
217     t_b(:,k) = t_a(:,k) - mu_d * norm(pen(:,k)) * ...
            tang_direction_normalized(:,k);
218
219     % calculation of the resultant force in the contact point ...
            of each
220     % bumper
221     resultant(:,k) = (k_b(:,k) - k_a(:,k)) + (b_b(:,k) - ...
            b_a(:,k)) + (t_b(:,k) - t_a(:,k));
222
223     % calculation of the end-point of the resultant forces at ...
            each contact point
224     p_r(:,k) = p_a(:,k) + resultant(:,k);
225
226     % CoG to contact point vector in Inertial frame (moment arm)
227     if p_a(:,k) ~= [0;0;0]
228         arm_inI(:,k) = p_a(:,k) - cog_pos;
229     else
230         arm_inI(:,k) = [0;0;0];
231     end
232
233     % calculation of the resultant moment at each bumper
234     moment(:,k) = cross(arm_inI(:,k), resultant(:,k));
235
236 end
237 % calculation of the Resultant force and moment on the ...
        drone's CoG
238 res_force_cog = resultant(:,1) + resultant(:,2) + ...
        resultant(:,3) + resultant(:,4);
239 res_moment_cog = moment(:,1) + moment(:,2) + moment(:,3) + ...
        moment(:,4);
240
241 % CoG resultant force and moment vectors end-point calculation
242 %p_f = cog_pos + res_force_cog;
```

```
243 %p_m = cog_pos + res_moment_cog;
244
245 % output assignment
246 FX = res_force_cog(1,1);
247 FY = res_force_cog(2,1);
248 FZ = res_force_cog(3,1);
249 MX = res_moment_cog(1,1);
250 MY = res_moment_cog(2,1);
251 MZ = res_moment_cog(3,1);
```